drones

MDPI

Article

Joint UAV Deployment and Task Offloading Scheme for Multi-UAV-Assisted Edge Computing

Fan Li, Juan Luo *, Ying Qiao and Yaqun Li

College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China
* Correspondence: juanluo@hnu.edu.cn

Abstract: With the development of the Internet of Things (IoT), IoT devices are increasingly being deployed in scenarios with large footprints, remote locations, and complex geographic environments. In these scenarios, base stations are usually not easily deployed and are easily destroyed, so unmanned aerial vehicle (UAV)-based edge computing is a good solution. However, the UAV cannot accomplish the computing tasks and efficiently achieve better resource allocation considering the limited communication and computing resources of the UAV. In this paper, a multi-UAV-assisted mobile edge computing (MEC) system is considered where multiple UAVs cooperate to provide a service to IoT devices. We formulate an optimization function to minimize the energy consumption of a multi-UAV-assisted MEC system. The optimization function is a complex problem with non-convex and multivariate coupling. Thus, a joint UAV deployment and task scheduling optimization algorithm are designed to achieve optimal values of UAV numbers, the hovering position of each UAV, and the best strategy for offloading and resource allocation. Experimental results demonstrate that the algorithm has positive convergence performance and can accomplish more tasks under the constraint of delay compared to the two benchmark algorithms. The proposed algorithm can effectively reduce the system energy consumption compared to the two state-of-the-art algorithms.

Keywords: edge computing; task scheduling; UAV-assisted; UAV deployment

Citation: Li, F.; Luo, J.; Qiao, Y.; Li, Y. Joint UAV Deployment and Task Offloading Scheme for Multi-UAV-Assisted Edge Computing. *Drones* **2023**, *7*, 284. https://doi.org/10.3390/drones7050284

Academic Editors: Panagiotis Sarigiannidis, Abderrahmane Lakas and Mauro Tropea

Received: 30 March 2023
Revised: 15 April 2023
Accepted: 19 April 2023
Published: 22 April 2023

1. Introduction

The development of the Internet of Things (IoT) drives humanity into the era of Internet of Everything. The deployment of 5G and beyond wireless networks creates advantageous conditions for new applications. IoT devices have limited battery capacity and computing resources constrained by size and environment. Therefore, it is challenging to apply services to these IoT device applications. Mobile edge computing (MEC) [1–3], which deploys servers to provide services close to IoT devices, reduces transmission energy, and local computation, is considered to be promising in addressing the above challenges. In general, IoT devices transmit data to a base station deployed with mobile edge computing servers. The base station decides whether to execute it locally or send it to the data centre by the size of the computation.

In some remote areas such as forests, farms, etc., wired communication cables are difficult to lay and base stations are very limited due to operating costs [4]. Therefore, aerial base stations based on unmanned aerial vehicles (UAVs) have become crucial in providing content coverage for IoT networks [5–7]. UAV can not only assist base stations to collect data from IoT devices but also replace base stations as edge servers to provide computing services for IoT devices. Therefore, the stability of UAVs is also critical in the execution of tasks. The current advanced control algorithm can theoretically solve this problem [8,9], so the UAV can realize the precise and massive deployment of base stations in remote areas.

Currently, smart agriculture, which is a typical scenario of IoT networks, requires UAVs to collect data and perform tasks, as shown in Figure 1. If the deployment number of UAVs is small, smart agriculture has the following two problems: (1) It is difficult to

provide services to a large range of IoT devices due to the limited coverage of individual UAVs. (2) UAVs have limited battery capacity and short endurance, and cannot handle tasks effectively. Therefore, smart agriculture generally adopts multi-UAV collaboration to collect data and provide computing services to IoT devices. Moreover, the collaboration of multi-UAV can accomplish the task more effectively [10].

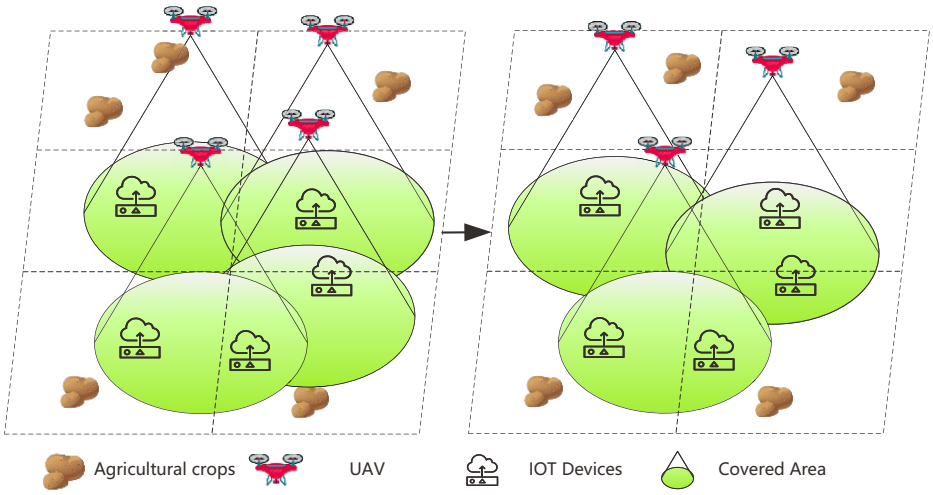


Figure 1. The multi-UAV edge computing scenario.

However, collaboration among UAVs in smart agriculture also faces many new challenges. First, the service range, communication, and computing capabilities of UAVs are limited by their battery capacity, communication coverage, and carrying capacity [11]. Second, due to mobility and changes in the surrounding environment, UAVs tend to collide with each other [12], resulting in the unsuccessful offloading of IoT devices. Finally, when UAVs are deployed, the number and location of UAVs affect the EC of UAV-assisted mobile edge computing system. Hence, in our paper, we propose a joint optimization algorithm for multi-UAV deployment and task scheduling to obtain the optimal number of UAVs, the hovering position of each UAV, and a strategy for offloading and resource allocation. The contributions of our work according to the above challenges are as follows:

- Aiming at the difficulty of providing services to IoT devices in scenarios where remote areas lack base stations, we propose a multi-UAV-assisted MEC system. Compared with the traditional fixed number of UAVs, the system optimizes the number and location of UAVs according to the location and number of IoT devices. Optimizing the number of UAVs can effectively reduce the system energy consumption (EC) and reduce the deployment cost of purchasing UAVs.
- To improve the service time of multi-UAV-assisted mobile edge computing systems and reduce system energy. We aim to cover the maximum number of IoT devices with the minimum number of UAVs. Thus, we formulate a non-convex optimization problem and propose a parametric adaptive differential evolution (PADE) algorithm to optimize the hovering position of the UAVs.
- We present extensive simulation results to evaluate the performance of our proposed algorithm. We have classified the different IoT device offloading tasks. Specifically, we employ eight use cases to verify that our algorithm achieves task completion rates subject to latency constraints and reduces average system EC. Compared with existing algorithms, the proposed PADE algorithm is verified to be effective in reducing the overall system EC.

2. Related Work

In some scenarios such as earthquakes, traffic jams, and remote areas, when the ground infrastructure is damaged and the computing and communication resources of

edge facilities are insufficient, it is difficult for IoT devices to benefit from the quality of service [13]. To improve the quality of service for IoT devices, UAVs are widely studied by many scholars as edge nodes to collect and process data [14]. UAVs as a mobile edge computing system face the challenges of energy consumption, latency, and task offload capacity [15].

UAV deployment: Limited EC is a significant challenge for UAVs to provide services. It is important to conserve energy for UAVs to perform their tasks efficiently. In smart agriculture, UAVs need to hover to collect, process, and then fly to another location. In recent years, a lot of work has been conducted in reducing the EC of the multi-UAV-assisted system, focusing on the EC of UAV and optimization of EC of IoT devices. The EC of the UAVs determines the period of service in the system. Hu et al. [16] maximized UAV energy efficiency by optimizing UAV flight trajectories and offload/cache decisions. Offloading tasks to other UAVs using collaboration between UAVs can reduce the EC of individual UAVs [17,18]. Optimizing the location of UAVs, maximizing the throughput from IoT devices to UAVs, and reducing UAV flight time can improve the energy efficiency of UAVs. Jing et al. [19] investigated the minimum trade-off between EC and training delay by jointly optimizing the location and resource allocation of the UAV. To further reduce the EC of UAVs, Guo et al. [20] optimized the number of UAVs, using fuzzy C-means (FCMs) to optimize the location of the UAVs and to ensure the stability of the service by replacing the UAVs. In addition, after the deployment location of the UAV was determined, the EC of the UAV-assisted mobile edge computing system could be effectively reduced by task offloading and resource allocation.

Resource Scheduling: Latency is an important indicator of user quality of service (QoS), and resource scheduling can reduce task latency effectively. A UAV-assisted mobile edge computing network is designed to optimize data transmission latency and computation latency to ensure that tasks are completed within a constrained time. To increase the transmission rate of the terminal, Hou et al. [21] obtained high reliability with low latency by joint optimization of computation, communication, and cache resource scheduling. To further improve the utilization of network resources, Chen et al. [22] optimized the UAV movement and MU association with deep reinforcement learning to reduce EC and system latency. Offloading tasks to appropriate UAVs can reduce local computational EC. Hoang et al. [23] used Lyapunov optimization to transform the original multi-stage problem into a problem at each time point, which was then solved using the DRL framework. Generally, optimizing the transmission rate of devices and scheduling tasks can effectively improve the reduction in latency and EC of devices.

The current study focuses on providing services for a fixed number of UAVs and then reducing system EC by optimizing network and computing resources. In practice, IoT devices receive computing resources, network transmission limitations, and other factors, so the offloading factor of end devices is considered. In this paper, we iterate the number of UAVs and user offloading decision factors to obtain a better UAV deployment location and user offloading decision. Considering the actual situation of IoT devices, we divide the tasks into multiple categories to simulate the offloading of different tasks. A comprehensive comparison of our proposed scheme with existing methods is shown in Table 1.

Table 1. Overview of current state-of-the-art technologies.

References	Number of UAVs	Optimized Objective	Method	Task Rank
[16]	Fixed	EC	Lagrange and SCA	Single
[17]	Fixed	EC	Lyapunov	Single
[18]	Fixed	EC & delay	machine learning	Multiple
[20]	Optimization	Load Balancing	fuzzy *C*-means & BP neural network	Single
[19]	Optimization	EC and training time	SCA	Single
[21]	Fixed	latency	HBPSO	Single
[22]	Fixed	latency and EC	Reinforcement Learning	Single
[23]	Fixed	average EC of the system	Lyapunov ad Deep Reinforcement Learning	Single
Ours	Optimization	EC and latency	PADE & Greedy	Multiple

3. System Model

As edge servers collect sensor data and perform related computational tasks, UAVs have become a hot research topic for scholars [24–26]. As servers handle the tasks offloaded by IoT devices, UAVs are equipped with a certain amount of computing and storage resources when they act as an edge server to collect sensor data or perform terminal tasks. IoT devices can choose the right UAV server to offload according to their needs. The UAV plans the flight route and resource allocation according to its situation and IoT device location.

In smart agriculture, we consider a multi-UAV-assisted edge computing system to collect and process agriculture data. We use $\mathcal{K} = \{1, \ldots, K\}$ and $\mathcal{N} = \{1, \ldots, N\}$ to denote the set of IoT devices and UAVs. N UAVs provide computing resources to K IoT devices, as shown in Figure 2. We use a three-dimensional (3D) Cartesian coordinate representation to represent the specific locations of UAVs and IoT devices. K ground-based IoT devices are randomly distributed over a horizontal plane. The locations of IoT devices are denoted as $w_k = (x_k, y_k)$, whereas the locations of n-th UAVs can be denoted as $L_n = (X_n, Y_n, H), n \in \mathcal{N}$, where H is a fixed height above the ground. The main notations are illustrated in Table 2.

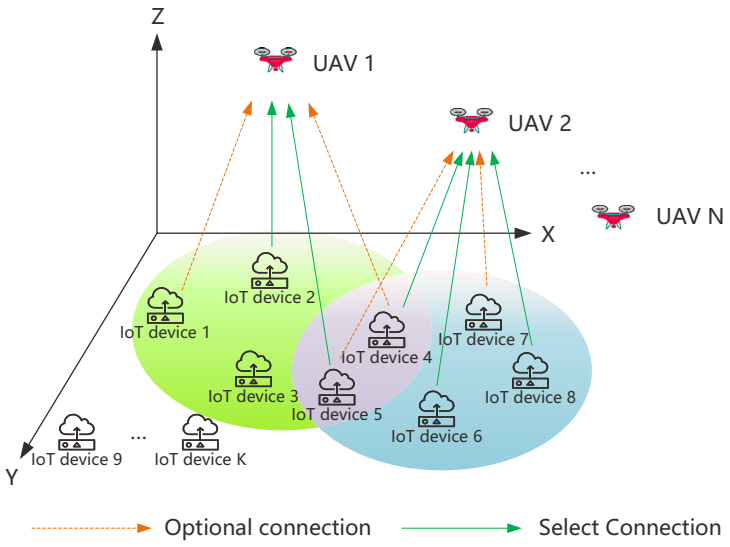


Figure 2. System Model.

Table 2. Main Notations.

Notation	Description
w_k, L_n	The locations of k-th IoT device and the locations of n-th UAV
N	Number of IoT devices
$b_{k,a}$	Offloading decision of the k-th device
η_1, η_2	Effective switched capacitance
d_{min}^{uu}	Minimum distance between two UAVs
d_{u_1,u_2}^{uu}	The distance between the UAV u_1 and UAV u_2
n_{max}	Maximum number of tasks each UAV can serve
R, θ	UAV coverage radius and signal coverage angle of the UAV
H	The height of the UAV
$h_{k,n}$	The uplink channel gain from the k-th IoT device to the n-th UAV
μ_0	Channel power gain at the unit distance
σ^2	The noise power of the UAV
C_k	Number of CPU cycles to complete the task of the k-th IoT device
D_k	Size of the k-th IoT device's task data
B	Channel bandwidth
$r_{k,n}$	The uplink data rate from the k-th IoT device to the n-th UAV
$t_{k,0}^{local}$	the time required to complete its task
$f_{k,0}$	The computation resources of k-th IoT devices
$E_{k,0}^{local}$	The locally calculated EC
$f_{k,a}$	The computation resources that the UAV a allocates to IoT device k
$t_{k,a}^{uav}$	The total consuming time contains transmission time and execution time on UAV
$E_{k,a}^{uav}$	The total EC of the k-th IoT device
P	Transmission power of IoT device
E^H	Hovering energy consumption of UAVs
T, P_0	Hovering time and hovering power

We use $\mathcal{A} = \{0, \ldots, N\}$ to denote the set of IoT devices' offloading decisions. We indicate the offloading decision of the k-th device by $b_{k,a}, k \in \mathcal{K}, a \in \mathcal{A}$. $a = 0$ indicates that the IoT device performs the task locally, and $a > 0$ means that the IoT device chooses to offload the task on the n-th UAV.

Proximity can lead to accidents when UAVs are deployed [27]. For any two UAVs u_1 and u_2 obtain:

$$d_{u_1,u_2}^{uu} \geq d_{min}^{uu}, \forall u_1, u_2 \in N, u_1 \neq u_2, \tag{1}$$

where d_{u_1,u_2}^{uu} is the distance between u_1 and u_2, and d_{min}^{uu} is the minimum distance among them.

3.1. Communication Model

We adopt an orthogonal frequency division multiple access (OFDMA) [28] in the system communication model to implement the link communication between the UAV and the device. The IoT devices offload their computational tasks to the UAV with equal bandwidth allocation. In the system, each IoT device decides whether to offload tasks to the UAV. The UAV has limited energy. The maximum total number of tasks performed by each UAV should satisfy the following constraint:

$$\sum_{k=1}^{K} b_{k,a} \leq n_{max}, \forall k \in \mathcal{K}, a \in \mathcal{A}/\{0\}, \tag{2}$$

where n_{max} denotes the maximum number of tasks performed by the n-th UAV.

For simplicity, we assume that the communication from the IoT device to the UAV is link by the line-of-sight [29], where the channel quality depends on the UAV–IoT device distance. Then, the uplink channel gain from the k-th IoT device to the n-th UAV is described by the following expression:

$$h_{k,n} = \mu_0 (d_{k,n})^{-2} = \frac{\mu_0}{H^2 + \|w_k - N_n\|}, \tag{3}$$

where μ_0 is the channel power gain at the unit distance. $d_{k,n}$ denotes the distance between the IoT device and the UAV, and $d_{k,n}^h$ denotes the distance in the horizontal direction from the k-th device to the n-th UAV. If the k-th device performs a task on the n-th UAV, then the k-th device must be within the communication range of the UAV, i.e., the following constraint is satisfied:

$$b_{k,a} d_{k,n}^h \leq R, \forall k \in \mathcal{K}, n \in \mathcal{N}, a = n, \tag{4}$$

where $R = H \tan\theta$ is the UAV coverage radius, and θ is the signal coverage angle of the UAV. The uplink transmission data rate of the k-th device in offloading decision a can be expressed as:

$$r_{k,n} = B \log_2 \left(1 + \frac{P h_{k,n}}{\sigma^2}\right), \tag{5}$$

where σ^2 denotes the noise power of the UAV and B is the channel bandwidth.

3.2. Computing Model

Each device has a computational task U_k to be completed, which is denoted by a binary $U_k = (C_k, D_k), k \in \mathcal{K}$, where C_k denotes the total number of CPU cycles required to complete the task of the k-th IoT device and D_k denotes the size of the k-th IoT device's task data.

3.2.1. Local Computing Model

When the k-th IoT device performs a task locally, the time required to complete its task is:

$$t_{k,0}^{local} = \frac{C_k}{f_{k,0}}, \forall k \in \mathcal{K}, \tag{6}$$

where $f_{k,0}$ is the computation resources of the k-th IoT devices, which is constant. The locally calculated energy consumed is:

$$E_{k,0}^{local} = \eta_1 (f_{k,0})^2 C_k, \ \forall k \in \mathcal{K}, \tag{7}$$

where η_1 is the effective switched capacitance of IoT devices.

3.2.2. UAV-Assisted Edge Computing

We use $f_{k,a}$ to denote the computation resources that the UAV a allocates to the IoT device k. If the k-th device offloads the task to the n-th UAV for execution, the total time consumption contains the transmission time and execution time on the UAV, which can be denoted as:

$$t_{k,a}^{uav} = \frac{D_k}{r_{k,a}} + \frac{C_k}{f_{k,a}}, \ \forall k \in \mathcal{K}, \ a = n. \tag{8}$$

The total EC of the k-th IoT device also contains the transmission energy and execution EC, which can be defined as:

$$E_{k,a}^{uav} = P \frac{D_k}{r_{k,a}} + \eta_2 (f_{k,a})^2 C_k, \forall k \in \mathcal{K}, \ a = n, \tag{9}$$

where P is the transmission power of the IoT device, and η_2 is the effective switched capacitance of the UAV.

3.2.3. UAV Hovering Model

When the UAV stays in a certain position, the energy required for the period of its stay is expressed as:

$$E^H = P_0 T, \tag{10}$$

where P_0 denotes the hover power and T denotes the hover time.

3.3. Problem Formulation

We aim to improve the system performance by minimizing the system EC, including local EC, device transmission EC, UAV computation EC, and UAV hovering EC. Thus, the problem of optimizing the number of UAVs, their location, and task scheduling is expressed as:

$$\mathcal{P}1: \min_{N, w_k, b_{k,a}, f_{k,n}} \sum_{k=1}^{K} \left(b_{k,0} E_{k,0}^{local} + \sum_{n=1}^{N} b_{k,a} E_{k,n}^{uav} \right) + N E^{H} \tag{11a}$$

$$\text{s.t.} \sum_{k=1}^{K} b_{k,a} \leq n_{max}, \forall k \in \mathcal{K}, a \in \mathcal{A}/\{0\}, \tag{11b}$$

$$\sum_{a=0}^{N} b_{k,a} = 1, \forall k \in \mathcal{K}, a \in \mathcal{A}, \tag{11c}$$

$$b_{k,a} d_{k,n} \leq R, \forall k \in \mathcal{K}, \forall n \in \mathcal{N}, a = n, \tag{11d}$$

$$d_{min}^{uu} \leq d_{n1,n2}, \forall n1, n2 \in \mathcal{N}, n1 \neq n2, \tag{11e}$$

$$f_{k,a} > 0, \forall b_{k,a} = 1, \forall k \in \mathcal{K}, \forall n \in \mathcal{N}, \tag{11f}$$

$$f_{k,a} = 0, \forall b_{k,a} = 0, \forall k \in \mathcal{K}, \forall n \in \mathcal{N}, \tag{11g}$$

$$b_{k,0} t_{k,0}^{local} \leq T, \forall k \in \mathcal{K}, \tag{11h}$$

$$b_{k,a} t_{k,a}^{uav} \leq T, \forall k \in \mathcal{K}, \tag{11i}$$

Equation (11c) ensures that all tasks are executed, and each can be executed in only one mode. Equations (11h) and (11i) are the delay constraints for each task.

4. Joint UAV Deployment and Mission Scheduling Optimization Algorithm Design

Problem $\mathcal{P}1$, an optimization problem with a non-convex and multivariate coupling of objective function and constraints, is difficult to solve directly. The reason is that the IoT device offloading decision variable $b_{k,a}$ is a binary decision variable, and the number of UAVs N is an integer decision variable. The differential evolution algorithm is suitable for solving mixed integer-type planning problems. Therefore, we solve this optimization problem based on the differential evolution (DE) algorithm [14,30].

We need to optimize a total of $2N + 1$ variables when optimizing the number and location of UAVs. Each contains two coordinate variables (x, y) and the optimized number of UAVs. K IoT devices have K offloading decision variables and K UAV resource allocation variables, Thus, $(2N + 1 + 2K)$ variables should be optimized, and the time complexity becomes increasingly complex as the number of variables increases with N and K. The problem becomes a large-scale evolutionary optimization problem because of the increase in the solution size with the number of UAVs and IoT devices. Therefore, we decompose the problem $\mathcal{P}1$ into two sub-problems $\mathcal{P}2$ and $\mathcal{P}3$ for solving to decouple the variables and reduce the problem size. The overall algorithm is given in Algorithm 1. $\mathcal{P}2$ optimizes the number of UAVs and hovering positions, and $\mathcal{P}3$ optimizes the task scheduling. Then, these two sub-problems are described separately.

The following describes the general framework proposed in Algorithm 1.

Algorithm 1: Overall framework design

input : Parameters K, w_k, U_k, $step_{max}$
output: $\{N, \mathcal{S}, \mathcal{B}, \mathcal{F}\}$.

Initialize the UAV location population $\mathcal{S}$ under constraint (11e), $flag = 0$, $step = 1$, $unf\_num = 0$.
According to $\mathcal{S}$ and Algorithm 2, $b_{k,a}$ and $f_{k,a}$ are obtained.
while $step = 1, \ldots, step_{max}$ **do**
    while $flag = 0$ *and* $\{N, \mathcal{S}, \mathcal{B}, \mathcal{F}\}$ *is Available* **do**
        $\{N_t, \mathcal{S}_t, \mathcal{B}_t, \mathcal{F}_t\} = \{N, \mathcal{S}, \mathcal{B}, \mathcal{F}\}$;
        The number of UAVs is adjusted, and two individuals with the smallest distance from the population $\mathcal{S}$ are selected;
        Their second minimum Euclidean distance is calculated, and the individual with the smallest second minimum Euclidean distance is deleted;
        If they have equal second minimum Euclidean distances, then their third minimum Euclidean distance is calculated;
        The same procedure follows for the succeeding minimum Euclidean distances;
        $N = N - 1$;
        According to $\mathcal{S}$ and Algorithm 2, $b_{k,a}$ and $f_{k,a}$ are obtained ;
        Calculating the new system EC ;
    The variation and crossover operations of the PADE algorithm are performed to generate offspring populations $\mathcal{S}_l$;
    for $i = 1, \ldots, N$ **do**
        Update $\{N, \mathcal{S}, \mathcal{B}, \mathcal{F}\}$ with the i-th individual in $\mathcal{S}_l$ using the update operator;
        if $\{N, \mathcal{S}, \mathcal{B}, \mathcal{F}\}$ *is not available* **then**
            $unf\_num = unf\_num+1$;
            if unf_num==*1000* **then**
                return $\{N, \mathcal{S}, \mathcal{B}, \mathcal{F}\}$ **else**
                unf_num=0 ;

4.1. UAV Deployment

UAV deployment includes UAV number and location optimization. For UAV location optimization, we adopt the PADE algorithm with improved parameters to solve the problem in the present study.

4.1.1. UAV Position Optimization

In the problem of optimizing the UAV's hovering positions, the flight altitude of the UAVs is set as a constant. Thus, each UAV has two variables to be optimized: X_n and Y_n. Existing works use an evolutionary algorithm to solve the UAV location coordinate problem, which deploys the entire UAV location as an individual. In our problem, the length of an individual is $2N$, and the number of N changes at each iteration, thereby complicating the evolutionary algorithm. We find that the coordinates X_n and Y_n of UAVs have the same length, and each take the same range of values. Moreover, we can encode the hovering position of each UAV as an individual and the hovering positions of all UAVs as a population. Thus, the length of each individual is 2, and the variable length optimization problem is successfully turned into a fixed-length optimization problem. At this time, the problem size of $\mathcal{P}2$ becomes $N + 1$.

Initializing the population:

First, we initialize the population to solve the problem $\mathcal{P}2$. The feasible domain randomly and uniformly generates a hovering position of the UAV to record it into the initial population $\mathcal{S}$. Then, the same approach is followed to generate the hovering position of the second UAV and determine whether the UAV and the other UAVs in the group $\mathcal{S}$ satisfy Equation (11e). Hence, the distance between the two UAVs is guaranteed to be safe. Moreover, no collision can cause damage to the UAV. If satisfied, then it is put into $\mathcal{S}$; otherwise, the generated UAV hovering position is not suitable and needs to be regenerated. When the number of consecutive failed attempts is greater than a set constant, initialization restarts until the hover positions of all UAVs are successfully obtained and the initial deployment of the UAVs is obtained (i.e., $\mathcal{S}$).

Mutation and crossover:

In PADE, mutant individuals $x_i = (x_{i,1}, x_{i,2}), i \in \{1, \ldots, N\}$ are generated for each individual in the v_i population as follows:

$$v_i = x_{r1} + F(l) * (x_{r2} - x_{r3}), \tag{12}$$

where l is the number of iterations, and $r1$, $r2$, and $r3$ are three randomly selected individuals. Each of these individuals is different from the population $\bar{\mathcal{S}}_l$, which determines the direction of mutation. $f(l)$ is the scaling factor, which is set as a constant in most studies. In the present study, the scaling factor is transformed into the dynamic adaptive adjustment of the parameters. The adaptive differential evolution algorithm of [31] is better approximated. According to this, we rewrite the scaling factor as:

$$F = \alpha\left(e^{\beta} - 1\right), \tag{13}$$

$$\beta = \frac{l_{max}}{l_{max} + G}, \tag{14}$$

where G indicates the current generation of evolution, and l_{max} denotes the maximum number of iterations. α is used to adjust the scaling factor, $\alpha \in (0.2, 0.6)$.

A crossover operation is applied after the mutation operation to fuse the mutant individuals with the corresponding original individuals.

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if } rand_j(1,0) \leq CR \text{, or } j = j_{rand}; \\ x_{i,j}, & \text{otherwise,} \end{cases} \tag{15}$$

where $\bar{u}_i = (u_{i,1}, u_{i,2})$ is the experimental individual, and $u_{i,j}, v_{i.j}, x_{i.j}$ is the j-th dimension of $\bar{u}_i, \bar{v}_i$ and $\bar{x}_i$. CR is the crossover probability that can be found by the following expression:

$$CR = 0.5 * (1 + rand), \tag{16}$$

where $rand$ is a uniformly distributed random number and $rand \in [0, 1]$.

Selection operation:

After the experimental population is obtained by the variational crossover operation, the selection operation is used to select individuals from the experimental population $\bar{\mathcal{S}}_{l+1}$. The original population $\bar{\mathcal{S}}_l$ is used to generate a new population $\bar{\mathcal{S}}_{l+1}$ for the next iteration. For the new population, if (11e) is satisfied, then one can compute the other two variables, i.e., the offloading decision and resource allocation variables are replaced with the new population if $\{N,\ \bar{\mathcal{S}}_{l+1}, \mathcal{B}, \mathcal{F}\}$ can accomplish the number of tasks greater than that in the last $\{N,\ \bar{\mathcal{S}}_l, b, f\}$ under the constraint or if the EC is small.

4.1.2. UAV Number Optimization

Attribute 1: The number of UAVs should be as small as possible when all tasks are guaranteed to be completed under the constraint.

Optimizing the number of UAVs is equivalent to optimizing the variable N. In the present study, the number of UAVs should be as small as possible when all tasks are guaranteed to be completed under the delay constraint to reduce the total consumption of flight energy of all UAVs in the system, reduce deployment costs, and simplify IoT device offloading decisions.

4.2. Task Scheduling Optimization

$\mathcal{P}3$ aims to optimize the task scheduling with a known number of UAVs and hovering positions, including IoT device offloading decisions and resource allocation. At this point, the optimization problem $\mathcal{P}1$ is only related to the offloading decision $b_{k,a}$ and the resource allocation $f_{k,a}$. The problem size is $2M$. Thus, (7) and (9) can be brought into (11a), and the optimization problem can be rewritten in the following form:

$$\begin{aligned} \mathcal{P}3.1 \min_{b_{k,a}, f_{k,a}} & \sum_{k=1}^{K} \left(b_{k,0} \eta_1 (f_{k,0})^2 C_k + \sum_{n=1}^{N} b_{k,a} \left(P \frac{D_k}{r_{k,a}} + \eta_2 (f_{k,a})^2 C_k \right) \right) \\ s.t. \quad & (11b)(11c)(11d)(11f)(11g)(11h)(11i) \end{aligned} \tag{17}$$

$\mathcal{P}3.1$ shows that the total EC is monotonically increasing concerning $f_{k,a}$. Thus, $f_{k,a}$ should be as small as possible to minimize the EC. However, the delay constraints (11h) and (11i) must be ensured when execution method a is used. Thus, $f_{k,a}$ must be not smaller than the minimum value calculated based on (11h) and (11i). We can obtain the following by bringing (6) and (8) into Equations (11h) and (11i):

$$\begin{cases} f_{k,a} \geq \dfrac{C_k}{T}, & a = 0, k \in \mathcal{K}; \\ f_{k,a} \geq \dfrac{C_k}{T - D_k / r_{k,a}}, & a \in \mathcal{A} \backslash \{0\}, k \in \mathcal{K}. \end{cases} \tag{18}$$

From (18), the optimal resource allocation $f_{k,a}$ can be expressed in terms of the smallest resource allocation:

$$f_{k,a}^* = \begin{cases} \dfrac{C_k}{T}, & \text{if } b_{k,a} = 1, a = 0; \\ \dfrac{C_k}{T - D_k / r_{k,a}}, & \text{if } b_{k,a} = 1, a \in \mathcal{A} \backslash \{0\}; \\ 0, & \text{others.} \end{cases} \tag{19}$$

After the optimal resource allocation is obtained, (19) can be brought into (17) to rewrite problem $\mathcal{P}1$ and find the offloading decision variable:

$$\begin{aligned} \mathcal{P}3.2 \min_{b_{k,a}, f_{k,a}^*} & \sum_{k=1}^{K} \left(b_{k,0} \eta_1 (f_{k,0*})^2 C_k + \sum_{n=1}^{N} b_{k,a} \left(P D_k / r_{k,a} + \eta_2 \left(f_{k,a}^* \right)^2 C_k \right) \right) \\ s.t. \quad & (11b)(11c)(11d). \end{aligned} \tag{20}$$

The minimum EC to complete the k-th IoT device using execution method a can also be determined, i.e., $E_{k,0}^* = \eta_1 \left(f_{k,0}^* \right)^2 C_k$, $E_{k,a}^* = P D_k / r_{k,a} + \eta_2 \left(f_{k,a}^* \right)^2 C_k$, when the optimal resource allocation is determined. Therefore, only $b_{k,a}$ needs to be optimized. At this point, problem $\mathcal{P}3$ becomes a 0–1 integer programming problem, which is solved in this study using the greedy algorithm, as shown in Algorithm 2.

Algorithm 2: Greedy strategy for solving resource allocation

1 Initialize the offload decision variable $\mathcal{B} = 0$;
2 The tasks are divided into three levels: K1, K2, and K3 ;
3 For K1 tasks: $b_{1,0} = \ldots = b_{K1,0} = 1$;
4 For K2 tasks: the task with the least candidate offloading decision in the second class is selected (by assuming that it is the m-th task). If this task has s candidate offloading decisions and the minimum EC corresponding to the s candidate offloading decisions is $E_{s1}, E_{s2}, \ldots, E_{ss}$, then the candidate mode with the least EC among them is selected, denoted as s'. Then, $b_{K1+m,s'}$ is set to 1, and the task is removed from the second category;
5 The number of serviceable tasks of the s' UAV is subtracted by 1, and the set of candidate execution methods for the remaining tasks in the second class is updated to find the offloading decision for all tasks in the second class. The same procedure follows for the succeeding classes;
6 The number of serviceable tasks of the s' UAV is subtracted by 1, and the set of candidate execution methods for the remaining tasks in the second class is updated. This continues to find the offloading decision for all tasks in the second class;
7 For K3 tasks, the k-th IoT device in the third class of tasks is assumed to have s candidate execution methods. The minimum EC corresponding to the s execution methods are also assumed to be $E_{s1}, E_{s2}, \ldots, E_{ss}$. First, the task with the least offloading decision and the least EC (assumed to be the i-th task in the third class) is selected. Its offloading decision s' is determined;
8 $b_{K1+K2+i,s'}$ is set to 1, and the task is removed from the third class. The number of serviceable tasks of the s' UAV is subtracted by 1, and the set of the candidate execution methods for the remaining tasks in the third category is updated. The same procedure follows for the succeeding categories.

The greedy algorithm makes the choice that seems best at the moment when solving the problem at each step. However, this choice is not necessarily the globally optimal solution when viewed as a whole. Considering it is not always possible to find the global optimal solution. The greedy algorithm does not have to consider the global at each step, only considering whether the solution is an optimal solution at the moment. It integrates the local optimal solution at each step to obtain a solution to the original problem. Compared with other methods, such as branching constraints, the greedy algorithm can reduce the algorithm running time.

In Algorithm 2, we define a set of candidate execution methods for each task. The task can be executed in each candidate execution method and satisfy the delay constraint. In particular, if the execution method is 0, then the EC of all other execution methods is smaller than the EC under the 0 method. Second, all tasks are divided into three classes. The set of candidate execution methods for the first class of tasks contains only local offloading decisions. The set of candidate execution methods for the second class of tasks does not contain local offloading decisions, and the set of the third class of tasks contains 0 and other offloading decisions. If K exists and each class has K_1, K_2, and K_3 tasks, then a total of $K = K_1 + K_2 + K_3$ tasks exist.

The tasks of the first rank can only be executed locally. Thus, the offloading decision variable can be determined as $b_{1,0} = \ldots = b_{K1,0} = 1$. Completing as many tasks as possible is the goal for the tasks of the second rank. Thus, the task with the least number of candidate offloading decisions can be selected by comparing the minimum EC to choose one candidate execution method for the task. The tasks of the third rank are similar to those of the previous rank, ensuring that the system completes all tasks with minimum EC, preferring tasks with the least candidate execution methods and EC.

5. Simulation Results

The experimental simulation scenario is an N UAV-assisted edge computing system. The number of IoT devices is large and widely distributed in a multi-UAV scenario to verify the scalability of the multi-UAV-assisted edge computing system. Eight sets of use cases containing different numbers of IoT devices are set in the experiments to verify the effectiveness of the proposed algorithm. All IoT devices are assumed to be randomly distributed in a square area with different side lengths. The specific values of the distribution range of each use case are given in Table 3. Each IoT device has a task U. The number of CPU cycles required to complete U is taken between 16 and 1600 M cycles. The size of the task U is between 10 and 1000 kB. The maximum value of the computational resource $f_{k,0}$ executed locally is 0.8 GHz, and the maximum value of the computational resource executed on the n-th UAV is 10 GHz.

Table 3. The number of IoT devices and the edge length of the IoT device distribution area.

K	100	200	300	400	500	600	700	800
Side length (m)	300	450	550	650	700	780	840	900

Other specific simulation experimental parameters are presented in Table 4. The experiments were performed on MATLAB 2020.

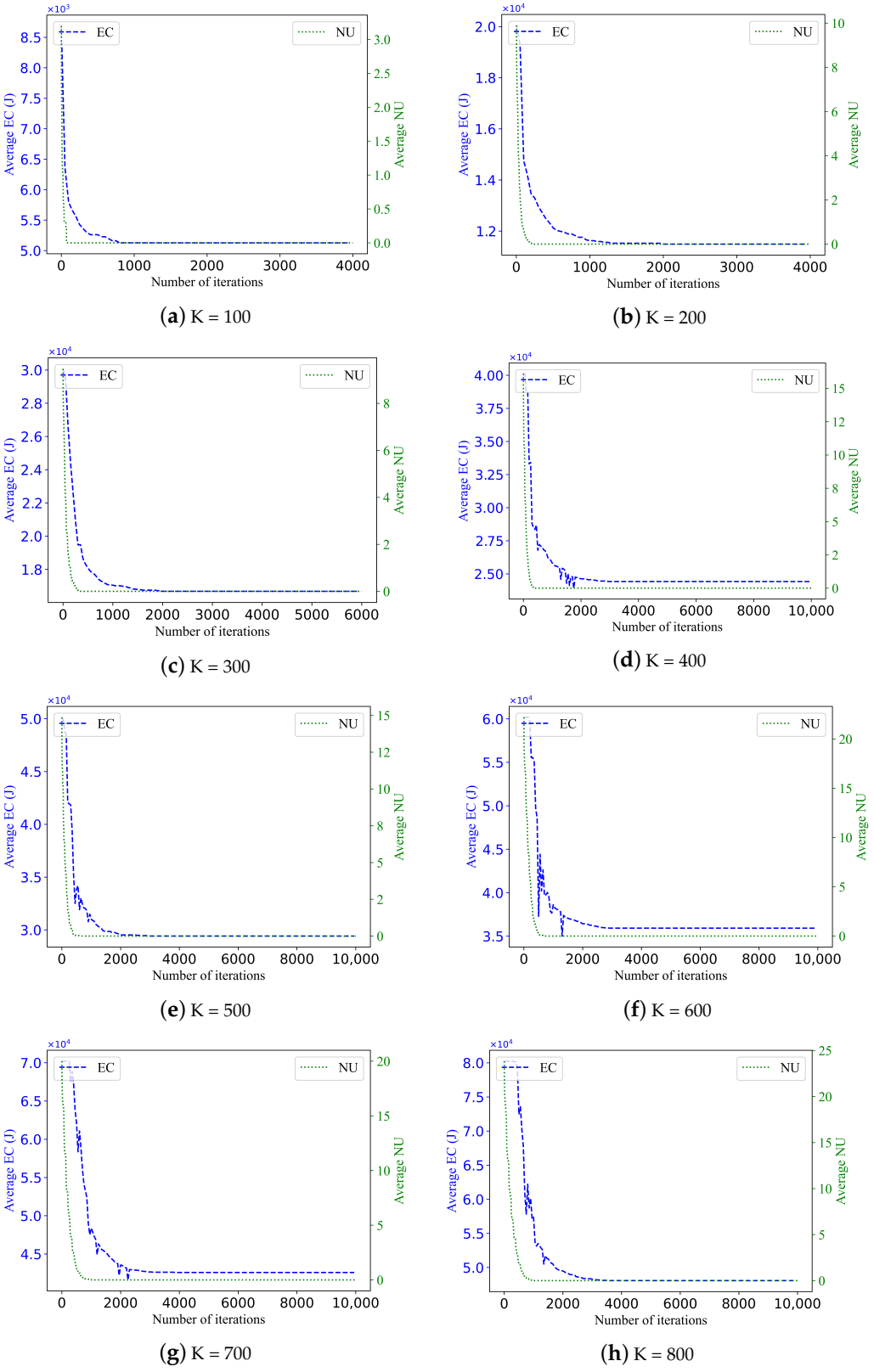
Table 4. Simulation parameters.

Description	Parameter and Value
Noise power	$\sigma^2 = -115$ dbm
Channel power gain at unit distance	$\mu_0 = 1.42 \times 10^{-4}$
Bandwidth	$B = 1$ MHz
Effective switched capacitance	$\eta_1 = \eta_2 = 10^{-27}$
Minimum distance between two UAVs	$d_{min}^{UU} = 10$ m
Maximum number of tasks each UAV can serve	$n_{max} = 10$
Transmission power of IoT device	$P = 1$W
Hovering time	$T = 1$s
Hovering power	$P_0 = 1$ kW
UAV height	$H = 100$ m

5.1. Convergence and Validity Analysis of the Algorithm

In this section, simulation experiments were conducted for all eight use cases. The average EC and the average number of uncompleted tasks (NU) of each instance, calculated after running the eight use cases 30 times, are used as performance metrics.

We experimented with the EC and convergence of the NU for different size regions. Figure 3 shows the trend of the average EC and average NU for the eight use cases with the number of iterations. The horizontal coordinate represents the number of iterations. The left vertical coordinate represents the average EC, and the right vertical coordinate represents the average number of NU. Figure 3 shows that the number of NU for all use cases eventually converges to 0, i.e., the algorithm can complete all tasks. The average EC of each use case fluctuates at first but stabilizes after 4000 iterations. It eventually converges, thereby showing the convergence of the proposed algorithm.



(**a**) K = 100 (**b**) K = 200

(**c**) K = 300 (**d**) K = 400

(**e**) K = 500 (**f**) K = 600

(**g**) K = 700 (**h**) K = 800

Figure 3. Trend of their average EC and the average number of outstanding tasks after running different instances.

5.2. Performance Analysis of the PADE Algorithm

The performance of the proposed parametric adaptive differential carry-out algorithm is compared with two other commonly used parameter setting schemes for differential evolution algorithms to verify the performance of the algorithms.

INaaF: Iteration number as a function (INaaF) of the scale factor [32]. The scale factor is set as a function of the number of iterations and is calculated as follows:

$$F(l) = \begin{cases} F_0 * 2^{\lambda_1}, & \text{if } 0 \le 1 \le l^*; \\ F_0 * (1-\lambda_2)^2, & \text{otherwise,} \end{cases} \tag{21}$$

where $\lambda_1 = \exp\left(1 - l^*/(l^* - l + 1)\right)$, $\lambda_2 = (l - l^*)/(l_{max} - l^*)$, $F_0 \in [0,1]$ is constant, and $l^* \in [0, l_{max}]$ is the predetermined division point of the segment definition function $F(l)$.

FVaaF: Fix-value as a function (FVaaF) of the scale factor, F = 0.9 and CR = 0.9 [14].

In Figure 4, we compare the impact of the improved difference-based algorithm proposed in this study with the differential evolution algorithm under the above two approaches on the system's EC. The EC of different instances calculated by the designed parametric adaptive differential carry-out algorithm is lower than that calculated by INaaF and FVaaF, thereby reducing the average EC of the system. The most significant reduction in EC is approximately 16% at several 100 IoT devices. The reason is that we use a parametric adaptive differential algorithm to optimize different numbers of IoT devices. Moreover, the INaaF algorithm had difficulty in achieving better results with the number of iterations, and the FVaaF algorithm with fixed parameter values was the worst, indicating that the performance of the algorithm can be improved by using reasonable parameter values.

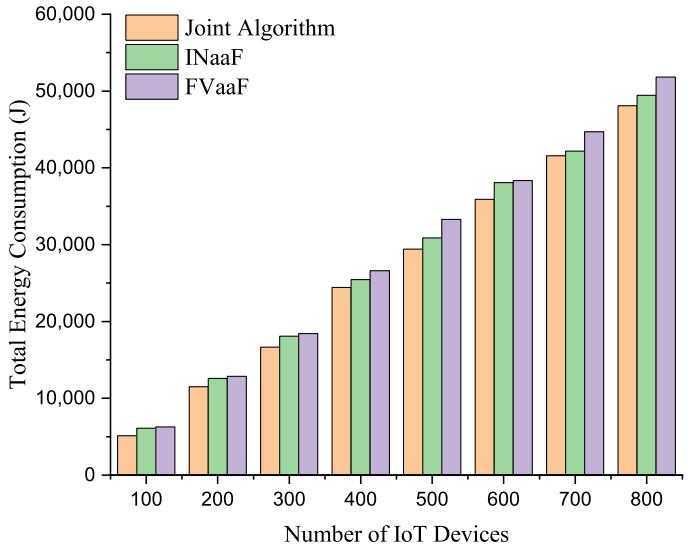


Figure 4. Comparison of total EC for different algorithms.

5.3. Comparison of Different Offloading Modes

The proposed joint UAV deployment and task scheduling algorithm are compared with two other methods. Both of these comparison algorithms use the optimization method for the number and deployment of UAVs in this study.

Only local: Only local execution exists, with no computational offloading.

Only UAVs: Only UAVs provide computational resources.

Local execution only without computation offloading mainly verifies the effectiveness of the approach of introducing UAV-assisted computation when no ground base station can help the IoT device to compute. All tasks can only be executed on the UAV when considering the local computing capability to verify whether the UAV and local co-computation can improve efficiency.

Table 5 shows the comparison between the algorithm proposed in this paper and the other two approaches regarding the number of tasks completed. Table 5 indicates that the other two offloading modes cannot complete all tasks, whereas the algorithm proposed here can complete all tasks. The reason is that when the tasks can only be performed locally, a shortage of local resources occurs. Thus, the tasks are not counted in their entirety. IoT devices are too scattered or unevenly distributed when they can only be executed on UAVs. Thus, some tasks are not covered by the UAVs, and the tasks cannot be completed. This finding verifies the effectiveness of the method proposed here. The proposed method can mitigate the limitations of the other two approaches and improve system performance.

Table 5. Number of tasks completed.

Number of IoT Devices	Only Local	Only UAVs	Our's
100	42	95	100
200	86	199	200
300	142	298	300
400	200	399	400
500	262	499	500
600	306	598	600
700	332	698	700
800	400	798	800

We compared the offloading strategy of the algorithm proposed in this study with PADE-random (PADE-R), as shown in Figure 5. Our proposed algorithm uses significantly fewer UAVs in each instance than the other strategies. The reduced number of UAVs saves UAV purchase and deployment costs and reduces the total UAV hovering EC.

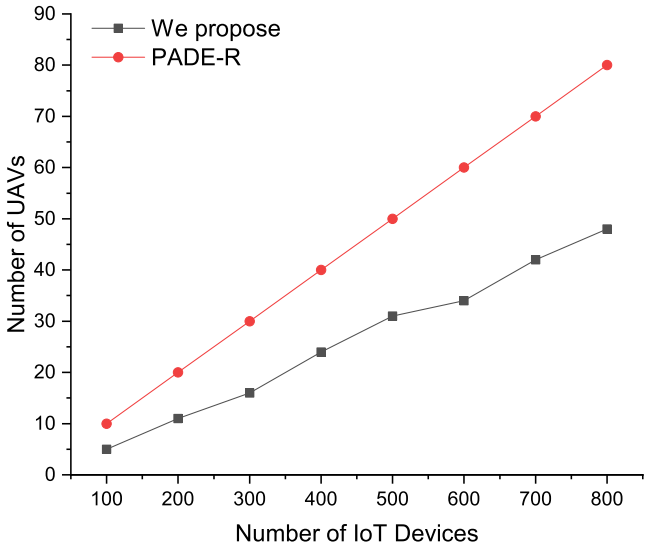


Figure 5. Comparison of the number of UAVs under the two offloading strategies.

6. Conclusions

In this study we considered a new multi-UAV-assisted mobile edge computing system for remote areas where multi-UAV can collaborate to provide services for IoT devices. With smart agriculture scenarios, we formulated a non-linear optimization objective to optimize the multiple variables of the UAV-assisted MEC system, such as UAV numbers, hovering locations, and the optimal strategy for offloading and resource allocation. Then, the optimization problem containing multiple variables was simplified into two sub-problems based on the idea of a block coordinate descent algorithm. The two sub-problems were solved using the

improved PADE and greedy algorithms. Finally, the performance of the proposed algorithm was analysed in simulation experiments and compared with two different algorithms, our algorithm's EC reduced by 16%.

In this study, the flight height of all UAVs off the ground was fixed when the UAV flight trajectory was optimized. In some scenarios, such as forests and city centres, with different heights of occlusion, fixed UAV heights may collide. We will investigate the 3D flight trajectories of different UAVs at non-fixed heights in the future to apply them accurately to different scenarios and provide reliable and stable services for IoT devices.

Property Analysis

Attribute 1 analysis: $E^*_{k,a}$ presumably denotes the minimum EC of the k-th IoT device under offloading decision a. ΔE^*_k denotes the variation of EC of the k-th IoT device under different UAV deployment scenarios. If the IoT device can execute locally and improve the EC to complete the task by offloading the task to the UAV, then the EC variation ΔE^*_k should be less than $E^*_{k,0}$, i.e., $\Delta E^*_k < E^*_{k,0} = \eta_1 \left(f^*_{k,0}\right)^2 C_k$. If the IoT device is unavailable for executing locally, the IoT device presumably executes on n-th UAV. The distance between the two is the shortest when the n-th UAV is directly above the k-th IoT device. Then, the ideal minimum EC to complete the task of the k-th IoT device can be obtained as $E^*_{min} = PD_k / r_{k,a,max} + \eta_2 \left(f^*_{k,a}\right)^2 C_k$. $r_{k,a,max} = B\log_2\left(1 + Ph_{k,n} \Big/ \sigma^2\right)$ denotes the maximum uplink rate, $h_{k,n} = \mu_0 (d_{k,n})^{-2}$, and $d_{k,n}$ denotes the distance between the IoT device and UAV, $d_{k,n} = 0$. If the k-th IoT device is located at the farthest distance between the two when the n-th UAV covers the boundary of the area, then the maximum EC to complete the task of the k-th IoT device can be obtained as $E^*_{max} = PD_k / r_{k,a,min} + \eta_2 \left(f^*_{k,a}\right)^2 C_k$. $r_{k,a,min} = B\log_2\left(1 + Ph_{k,n} \Big/ \sigma^2\right)$ denotes the minimum uplink rate, $h_{k,n} = \mu_0 (d_{k,n})^{-2}$, $d_{k,n}$ denotes the distance between the IoT device and UAV, and $d_{k,n} = Htan\theta$. Thus, in this case, $\Delta E^*_k = E^*_{max} - E^*_{min}$. We can obtain $\sum_{k=1}^{K} \Delta E^*_k < E^H$ according to the parameter settings in the simulation experiments.

This finding indicates that in different UAV deployment scenarios, the maximum EC boost for all tasks is less than the hovering EC of the UAV. Thus, the total system EC increases even though adding a*-UAV reduces the EC to complete all tasks. Therefore, the number of UAVs should be as small as possible when all tasks can be performed under the delay constraint.

Author Contributions: Conceptualization, Y.Q. and F.L.; methodology, F.L. and Y.L.; software, F.L.; validation, Y.L. and Y.Q.; formal analysis, Y.Q.; investigation, F.L.; resources, Y.Q.; data curation, Y.L.; writing—original draft preparation, F.L.; writing—review and editing, F.L.; visualization, Y.L.; supervision, J.L.; project administration, J.L.; funding acquisition, J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China under Grant 61972140, Natural Science Foundation of Chongqin under Grant ZE20210011, Key scientific and technological research and development plan of Hunan Province under Grant 2022NK2046.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

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

References

1. Chen, M.; Hao, Y. Task Offloading for Mobile Edge Computing in Software Defined Ultra-Dense Network. *IEEE J. Sel. Areas Commun.* **2018**, *36*, 587–597. [CrossRef]
2. Mach, P.; Becvar, Z. Mobile Edge Computing: A Survey on Architecture and Computation Offloading. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 1628–1656. [CrossRef]
3. Abkenar, F.S.; Ramezani, P.; Iranmanesh, S.; Murali, S.; Chulerttiyawong, D.; Wan, X.; Jamalipour, A.; Raad, R. A Survey on Mobility of Edge Computing Networks in IoT: State-of-the-Art, Architectures, and Challenges. *IEEE Commun. Surv. Tutor.* **2022**, *24*, 2329–2365. [CrossRef]
4. Khairy, S.; Balaprakash, P.; Cai, L.X.; Cheng, Y. Constrained Deep Reinforcement Learning for Energy Sustainable Multi-UAV Based Random Access IoT Networks With NOMA. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 1101–1115. [CrossRef]
5. Liu, C.; Feng, W.; Chen, Y.; Wang, C.-X.; Ge, N. Cell-Free Satellite-UAV Networks for 6G Wide-Area Internet of Things. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 1116–1131. [CrossRef]
6. Li, B.; Fei, Z.; Zhang, Y. UAV Communications for 5G and Beyond: Recent Advances and Future Trends. *IEEE Internet Things J.* **2019**, *6*, 2241–2263. [CrossRef]
7. Jia, Z.; Sheng, M.; Li, J.; Niyato, D.; Han, Z. LEO-Satellite-Assisted UAV: Joint Trajectory and Data Collection for Internet of Remote Things in 6G Aerial Access Networks. *IEEE Internet Things J.* **2021**, *8*, 9814–9826. [CrossRef]
8. Zhang, Z.; Wang, Y.; Zhang, J.; Zhang, H.; Ai, Z.; Liu, K.; Liu, F. Asymptotic Stabilization Control of Fractional-Order Memristor-Based Neural Networks System via Combining Vector Lyapunov Function with M-Matrix. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *53*, 1734–1747. [CrossRef]
9. Zhang, Z.; Wang, Y.; Zhang, J.; Ai, Z.; Liu, F. Novel stability results of multivariable fractional-order system with time delay. *Chaos Solitons Fractals* **2022**, *157*, 111943. [CrossRef]
10. Nour, E.; Boudjit, S.; Abdennebi, M.; Boubiche, D. A Flocking-Based on Demand Routing Protocol for Unmanned Aerial Vehicles. *J. Comput. Ence Technol.* **2018**, *33*, 263–276.
11. Languell, Z.P.; Gu, Q. A Multi-Point Distance-Bounding Protocol for Securing Automatic Dependent Surveillance-Broadcast in Unmanned Aerial Vehicle Applications. *J. Comput. Sci. Technol.* **2020**, *35*, 825–842. [CrossRef]
12. Yu, Z.; Gong, Y.; Gong, S.; Guo, Y. Joint Task Offloading and Resource Allocation in UAV-Enabled Mobile Edge Computing. *IEEE Internet Things J.* **2020**, *7*, 3147–3159. [CrossRef]
13. Zeng, Y.; Chen, S.; Cui, Y.; Yang, J.; Fu, Y. Joint Resource Allocation and Trajectory Optimization in UAV–Enabled Wirelessly–Powered MEC for Large Area. *IEEE Internet Things J.* **2023**. [CrossRef]
14. Wang, Y.; Ru, Z.Y.; Wang, K.; Huang, P.Q. Joint Deployment and Task Scheduling Optimization for Large-Scale Mobile Users in Multi-UAV-Enabled Mobile Edge Computing. *IEEE Trans. Cybern.* **2020**, *50*, 3984–3997. [CrossRef]
15. Zhang, Q.; Luo, Y.; Jiang, H.; Zhang, K. Aerial Edge Computing: A Survey. *IEEE Internet Things J.* **2023**. [CrossRef]
16. Hu, X.; Wong, K.-K.; Yang, K.; Zheng, Z. Task and Bandwidth Allocation for UAV-Assisted Mobile Edge Computing with Trajectory Design. In Proceedings of the IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 9–13 December 2019; pp. 1–6. [CrossRef]
17. Tang, J.; Nie, J.; Zhao, J.; Zhou, Y.; Xiong, Z.; Guizani, M. Slicing-Based Software-Defined Mobile Edge Computing in the Air. *IEEE Wirel. Commun.* **2022**, *29*, 119–125. [CrossRef]
18. Yu, J.; Vanapu, A.; Qu, C.; Wang, S.; Calyam, P. Energy-aware Dynamic Computation Offloading for Video Analytics in Multi-UAV Systems. In Proceedings of the 2020 International Conference on Computing, Networking and Communications (ICNC), Big Island, HI, USA, 17–20 February 2020.
19. Jing, Y.; Qu, Y.; Dong, C.; Ren, W.; Shen, Y.; Wu, Q.; Guo, S. Exploiting UAV for Air-Ground Integrated Federated Learning: A Joint UAV Location and Resource Optimization Approach. *IEEE Trans. Green Commun. Netw.* **2023**. [CrossRef]
20. Guo, H.; Zhou, X.; Wang, Y.; Liu, J. Achieve Load Balancing in Multi-UAV Edge Computing IoT Networks: A Dynamic Entry and Exit Mechanism. *IEEE Internet Things J.* **2022**, *9*, 18725–18736. [CrossRef]
21. Hou, X.; Ren, Z.; Wang, J.; Zheng, S.; Zhang, H. Latency and Reliability Oriented Collaborative Optimization for Multi-UAV Aided Mobile Edge Computing System. In Proceedings of the IEEE INFOCOM 2020—IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Toronto, ON, Canada, 6–9 July 2020; pp. 150–156. [CrossRef]
22. Chen, J.; Cao, X.; Yang, P.; Xiao, M.; Ren, S.; Zhao, Z.; Wu, D. Deep Reinforcement Learning Based Resource Allocation in Multi-UAV-Aided MEC Networks. *IEEE Trans. Commun.* **2023**, *71*, 296–309. [CrossRef]
23. Hoang, L.T.; Nguyen, C.T.; Pham, A.T. Deep Reinforcement Learning-Based Online Resource Management for UAV-Assisted Edge Computing With Dual Connectivity. *IEEE Trans. Netw.* **2023**, 1–16. [CrossRef]
24. Xu, Y.; Zhang, T.; Liu, Y.; Yang, D.; Xiao, L.; Tao, M. Computation Capacity Enhancement by Joint UAV and RIS Design in IoT. *IEEE Internet Things J.* **2022**, *9*, 20590–20603. [CrossRef]
25. Lu, W.; Ding, Y.; Gao, Y.; Hu, S.; Wu, Y.; Zhao, N.; Gong, Y. Resource and Trajectory Optimization for Secure Communications in Dual Unmanned Aerial Vehicle Mobile Edge Computing Systems. *IEEE Trans. Ind. Informatics* **2022**, *18*, 2704–2713. [CrossRef]
26. Faraci, G.; Grasso, C.; Schembra, G. Design of a 5G Network Slice Extension With MEC UAVs Managed With Reinforcement Learning. *IEEE J. Sel. Areas Commun.* **2020**, *38*, 2356–2371. [CrossRef]
27. Wang, Z.; Liu, R.; Liu, Q.; Thompson, J.S.; Kadoch, M. Energy-Efficient Data Collection and Device Positioning in UAV-Assisted IoT. *IEEE Internet Things J.* **2020**, *7*, 1122–1139. [CrossRef]

28. Wu, Q.; Zhang, R. Common throughput maximization in UAV-enabled OFDMA systems with delay consideration. *IEEE Trans. Commun.* **2018**, *66*, 6614–6627. [CrossRef]
29. Zhou, F.; Wu, Y.; Hu, R.Q.; Qian, Y. Computation Rate Maximization in UAV-Enabled Wireless-Powered Mobile-Edge Computing Systems. *IEEE J. Sel. Areas Commun.* **2018**, *36*, 1927–1941. [CrossRef]
30. Yang, L.; Yao, H.; Wang, J.; Jiang, C.; Benslimane, A.; Liu, Y. Multi-UAV-Enabled Load-Balance Mobile-Edge Computing for IoT Networks. *IEEE Internet Things J.* **2020**, *7*, 6898–6908. [CrossRef]
31. Fang, L.; Zhang, C.; Yi, F. Optimization Approach Base on Modified Differential Evolution Algorithm for ABF Network. *J. Chengdu Univ. Nat. Sci. Ed.* **2009**, *28*, 231–233.
32. McEnroe, P.; Wang, S.; Liyanage, M. A Survey on the Convergence of Edge Computing and AI for UAVs: Opportunities and Challenges. *IEEE Internet Things J.* **2022**, *9*, 15435–15459. [CrossRef]