Article

Extended Genetic Algorithm for Unmanned Aerial Vehicle Collaboration in Three-Layer Mobile Edge Computing Task Allocation and Optimization

Jiale Lin [1,2], Yao Nie [1,2], and Yue Chen [1,2,*]

[1] Department of Computer Science and Technology, Tongji University, Shanghai 201804, China
[2] Key Laboratory of Embedded System and Service Computing, Ministry of Education, Shanghai 201804, China
[*] Correspondence: chenyue_j@tongji.edu.cn

Received: 17 January 2025
Accepted: 11 March 2025
Published: 17 September 2025

Abstract: In recent years, low-latency tasks such as real-time communication, virtual reality, and augmented reality have higher latency requirements. Since users have relatively weak computing power, the past practice was to offload low-latency tasks to base stations (BSs) with more powerful computing power for execution, which can reduce computing latency in real-time communication. Due to the inflexible deployment of BSs and the uneven distribution of computing resources, users often find it difficult to meet the needs of low-latency tasks in places that lack BSs, so unmanned aerial vehicles (UAVs) are needed to alleviate this problem. UAVs can serve as communication relays between users and BSs and provide computing services directly to users. However, when there are few BSs, UAVs may experience load imbalance, that is, some UAVs are overloaded while other UAVs are idle. To solve this problem, we propose a UAV collaborative three-layer mobile edge computing (MEC) system model and describe the problem as a mixed integer nonlinear programming (MINLP) problem. Then, we propose an extended genetic algorithm for UAV collaboration in three-layer MEC task allocation and optimization (EGAC-3MEC) where the three-layer structure includes users, UAVs, and BSs. The proposed EGAC-3MEC achieves reasonable allocation of tasks and resources through offloading and resource allocation decisions, reduces system delay and energy consumption as much as possible, and prioritizes the delay of high-priority users. Experimental results show that the algorithm in this paper converges quickly when various parameters change, achieves better performance than the baseline algorithms when the user task size is different, and moreover, the necessity of UAV collaboration is proven.

Keywords: MEC; UAV collaboration; task offloading; resource allocation; MINLP; EGAC-3MEC; priority

1. Introduction

In recent years, 5G communications have paved the way for technologies like virtual reality (VR), augmented reality (AR), and self-driving cars. These technologies rely heavily on low latency to ensure real-time interaction and immersive experiences [1]. For instance, VR and AR applications require the minimal latency to synchronize visual and auditory feedback with user movements, while self-driving cars demand ultra-low latency to react to changing road conditions instantaneously. Therefore, optimizing time delay in cellular networks has become crucial to meet the demanding requirements of these emerging technologies. Cellular networks can provide wide-area coverage mobile communication services, and are composed of base stations (BSs) and corresponding wireless communication equipments such as users and unmanned aerial vehicles (UAVs). BSs deployed with mobile edge computing (MEC) servers can provide computing, storage, and other services. Compared with users, BSs have more powerful computing and bandwidth resources. Users with low latency requirements can transmit their computing tasks to the BS for calculation, which can reduce the time delay of the computing process as much as possible.

Generally speaking, due to their fixed setup, BSs may encounter challenges in adapting effectively to the fluctuations in users' computing power requirements. This inevitably leads to an unequal distribution of computing resources across the network. For instance, in scenarios where users are sparsely distributed, certain users may fall outside the service range of the BS. Consequently, their tasks cannot be offloaded to the BS, making it difficult to meet the low-latency requirements of user applications. Fortunately, we can leverage UAVs with their strong mobility and flexibility to help balance the distribution of computing resources in scenarios where ground infrastructure is scarce or absent [2–4]. Nevertheless, UAVs typically own fewer computing and communication resources in comparison to BSs, and when a UAV is tasked to provide computing services to multiple users, it may face challenges such as task backlogs and resource insufficiency. It is important to note that UAVs can not only directly offer temporary computing services to users but also act as communication relays to transmit tasks from users to BSs. This dual role allows users to have enhanced computing resources, thereby effectively reducing computing latency [4].

Unfortunately, in networks with few available BSs, utilizing UAVs for task offloading can potentially result in load imbalance, where certain UAVs may bear a disproportionate number of tasks while others remain idle. The load imbalance can significantly decrease system performance and efficiency. To be specific, an overloaded UAV may experience resource shortages, increased latency, and excessive energy consumption, while an idle UAV squanders available resources and fails to ralize its full potential. To address the aforementioned challenges, this paper introduces a UAV collaborative three-layer MEC system model, where the task allocation and optimization problem is reformulated as a mixed integer nonlinear programming (MINLP) problem with the goal of facilitating efficient task offloading and resource allocation among users, UAVs, and BSs through coordinated UAV collaboration. The extended genetic algorithm for UAV collaboration in three-layer MEC task allocation and optimization (EGAC-3MEC) is proposed to tackle the problem, aiming to minimize the total system time delay and energy consumption. The main contributions of this work are summarized below.

1) To alleviate the load imbalance problem of BSs, a novel UAV collaboration scheme is integrated into the MEC system. The problem is structured as a comprehensive optimization challenge with a three-layer task offloading approach, presented as a MINLP problem. By addressing the problem, one could minimize the weighted sum of the total system delay, total energy consumption, and high-priority user delay.

2) A hierarchical decision-making method is proposed which divides the optimization problem into three sub-problems: user task offloading and resource allocation, UAV task offloading, and UAV resource allocation. In order to minimize the overall system cost, a genetic algorithm is proposed for solving the first sub-problem, while the branch-and-bound method along with the interior point method are employed to tackle the remaining sub-problems.

3) An improved genetic algorithm is proposed to solve the user's task offloading sub-problem effectively. By adjusting the selection probability, individuals exhibiting lower total system costs are more likely to be chosen. After crossover and mutation operations, new individuals will replace the worst one in the population, thereby enhancing the overall fitness of the population and gradually reducing the cost of the original problem.

The rest of this paper is organized as follows. In Section 2, related works are reviewed. We present the system model and formulation in Section 3. The extended genetic algorithm is given in Section 4. Section 5 presents the experiment results. Finally, we conclude this paper in Section 6.

2. Related Work

2.1. Partial Offloading vs. Binary Offloading

In MEC, two commonly adopted task offloading modes are the partial offloading mode and binary offloading mode [1]. The partial offloading mode involves transferring part of the user's computing tasks to the edge or cloud servers for processing, while retaining other segments for local execution on the mobile device. On the other hand, the binary offloading mode involves offloading the entire task to the edge server or cloud server. To maximize the task sharing and collaborative processing capabilities, research works such as [5–7] primarily focused on computation offloading in the partial offloading mode. Wang *et al.* formulated the task allocation problem as a constraint satisfaction problem and introduced a lightweight heuristic algorithm to enhance resource utilization on the MEC side [7].

To minimize time delay and energy consumption, the works [8, 9–11] primarily delved into the realm of computation offloading within the binary offloading mode. Peng *et al.* viewed the task allocation problem as an online decision-making process and developed a task allocation technique that supports the real-time user assignment [9]. To address the task offloading problem, Saleem *et al.* not only introduced an evolutionary algorithm based on genetic algorithms that can directly derive solutions but also proposed a low-complexity heuristic algorithm that considers network dynamics [10].

2.2. Collaboration between BSs and UAVs

The collaboration between BSs and UAVs enhances the efficiency and reliability of computing and communication services in MEC. UAVs play a crucial role as communication relays to transfer the users' computing tasks to BSs for processing, and thus can significantly reduce the computation delay. Additionally, in remote areas or emergencies, UAVs can extend the service range of MEC, providing temporary network coverage and immediate services to meet urgent demands. In certain scenarios, the closed-form analytical solutions for BS and UAV collaboration could be obtained [2–4]. For example, Yang *et al*. proposed a hierarchical machine learning task distribution framework for a visual target tracking system [2]. They formulated a weighted sum cost minimization problem for binary offloading and partial offloading schemes and derived a closed-form optimal offloading probability and optimal offloading rate for the binary offloading and partial offloading schemes. Unfortunately, for majority of the scenarios, obtaining analytical solutions is challenging or even infeasible due to the complexity and non-linearity of the collaboration problems involved. Therefore, some non-convex optimization algorithms or numerical algorithms like heuristic algorithms, reinforcement learning, and deep learning are widely employed to optimize the collaboration between BSs and UAVs.

The non-convex optimization algorithms approaches the optimal solution through an iterative optimization process [12–14]. Ei *et al*. investigated a two-level MEC system and multiple UAVs are deployed to provide computing and relay services for mobile devices. The problem is formulated as a joint optimization problem of task offloading, communication, and computing resource allocation, and then addressed based on the block successive upper bound minimization method [12]. Heuristic algorithms utilize experiential knowledge and rules to systematically explore solution spaces, gradually select and update solutions based on specified criteria [15, 16]. To solve the joint optimization problem of computing offloading and resource allocation, Goudarzi *et al*. introduced an optimized computing resource allocation approach using co-evolutionary computing, which guarantees that the task computation delays (for all users) are within a time block [17]. Economic methods, like the game theory, can explore the cooperation and competition dynamics between BS and UAV and then design the corresponding appropriate mechanisms and strategies [16, 18]. Apostolopoulos *et al*. framed the task offloading problem of UAVs or BSs as a user satisfaction maximization problem, and treated the framed problem as a non-cooperative game among users [18]. Reinforcement learning and deep learning techniques can enhance BS-UAV collaboration for intelligent decision-making and optimization [19–23]. Zhou *et al*. studied the joint impact of task priority and mobile computing services over MEC networks. A deep Q-learning algorithm was proposed to learn an effective solution through continuous interaction between the UAV agent and the system environment [23]. Additionally, it is noted that multiple solutions can be combined to achieve collaboration between BSs and UAVs in practical applications [24–26].

2.3. Collaboration among UAVs

The collaboration mechanism among UAVs is crucial to achieving flexible and efficient MEC, as it could optimize network resource utilization and improve service quality and user experience. In MEC, UAV collaboration involves multiple UAVs working together to complete tasks, share resources and information, and enhance overall computing and communication efficiency through collaborative efforts. Collaboration among UAVs requires the establishment of a reliable communication network to facilitate real-time information exchange and coordinated actions. Additionally, appropriate position and trajectory planning are vital to ensure seamless teamwork and collision avoidance among the UAVs.

Up to now, a diverse array of independent methodologies, including reinforcement learning and game theory, have been explored to confront these challenges. The overall objective is to enhance the collaborative performance and operational efficiency of UAV systems [27–33]. Reinforcement learning seeks to acquire optimal behavioral strategies by facilitating interactions between an agent and its environment [28–33]. Within the context of UAV cooperation, each UAV can be viewed as an autonomous agent tasked with learning the most effective cooperation strategy by engaging with its environment, which includes other UAVs, BSs, sensor data, and more. Kong *et al*. introduced a service quality optimization framework for multi-UAV collaboration. They formulated a service quality model and utilized the deep deterministic policy gradient (DDPG) algorithm to elevate service quality performance across diverse sets of subtasks [33]. The game theory serves as a mathematical tool and theoretical framework for examining decision-making processes among decision-makers within interactive environments. In the context of UAV cooperation, UAVs can be seen as decision-makers, and the collaboration conundrum can be tackled by scrutinizing the interactions and strategic choices between them. Gao *et al*. delved into the optimal three-dimensional deployment of UAVs and devised a decentralized learning algorithm to attain the optimal Nash equilibrium of the system [27]. Each UAV is tasked with solely exchanging local information with its neighboring UAVs and conducting local real-time computations to explore its optimal 3D deployment, thereby maximizing the system's channel capacity. The iterative optimization algorithm is often used to solve optimization problems, and gradually approaches the optimal solution by iteratively updating parameters or strategies. He *et al*. proposed a multi-hop task offloading scheme with real-time computing capabilities to achieve a more powerful multi-UAV remote edge computing network [32]. To tackle the associated joint resource allocation and UAV deployment dilemmas, a robust local optimal strategy was designed which employed the gradient descent method after chain rule conversion.

In addition to utilizing individual methods, multiple methods have also been integrated for collaborative deployment in resolving UAV cooperation challenges. These kinds of integration comprehensive combine the advantages of different methods, involving the cross-application of techniques such as reinforcement learning and heuristic algorithms [34–40]. For example, to solve the joint optimization problem of UAV deployment and offloading decisions, Xia *et al*. proposed a two-layer optimization algorithm and solved the problem by using alternating optimization [36]. In these two layers of optimization algorithms, the differential evolution (DE) learning algorithm and the distributed deep neural network (DDNN) were used, respectively, to generate the deployment and offloading decision-making scheme of the UAV. Kaitao Meng *et al*. introduced a novel multi-UAV cooperative sensing and transmission scheme that incorporates overlapping sensing task allocation [39]. In scenarios where sensing tasks overlap, the problem was reformulated as a monotonic optimization challenge and addressed using the generic Polyblock algorithm. Conversely, in situations where sensing tasks do not overlap, a double-loop binary search algorithm was devised to tackle the optimization problem.

Based on the aforementioned work, it can be observed that most of the research in UAV-assisted MEC networks tends to focus on a single type of MEC server deployment, such as either deploying the MEC server on the BS or carrying it on the UAV. However, existing research often overlooks the coordination mechanism between the MEC servers deployed on UAVs and BSs. This lack of coordination limits the overall flexibility and performance of the system. In particular, in complex dynamic environments, how to effectively coordinate MEC servers across heterogeneous deployments for real-time data processing and efficient task offloading remains an open challenge. Moreover, many studies fail to fully consider the load balancing problem of the MEC server carried by UAVs. In practical applications, UAV-mounted MEC servers face significant constraints in terms of computing and storage resources. As multiple UAVs collaborate, how to optimally allocate tasks and balance loads among them to prevent overloading or underutilization is a critical challenge for improving task processing efficiency and overall network coverage. Without addressing this issue, load imbalance can result in a waste of computational resources and degrade network stability and service quality. Therefore, in UAV collaboration and MEC resource management, solving the load balancing problem is vital for enhancing the overall system performance and optimizing resource usage.

This study differentiates itself from previous works in the field by addressing the coordination of MEC servers deployed across both UAVs and BSs in a collaborative three-layer system. Unlike existing approaches that focus solely on a single deployment strategy, this work introduces a more flexible system where UAVs not only provide computational support but also act as relays and manage task offloading for load balancing. In the paper, we consider the task offloading and resource scheduling problems with UAV collaborative MEC systems. In such a scenario, the decision variables typically encompass task allocation, resource allocation, transmission path selection, power control, and more. The intricate relationships among these variables, coupled with numerous constraints that must be satisfied, render problem-solving a challenging endeavor. To address this complexity, we propose the EGAC-3MEC, which decomposes the problem into three sub-problems. This decomposition strategy aims to simplify the overall problem-solving process by breaking down the intricate interdependencies into more manageable components. By dividing the problem into distinct layers where each layer focuses on specific aspects such as task allocation, resource allocation, and transmission path selection, the complexity of the optimization task can be reduced to obtain more efficient and effective solutions.

3. System Model

In this section, we will introduce a UAV collaborative three-layer MEC system model, describe the offloading, computation, and communication models, and propose a problem description.

3.1. UAV Collaborative Three-Layer MEC System Model

As shown in Figure 1, the considered UAV collaborative three-layer MEC system model consists of a BS, UAVs, and users. Users generate computing tasks, which can be processed locally or offloaded to either the BS or UAVs. Each UAV is equipped with a computing server, while the BS is equipped with an MEC server to provide computing service to users. Assume that the number of users and UAVs in the system are U and A, respectively. The user set is $\mathcal{U} = \{1, 2, 3, \cdots, U\}$ and the UAV set is $\mathcal{A} = \{1, 2, 3, \cdots, A\}$.

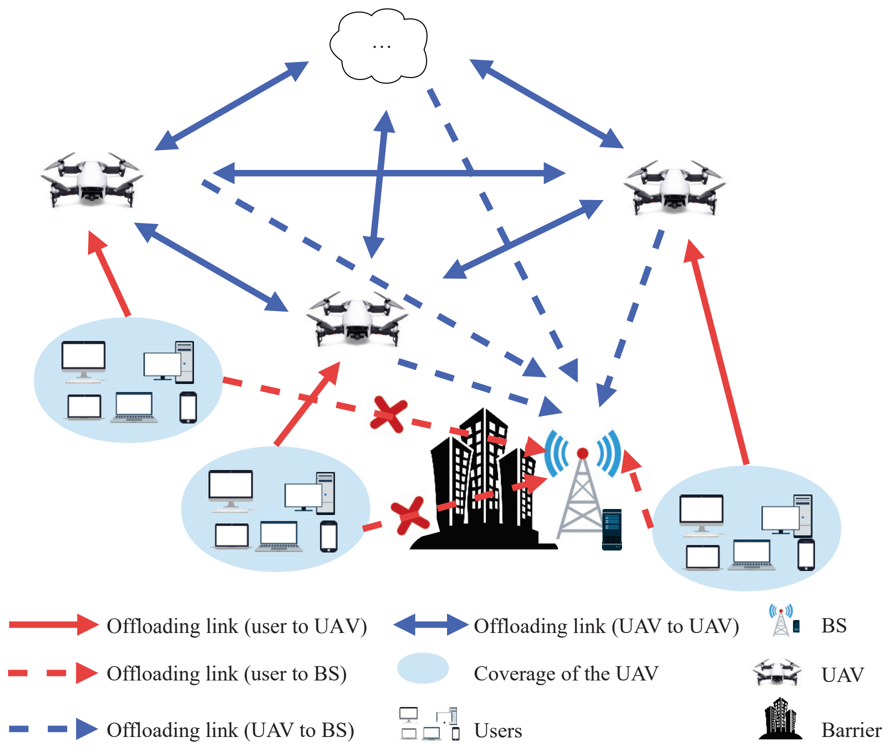
…
Offloading link (user to UAV)
Offloading link (UAV to UAV)
BS
Offloading link (user to BS)
Coverage of the UAV
UAV
Offloading link (UAV to BS)
Users
Barrier

Figure 1. Task offloading paths and communication links in UAV-assisted three-layer MEC model with obstacle consideration.

Note that the tasks generated by users exhibit variations in terms of data sizes, required CPU cycles, maximum tolerated time delays, and priority levels. Therefore, we describe the tasks generated by user i as $\mathcal{M}_i = (S_i, C_i, T_i, d_i)$, where S_i represents the task size, C_i represents the number of CPU cycles required to compute a one-bit task, T_i represents the maximum tolerated time delay of this user task and d_i represents the task priority.

3.2. Offloading Model

The binary offloading method is employed to minimize task processing delays, whereby the user's computing tasks are considered as indivisible entities. Specifically, each task must be processed in its entirety at one of three locations: either locally on the user's device, at the UAV, or at the BS. Tasks cannot be divided or split between these locations and must be fully offloaded to one of these three options.

We consider that the user can offload its computing tasks to UAVs when it is located within the service range of the UAV. Therefore, the users could offload the local computing tasks to BS either directly or indirectly through a UAV. More specifically, if the user is located outside the coverage area of the BS or if there are obstacles obstructing signal transmission between the user and the BS, direct access to the BS is impossible. In such cases, users can offload their tasks to a UAV first if they would like to have their tasks executed on the BS. The UAV now plays the role as a communication relay, facilitating the offloading of users' tasks to the BS, thus completing the indirect offloading process.

As such, a binary variable $a_{ij} \in \{0,1\}$ could be utilized to represent the above process as shown below.

$$a_{ij} = \begin{cases} 1, & \text{user } i \text{ offloads the task to} \\ & \text{BS } (j=0) \text{ or UAV } (j \in \mathcal{A}) \\ 0, & \text{user } i \text{ does not offload the task to} \\ & \text{BS } (j=0) \text{ or UAV } (j \in \mathcal{A}) \end{cases}$$

Since the task of the user cannot be split, it is either processed locally or offloaded to the BS or a UAV. The offloading constraint can be expressed as

$$\sum\nolimits_{j \in \mathcal{A} \cup \{0\}} a_{ij} \le 1, \ \ \forall i \in \mathcal{U},$$

where $j \in \mathcal{A} \cup \{0\}$ and

$$a_{ij} \in \{0,1\}, \ \forall i \in \mathcal{U}, \ j \in \mathcal{A} \cup \{0\}.$$

When a UAV receives the computing task offloaded from a user, the UAV can choose to either execute the task locally or offload to the BS. A binary variable $c_i^{j0} \in \{0,1\}$ is defined to indicate whether or not the computing task (transmitted by user i to UAV j) is further offloaded to the BS as follows:

$$c_i^{j0} = \begin{cases} 1, \text{ UAV } j \text{ offloads the task from user } i \\ \quad \text{directly or through UAV to BS} \\ 0, \text{ UAV } j \text{ does not offload the received} \\ \quad \text{task from user } i \text{ to BS} \end{cases}$$

Noting that only when a user offloads the task to a UAV, the UAV can offload the task received to the BS. The following relation holds:

$$c_i^{j0} \in \{0, a_{ij}\}, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A}.$$

In addition to offloading tasks to the BS, UAVs can also offload computing tasks from users to other UAVs. A binary variable $b_i^{jj'} \in \{0,1\}$ is used to indicate whether or not the computing task transmitted by user i to UAV j is further offloaded to UAV j' as shown below.

$$b_i^{jj'} = \begin{cases} 1, \text{UAV } j \text{ offloads the task from user } i \\ \quad \text{to UAV } j' \\ 0, \text{UAV } j \text{ does not offload the received} \\ \quad \text{task to UAV } j' \end{cases}$$

Noting that only when a user offloads the task to a UAV, the UAV can offload the task received to another UAV, we have

$$b_i^{jj'} \in \{0, a_{ij}\}, \ \ \forall i \in \mathcal{U}, j, j' \in \mathcal{A}, j \neq j'.$$

Moreover, if the UAV decides to offload the local task to another UAV, only a single UAV can be selected, i.e.,

$$\sum\nolimits_{j' \in \mathcal{A} \backslash j} b_i^{jj'} \leq 1, \forall i \in \mathcal{U}, j \in \mathcal{A}.$$

According to the above setup, only when user i offloads the computing task to UAV j, can $b_i^{jj'}$ and c_i^{j0} potentially have value 1, otherwise 0, i.e.,

$$(1 - a_{ij}) + c_i^{j0} + \sum\nolimits_{j' \in \mathcal{A} \backslash j} b_i^{jj'} \leq 1, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A}.$$

Taking into account the UAV capacity limitation, which restricts the number of offloading connections it can establish with users, the capacity constraint is formulated as follows:

$$\sum\nolimits_{i \in \mathcal{U}} (a_{ij} - c_i^{j0} + \sum\nolimits_{j' \in \mathcal{A} \backslash j} (b_i^{j'j} - b_i^{jj'})) \leq C_j, \ \ \forall j \in \mathcal{A},$$

where C_j represents the maximum number of offloading connections that UAV j can establish with users.

To establish an offloading relationship, the constraints of the service scope must be satisfied. For instance, for a user to offload a task to a UAV, the user must fall within the UAV's service scope. Similarly, this constraint must also be adhered to when offloading a user task to the BS or offloading a UAV task to another UAV or BS. The service scope constraints can be described as follows:

$$a_{ij} d_{ij} \leq R_j, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A} \cup \{0\},$$

$$c_i^{j0} d_{j0} \leq R_0, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A},$$

$$b_i^{jj'} d_{jj'} \leq R_{j'}, \ \ \forall i \in \mathcal{U}, j, j' \in \mathcal{A}, j \neq j',$$

where d_{ij} is the Euclidean distance between user i and UAV j, d_{j0} is the Euclidean distance between UAV j and BS, and $d_{jj'}$ is the Euclidean distance between UAV j and UAV j'. R_j, $R_{j'}$ and R_0 are the service radius of UAV j, UAV j' and BS, respectively.

3.3. Computation Model

a) Computing at Local User: The time spent on the local computation of ith user is given by

$$t_i^{loc} = C_i S_i (1 - \sum_{j=0}^{A} a_{ij}) / f_i, \ \forall i \in \mathcal{U},$$

where C_i is the CPU cycles required to calculate one bit of the user task, S_i is the user task size, and f_i is the user's local CPU frequency.

According to [34], the energy consumed by user i in the local computation is given by

$$E_i^{loc} = k f_i^2 C_i S_i (1 - \sum_{j=0}^{A} a_{ij}), \ \forall i \in \mathcal{U},$$

where k is a constant depending on the chip architecture of the user device.

b) Computing at UAVs: The local tasks that UAV j needs to handle, denoted as M_j^{all}, are composed of the tasks offloaded by the users M_j^1, the tasks offloaded to other UAVs M_j^2, the tasks offloaded to the BS M_j^3, and the tasks offloaded by other UAVs M_j^4

$$M_j^{all} = M_j^1 - M_j^2 - M_j^3 + M_j^4, \ \forall j \in \mathcal{A},$$

with

$$\begin{aligned} M_j^1 &= \sum_{i \in \mathcal{U}} a_{ij} S_i, \\ M_j^2 &= \sum_{j' \in \mathcal{A} \setminus j} \sum_{i \in \mathcal{U}} a_{ij} b_i^{jj'} S_i, \\ M_j^3 &= \sum_{i \in \mathcal{U}} a_{ij} c_i^{j0} S_i, \\ M_j^4 &= \sum_{j' \in \mathcal{A} \setminus j} \sum_{i \in \mathcal{U}} a_{ij'} b_i^{j'j} S_i. \end{aligned}$$

Define M_{ij}^{all} as the tasks of user i executed at UAV j. It is obvious that

$$M_{ij}^{all} = a_{ij} S_i (1 - c_i^{j0} - \sum_{j' \in \mathcal{A} \setminus j} b_i^{jj'}) + \sum_{j' \in \mathcal{A} \setminus j} a_{ij'} b_i^{j'j} S_i, \ \forall i \in \mathcal{U}, \ j \in \mathcal{A}.$$

and therefore

$$M_j^{all} = \sum_{i \in \mathcal{U}} M_{ij}^{all}, \ \forall j \in \mathcal{A}.$$

The computation delay at UAV j is

$$t_j^{UAV,comp} = \sum_{i \in \mathcal{U}} t_{ij}^{UAV,comp}, \ \forall j \in \mathcal{A}$$

where $t_{ij}^{UAV,comp}$ is the computation delay of user i at UAV j defined by

$$t_{ij}^{UAV,comp} = M_{ij}^{all} C_i / f_{ij}, \ \forall i \in \mathcal{U}, \ j \in \mathcal{A}.$$

The computation energy consumption at UAV j is

$$E_j^{UAV,comp} = \sum_{i \in \mathcal{U}} E_{ij}^{UAV,comp}, \ \forall j \in \mathcal{A},$$

where $E_{ij}^{UAV,comp}$ is the computing energy consumption of user i at UAV j defined by

$$E_{ij}^{UAV,comp} = k f_{ij}^2 C_i S_i \left[a_{ij} (1 - c_i^{j0} - \sum_{j' \in \mathcal{A} \setminus j} b_i^{jj'}) + \sum_{j' \in \mathcal{A} \setminus j} a_{ij'} b_i^{j'j} \right], \ \forall i \in \mathcal{U}, j \in \mathcal{A},$$

where f_{ij} is the computing power allocated by UAV j to user i.

c) Computing at BS: Due to the powerful computing resources of BS, we assume that the computation delay and energy consumption at the BS can be ignored.

3.4. Communication Model

We use frequency division multiple access (FDMA) as the access method for MEC devices. FDMA divides the system bandwidth into multiple sub-channels. Each sub-channel occupies a different frequency range in the frequency domain. The sub-channels will not interfere with each other so multiple users can access them simultaneously.

a) Transmission from User to UAV: According to [35], the communication link between the user and UAV is assumed to be controlled by a line-of-sight (LoS) channel, so the channel gain between user i and UAV j can be expressed as

$$h_{ij} = \eta_0 / d_{ij}^2,$$

where η_0 represents the channel gain at a relative distance of 1 meter. The transmission rate from user i to UAV j is

$$R_{ij} = B_{ij}\log_2(1+P_i h_{ij}/\sigma^2),$$

where σ^2 is the communication channel noise power, B_{ij} is the communication bandwidth allocated by UAV j to user i, P_i is the transmission power of user i. In this scenario, the transmission delay from user i to UAV j is

$$t_{ij}^{up} = a_{ij}S_i/R_{ij},$$

and the transmission energy consumption from user i to UAV j is

$$E_{ij}^{up} = P_i t_{ij}^{up}.$$

b) Transmission from User to BS: The channel gain between user i and BS can be expressed as

$$h_{i0} = \eta_0/d_{i0}^2,$$

where d_{i0} is the Euclidean distance between user i and BS. The transmission rate from user i to BS is

$$R_{i0} = B_{i0}\log_2(1+P_i h_{i0}/\sigma^2),$$

where B_{i0} is the communication bandwidth allocated by BS to user i. The transmission delay from user i to BS is

$$t_{i0} = a_{i0}S_i/R_{i0},$$

and the transmission energy consumption from user i to BS is

$$E_{i0} = P_i a_{i0} S_i/R_{i0}.$$

c) Transmission from UAV to another UAV: The channel gain between UAV j and UAV j' can be expressed as

$$h_{jj'} = \eta_0/d_{jj'}^2,$$

where $d_{jj'}$ is the Euclidean distance between UAV j and UAV j'. The transmission rate from UAV j to UAV j' is

$$R_{jj'} = B_{jj'}\log_2(1+P_{jj'}h_{jj'}/\sigma^2),$$

where $B_{jj'}$ is the communication bandwidth allocated by UAV j' to UAV j and $P_{jj'}$ is the transmission power of UAV j. The transmission delay from UAV j to UAV j' is

$$t_i^{jj',trans} = a_{ij}b_i^{jj'}S_i/R_{jj'},$$

and the transmission energy consumption from UAV j to UAV j' is

$$E_i^{jj',trans} = \frac{P_{jj'}a_{ij}b_i^{jj'}S_i}{R_{jj'}}.$$

d) Transmission from UAV to BS: The channel gain between UAV j and BS can be expressed as

$$h_{j0} = \eta_0/d_{j0}^2,$$

where d_{j0} is the Euclidean distance between UAV j and BS. The transmission rate from UAV j to BS is

$$R_{j0} = B_{j0}\log_2(1+P_{j0}h_{j0}/\sigma^2),$$

where B_{j0} is the communication bandwidth allocated by BS to UAV j, P_{j0} is the transmission power of UAV j. The transmission delay from UAV j to BS is

$$t_i^{j0,trans} = a_{ij}c_i^{j0}S_i/R_{j0},$$

and the transmission energy consumption from UAV j to BS is

$$E_i^{j0,trans} = P_{j0}a_{ij}c_i^{j0}S_i/R_{j0}.$$

3.5. Problem Formulation

Given the aforementioned offloading and computation models, we consider the UAV collaborative three-layer MEC system framework by optimizing a comprehensive performance in terms of system delay, energy consumption, and user priority.

The user-UAV-BS task offloading and resource allocation research problem can be formulated as follows:

$$\min_{\{\alpha,\beta,\gamma,P,B,F\}} \Psi = \mu T_1 + (1-\mu)T_2 \tag{1}$$

$$\begin{aligned}
&s.t.\\
&C_1 : \textstyle\sum_{j=0}^{A} a_{ij} \leqslant 1, \ \ \forall i \in \mathcal{U},\\
&C_2 : a_{ij} \in \{0,1\}, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A} \cup \{0\},\\
&C_3 : \textstyle\sum_{j' \in \mathcal{A} \setminus j} b_i^{jj'} \leqslant 1, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A},\\
&C_4 : b_i^{jj'} \in \{0, a_{ij}\}, \ \ \forall i \in \mathcal{U}, j, j' \in \mathcal{A}, j \neq j',\\
&C_5 : (1-a_{ij}) + c_i^{j0} + \textstyle\sum_{j' \in \mathcal{A} \setminus j} b_i^{jj'} \leqslant 1, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A},\\
&C_6 : c_i^{j0} \in \{0, a_{ij}\}, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A},\\
&C_7 : 0 \leqslant B_{ij} \leqslant B_j^{\max}, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A} \cup \{0\},\\
&C_8 : \textstyle\sum_{i \in \mathcal{U}} a_{ij} B_{ij} \leqslant B_j^{\max}, \ \ \forall j \in \mathcal{A} \cup \{0\},\\
&C_9 : 0 \leqslant f_{ij} \leqslant F_j^{\max}, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A},\\
&C_{10} : \textstyle\sum_{i \in \mathcal{U}} a_{ij} f_{ij} \leqslant F_j^{\max}, \ \ \forall j \in \mathcal{A},\\
&C_{11} : 0 \leqslant P_{jj'} \leqslant P_j^{\max}, \ \ \forall j, j' \in \mathcal{A}, j \neq j',\\
&C_{12} : a_{ij}(\textstyle\sum_{j' \in \mathcal{A} \setminus j} b_i^{jj'} P_{jj'} + c_i^{j0} P_{j0}) \leqslant P_j^{\max}, \quad \forall i \in \mathcal{U}, j \in \mathcal{A},\\
&C_{13} : L_{total}^{i} \leqslant T_i, \ \ \forall i \in \mathcal{U},\\
&C_{14} : \textstyle\sum_{i \in \mathcal{U}} (a_{ij} - c_i^{j0} + \sum_{j' \in \mathcal{A} \setminus j} (b_i^{j'j} - b_i^{jj'})) \leqslant C_j, \quad \forall j \in \mathcal{A},\\
&C_{15} : a_{ij} d_{ij} \leqslant R_j, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A} \cup \{0\},\\
&C_{16} : c_i^{j0} d_{j0} \leqslant R_0, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A},\\
&C_{17} : b_i^{jj'} d_{jj'} \leqslant R_{j'}, \ \ \forall i \in \mathcal{U}, j, j' \in \mathcal{A}, j \neq j'.
\end{aligned}$$

The decision variables to be optimized consist of task offloading variables and resource allocation variables. Task offloading variables consisit of user task offloading variable $\alpha = \{a_{ij} | i \in \mathcal{U}, j \in \mathcal{A} \cup \{0\}\}$, UAV-UAV offloading variable $\beta = \{b_i^{jj'} | i \in \mathcal{U}, j, j' \in \mathcal{A}, j \neq j'\}$, and UAV-BS offloading variable $\gamma = \{c_i^{j0} | i \in \mathcal{U}, j \in \mathcal{A}\}$. These offloading variables are all zero-one variables. The resource allocation variables consist of the bandwidth resource $B = \{B_{ij} | i \in \mathcal{U}, j \in \mathcal{A} \cup \{0\}\}$ (allocated by UAVs and BS to users), the computing resource $F = \{f_{ij} | i \in \mathcal{U}, j \in \mathcal{A}\}$ (allocated by UAVs to users), and the offloaded transmission power $P = \{P_{jj'} | j, j' \in \mathcal{A}, j \neq j'\}$ of UAVs.

In the optimization problem outlined above, the objective function is a combination of the overall system delay and energy consumption cost, denoted as T_1, and the total delay experienced by high-priority users is denoted as T_2. T_1 is the weighted sum of the total system delay L_{total} and the total energy consumption E_{total}. In order to minimize the delay for high-priority users and enhance service quality, T_2 is set to the total delay specifically for high-priority users. The cost of the objective function is the weighted cost sum of T_1 and T_2, expressed as follows:

$$\begin{aligned}
T_1 &= \lambda L_{total} + (1-\lambda) E_{total},\\
T_2 &= \textstyle\sum_{i \in \mathcal{U}} I(d_i > d) L_{total}^{i},
\end{aligned}$$

where the total system delay includes both computation delay and transmission delay

$$L_{total} = L_{total}^{comp} + L_{total}^{trans},$$

with the computation delay

$$L_{total}^{comp} = \sum_{i \in \mathcal{U}} t_i^{loc} + \sum_{j \in \mathcal{A}} t_j^{UAV,comp},$$

and the transmission delay

$$L_{total}^{trans} = \sum_{i \in \mathcal{U}} \sum_{j \in \mathcal{A}} t_{ij}^{up} + \sum_{i \in \mathcal{U}} \sum_{j \in \mathcal{A}} t_i^{j0,trans} + \sum_{i \in \mathcal{U}} t_{i0} + \sum_{i \in \mathcal{U}} \sum_{j' \in \mathcal{A} \setminus j} \sum_{j \in \mathcal{A}} t_i^{jj',trans}.$$

Define the computation delay of user i as

$$L_{total}^{i} = t_i^{loc} + t_{i0} + \sum_{j \in \mathcal{A} \setminus j'} \sum_{j' \in \mathcal{A}} t_i^{jj',trans} + \sum_{j \in \mathcal{A}} (t_{ij}^{up} + t_{ij}^{UAV,comp} + t_i^{j0,trans}),$$

and thus we have

$$L_{total} = \sum\nolimits_{i\in\mathcal{U}} L_{total}^{i}.$$

The total energy consumption consists of computation energy consumption and transmission energy consumption as follows

$$E_{total} = E_{total}^{comp} + E_{total}^{trans},$$

with the computation energy consumption

$$E_{total}^{comp} = \sum\nolimits_{i\in\mathcal{U}} E_{i}^{loc} + \sum\nolimits_{j\in\mathcal{A}} E_{j}^{UAV,comp},$$

and the transmission energy consumption

$$E_{total}^{trans} = \sum\nolimits_{i\in\mathcal{U}}\sum\nolimits_{j\in\mathcal{A}} E_{ij}^{up} + \sum\nolimits_{i\in\mathcal{U}}\sum\nolimits_{j\in\mathcal{A}} E_{i}^{j0,trans} + \sum\nolimits_{i\in\mathcal{U}} E_{i0} + \sum\nolimits_{i\in\mathcal{U}}\sum\nolimits_{j'\in\mathcal{A}\setminus j}\sum\nolimits_{j\in\mathcal{A}} E_{i}^{jj',trans}.$$

The constraints C_1 and C_2 indicate that a user task cannot be divided and must be executed either locally or offloaded solely to the BS or a single UAV. C_3 and C_4 govern the cooperative offloading variables for UAVs, stipulating that a UAV can only pass on a task to another UAV if the task was initially offloaded to it by a user. C_5 and C_6 represent that when a user offloads a task to a UAV, the UAV can further offload the received task to either the BS or another UAV. C_7 and C_{12} define the boundaries for bandwidth resources, computing resources, and UAV transmission power. C_{13} sets the maximum delay threshold that a user task can tolerate. C_{14} imposes a cap on the number of UAVs that can engage in offloading relationships with users. C_{15} and C_{16} establish coverage limitations for UAVs and the BS. A user can only establish an offloading connection with a UAV or BS whose service range covers the user. Similarly, a UAV can only establish an offloading relationship with another UAV or BS within its service range. C_{17} governs the coverage relationship among UAVs, specifying that a UAV can only establish an offloading connection within the coverage range of another UAV.

The objective function (1) involves a diverse set of decision variables, including continuous ones like resource allocation variables and binary ones like offloading variables. These variables are constrained by a mix of linear and nonlinear constraints, and the value of one variable may affect the optimal value of other variables, which further increases the complexity of the optimization task. To tackle the intricate multi-variable mixed constraint optimization challenge effectively, we employ a decomposition strategy. This approach decomposes the complex original optimization problem into several simpler and more tractable sub-problems. This deliberate decomposition substantially diminishes the overall complexity, allowing for the application of tailored techniques and algorithms to tackle each sub-problem effectively. By solving these sub-problems in sequence, one thus can gradually approach the global optimal solution.

4. Extended Genetic Algorithm for UAV Collaboration in Three-Layer MEC Task Allocation and Optimization

In this section, we will propose an extended genetic algorithm, i.e., EGAC-3MEC, to solve the user-UAV-BS task offloading and resource allocation problem. Before proceeding, we decompose the original optimization problem into three sub-problems: user task offloading and related resource allocation, UAV tasks offloading, and UAV resources allocation.

4.1. User Task Offloading and Related Resource Allocation

The first sub-problem is to determine the user-related decision variables, including the task offloading variable α and the related resource allocation variables B and F. As such, we treat the remaining decision variables in the original optimization problem as constants. This allows us to concentrate solely on optimizing these three user-related decision variables. Now, the first sub-problem can be formulated as

$$\min_{\{\alpha,B,F\}} \Psi(\alpha,B,F) \tag{2}$$

$$\begin{aligned}
&s.t.\\
&C_1 : \textstyle\sum_{j=0}^{A} a_{ij} \leqslant 1,\ \ \forall i\in\mathcal{U},\\
&C_2 : a_{ij}\in\{0,1\},\ \ \forall i\in\mathcal{U}, j\in\mathcal{A}\cup\{0\},\\
&C_5 : (1-a_{ij}) + c_i^{j0} + \textstyle\sum_{j'\in\mathcal{A}\setminus j} b_i^{jj'} \leqslant 1,\ \ \forall i\in\mathcal{U}, j\in\mathcal{A},
\end{aligned}$$

$$\begin{aligned}
&C_7: 0 \leqslant B_{ij} \leqslant B_j^{\max}, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A} \cup \{0\},\\
&C_8: \textstyle\sum_{i \in \mathcal{U}} a_{ij} B_{ij} \leqslant B_j^{\max}, \ \ \forall j \in \mathcal{A} \cup \{0\},\\
&C_9: 0 \leqslant f_{ij} \leqslant F_j^{\max}, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A},\\
&C_{10}: \textstyle\sum_{i \in \mathcal{U}} a_{ij} f_{ij} \leqslant F_j^{\max}, \ \ \forall j \in \mathcal{A},\\
&C_{12}: a_{ij}(\textstyle\sum_{j' \in \mathcal{A} \setminus j} b_i^{jj'} P_{jj'} + c_i^{j0} P_{j0}) \leqslant P_j^{\max}, \quad \forall i \in \mathcal{U}, j \in \mathcal{A},\\
&C_{13}: L_{total}^i \leqslant T_i, \ \ \forall i \in \mathcal{U},\\
&C_{14}: \textstyle\sum_{i \in \mathcal{U}} (a_{ij} - c_i^{j0} + \sum_{j' \in \mathcal{A} \setminus j} (b_i^{j'j} - b_i^{jj'})) \leqslant C_j, \quad \forall j \in \mathcal{A},\\
&C_{15}: a_{ij} d_{ij} \leqslant R_j, \ \ \forall i \in \mathcal{U}, j \in \mathcal{A} \cup \{0\},
\end{aligned}$$

where $\Psi(\alpha, B, F)$ is the cost function in the original optimization problem (1) with decision variables α, B, F. Considering that the user-related optimization sub-problem (2) involve multiple decision variables, genetic algorithms offer a potent means to traverse extensive solution spaces efficiently. To address this, we introduce an enhanced genetic algorithm as shown below.

a) Population Individuals: The individuals in the population are represented as follows

$$individual = [\overrightarrow{\alpha_1}, \overrightarrow{\alpha_2}, ..., \overrightarrow{\alpha_U};\ \overrightarrow{B_1}, \overrightarrow{B_2}, ..., \overrightarrow{B_U};\ \overrightarrow{F_1}, \overrightarrow{F_2}, ..., \overrightarrow{F_U}],$$

where

$$\overrightarrow{\alpha_i} = [a_{i0}, a_{i1}, a_{i2}, ..., a_{ij}, ..., a_{iA}],$$

$$\overrightarrow{B_i} = [B_{i0}, B_{i1}, B_{i2}, ..., B_{ij}, ..., B_{iA}],$$

$$\overrightarrow{F_i} = [f_{i1}, f_{i2}, ..., f_{ij}, ..., f_{iA}].$$

b) Population Initialization and Fitness Evaluation: During the initialization process, each individual must satisfy the constraints of the original problem such as delay constraints and computing power constraints to ensure the legitimacy of all individuals in the population. When evaluating fitness, the fitness function uses the weighted total cost objective function of the original problem, so the smaller the individual's fitness value, the better, as shown below.

$$fitness(individual) = \Psi(individual).$$

c) Individual Selection: The traditional roulette selection algorithm is improved in the selection process. By adjusting the selection probability, individuals with smaller fitness values can be selected with a greater probability, to introduce more excellent individuals and promote the development of population evolution. The adjusted individual fitness is outlined as follows:

$$\begin{aligned} fitnessInverse(individual) &= \min(FITNESS)\\ &+ n \times [\max(FITNESS) - fitness(individual)], \end{aligned}$$

The individual selection probability is

$$selectionProbability(individual) = \frac{fitnessInverse(individual)}{\sum fitnessInverse},$$

where $fitnessInverse$ is the value after individual fitness conversion, n is the amplification factor, $FITNESS$ is the fitness vector composed of the fitness of all individuals in the population and $selectionProbability$ is the probability of an individual being selected.

d) Individual Crossover: The crossover operation employs a two-point crossover method. Following the generation of offspring, a validation process is conducted to ensure that the offspring complies with all constraints of the original problem. If the offspring successfully meets all constraints, it will replace the individual with the worst fitness in the population. In cases where the offspring fails to meet the constraints, the crossover process is deemed unsuccessful, and no replacement occurs. This systematic approach aims to enhance the overall fitness of the population and progressively reduce the cost associated with the original problem, as shown in Figure 2.

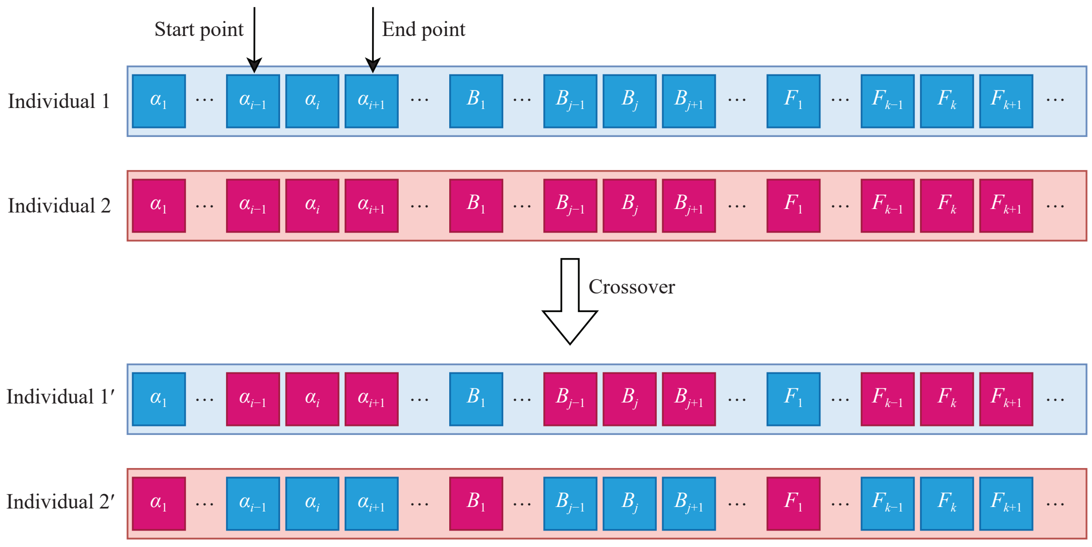
Start point
End point
Individual 1
Individual 2
Crossover
Individual 1′
Individual 2′
α_1 ⋯ α_{i-1} α_i α_{i+1} ⋯ B_1 ⋯ B_{j-1} B_j B_{j+1} ⋯ F_1 ⋯ F_{k-1} F_k F_{k+1} ⋯

Figure 2. Schematic diagram of the individual crossover operation in genetic algorithm, in which two initial individuals exchange gene fragments between specified starting and ending points to generate new individuals.

e) Individual Mutation: The mutation operation occurs for each individual in the population, and each individual has a certain probability of mutating at one of its sites. The mutation operation must be performed within a reasonable range and must meet the constraints of the original problem. The user's offloading variable α is inverted, the communication resource B allocated by the UAV and BS to the user, and the computing resource F allocated by the UAV to the user is taken to a random value within its value range that is different from before mutation. To improve the overall fitness of the population and gradually reduce the cost of the original problem, mutant individuals will replace the individuals with the worst fitness in the population, as shown in Figure 3.

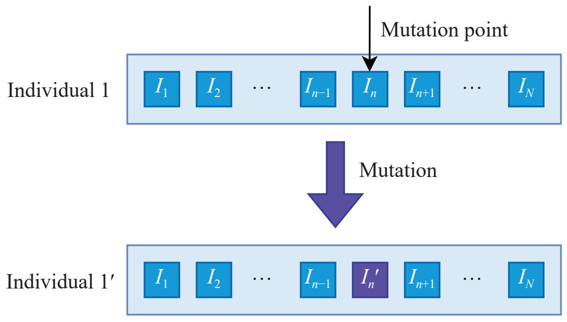
Mutation point
Individual 1
Mutation
Individual 1′
I_1 I_2 ⋯ I_{n-1} I_n I_{n+1} ⋯ I_N
I_1 I_2 ⋯ I_{n-1} I_n' I_{n+1} ⋯ I_N

Figure 3. Schematic diagram of the process in which individual 1 undergoes genetic mutation at the mutation point to generate new individual 1' in the genetic algorithm.

By incorporating these tailored adjustments, the genetic algorithm can better meet the constraints of the original problem. It enhances the population's fitness and systematically refines the solution throughout key stages such as population initialization, fitness evaluation, selection, crossover, and mutation.

Compared to traditional genetic algorithms, the enhanced genetic algorithm significantly improves global search capabilities and adaptability. By employing smarter initialization, selection, crossover, and mutation operations, it can effectively avoid getting stuck in the local optima and adaptively adjust the search process based on the problem's characteristics. In contrast, traditional genetic algorithms are more likely to encounter local optima in the early stages. However, the drawback of the enhanced genetic algorithm is its slower convergence, especially when the solution space is large, as it may require more iterations to find the optimal solution.

4.2. UAV Task Offloading

The second sub-problem is to determine the UAV task offloading decision variables β and γ. The user-related decision variables α, B and F are solved through the first sub-problem and kept as constants. In this part, we only focus on the optimization of the UAV task offloading decision variables as follows:

$$\min_{\{\beta,\gamma\}} \Psi(\beta,\gamma) \tag{3}$$

$$
\begin{aligned}
&s.t.\\
&C_3 : \textstyle\sum_{j'\in\mathcal{A}\setminus j} b_i^{jj'} \leqslant 1, \forall i\in\mathcal{U}, j\in\mathcal{A},\\
&C_4 : b_i^{jj'} \in \{0, a_{ij}\},\ \ \forall i\in\mathcal{U}, j, j'\in\mathcal{A}, j\neq j',\\
&C_5 : (1-a_{ij}) + c_i^{j0} + \textstyle\sum_{j'\in\mathcal{A}\setminus j} b_i^{jj'} \leqslant 1,\ \ \forall i\in\mathcal{U}, j\in\mathcal{A},\\
&C_6 : c_i^{j0} \in \{0, a_{ij}\},\ \ \forall i\in\mathcal{U}, j\in\mathcal{A},\\
&C_{12} : a_{ij}(\textstyle\sum_{j'\in\mathcal{A}\setminus j} b_i^{jj'} P_{jj'} + c_i^{j0} P_{j0}) \leqslant P_j^{\max},\quad \forall i\in\mathcal{U}, j\in\mathcal{A},\\
&C_{13} : L_{total}^i \leqslant T_i,\ \ \forall i\in\mathcal{U},\\
&C_{14} : \textstyle\sum_{i\in\mathcal{U}} (a_{ij} - c_i^{j0} + \sum_{j'\in\mathcal{A}\setminus j} (b_i^{j'j} - b_i^{jj'})) \leqslant C_j,\quad \forall j\in\mathcal{A},\\
&C_{16} : c_i^{j0} d_{j0} \leqslant R_0,\ \ \forall i\in\mathcal{U}, j\in\mathcal{A},\\
&C_{17} : b_i^{jj'} d_{jj'} \leqslant R_{j'},\ \ \forall i\in\mathcal{U}, j, j'\in\mathcal{A}, j\neq j',
\end{aligned}
$$

where $\Psi(\beta,\gamma)$ represents the cost function in the original optimization problem (1) with decision variables β and γ. The second sub-problem is an integer linear programming problem characterized by integer decision variables with intricate constraints, and hence the solution space becomes notably vast and intricate. To adeptly address the integer constraints and navigate the complex search space inherent in this sub-problem, the branch-and-bound method emerges as an effective approach for resolution.

The branch-and-bound method is an exact algorithm that guarantees the global optimal solution, especially for integer linear programming problems. It features clear branching strategies and constraint optimization rules, is highly systematic, and is easy to understand and implement. Traditional integer programming methods are relatively simple, but they become less efficient when solving large-scale or complex problems and often lack accuracy. A disadvantage of the branch-and-bound method is its high computational complexity. As the size of the problem increases, the computational time grows sharply, especially when the solution space is large or when there are many constraints.

4.3. UAV Resource Allocation

In the third sub-problem, the objective is to determine the UAV resource allocation variable P. Following the resolution of the initial two sub-problems, we have deduced the remaining decision variables excluding P. For this segment, we focus on optimizing the UAV resource allocation variable. Consequently, the third sub-problem can be outlined as follows:

$$\min_{\{P\}} \Psi(P) \tag{4}$$

$$
\begin{aligned}
&s.t.\\
&C_{11} : 0 \leqslant P_{jj'} \leqslant P_j^{\max},\ \ \forall j, j'\in\mathcal{A}, j\neq j',\\
&C_{12} : a_{ij}(\textstyle\sum_{j'\in\mathcal{A}\setminus j} b_i^{jj'} P_{jj'} + c_i^{j0} P_{j0}) \leqslant P_j^{\max},\quad \forall i\in\mathcal{U}, j\in\mathcal{A},\\
&C_{13} : L_{total}^i \leqslant T_i,\ \ \forall i\in\mathcal{U},
\end{aligned}
$$

where $\Psi(P)$ represents the cost function in the original optimization problem (1) with the decision variable P. Given that the third sub-problem is a nonlinear programming problem characterized by smooth objective and constraint functions, it can be classified as a convex optimization problem. In this context, standard convex optimization algorithms like the interior point method (IPM) can be effectively employed to derive the optimal UAV power resource allocation [41]. The convex nature of the problem facilitates the application of such algorithms, ensuring efficient and accurate determination of the UAV power allocation for optimal performance.

The interior point method is more efficient in solving convex optimization problems and can quickly find the optimal solution, especially when dealing with large-scale problems, which makes it superior to the traditional gradient descent method. The interior point method converges faster, and with multiple iterations, it can better adjust constraints and improve solution efficiency.

According to the description and solutions of the aforementioned three sub-problems, we integrate them to form a comprehensive Algorithm 1.

Algorithm 1: Extended Genetic Algorithm for UAV Collaboration in Three-Layer MEC Task Allocation and Optimization (EGAC-3MEC)

```
Input: Variables besides the decision variables, Number of iterations $epochs$, Tolerance gap $\epsilon$
Output: The optimal decision variables include $\alpha^*$, $B^*$, $F^*$, $\beta^*$, $\gamma^*$ and $P^*$
1 Randomly initialize population $population$, where each individual consists of $\alpha$, $B$, $F$;
2 Initialize $\beta$, $\gamma$ to all zeros;
3 Initialize $P$ (average initialization);
4 Initialize iteration epoch $epoch = 1$;
5 repeat
6     Evaluate: Evaluate the fitness of each individual in $population$;
7     Perform the Improved Genetic Algorithm to solve problem (2) and find $\alpha^*$, $B^*$, $F^*$;
8         Selection: Select parents for crossover from $population$ using roulette wheel selection;
9         Crossover: Perform crossover to create offspring;
10        Mutation: Apply mutation to offspring;
11        Evaluate: Evaluate the fitness of new individuals;
12        Update: Replace old population with new population based on fitness;
13     Use the branch-and-bound method to solve problem (3) and find $\beta^*$, $\gamma^*$;
14     Use the interior point method to solve problem (4) and find $P^*$;
15     $\alpha = \alpha^*$, $B = B^*$, $F = F^*$, $\beta = \beta^*$, $\gamma = \gamma^*$, $P = P^*$;
16     $\alpha^*$, $B^*$, and $F^*$ form a new individual and replace the individual with the worst fitness in the population;
17     $epoch \leftarrow epoch + 1$;
18 until $epoch = epochs$ or change rate of total cost $\leqslant \epsilon$;
```

5. Performance Evaluation

5.1. Experiment Settings

In the experiment, the geographical distribution of user, UAV, and BS is shown in Figure 4. There are ten users, three UAVs, and a BS distributed in an area of 400 m × 400 m. The service radius of BS and UAV is set to 80 m and 43 m, respectively, the user task size is set to 50 MB, and the maximum tolerated time delay is set to 15 s. The computing frequencies of the UAV-MEC server and user are 20 GHz and 1 GHz, respectively. The UAV allocatable bandwidth is set to 80 Mbps. Table 1 gives the main parameters.

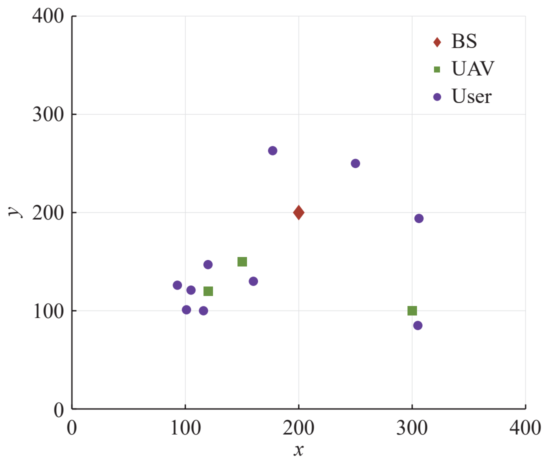

Figure 4. Scatter plot of geographical distribution of users, UAVs and the BS, which are distributed in an area of 400 m × 400 m.

Table 1 Experiment Parameters

Notation	Definition	Value
λ	The weight parameter that weighs the proportion of delay and energy consumption in T1	0.9
μ	Weight coefficients of T1 and T2	0.5
H	UAV hovering height (m)	30
R_j	Service radius of UAV j (m)	43
R_0	Service radius of BS (m)	80
C_i	The number of CPU cycles (cycles/bit) required for the one-bit task of user i	20
S_i	user i task size (MB)	50
T_i	Task maximum tolerated delay of user i (sec)	15
f_i	Local calculation CPU frequency of user i (GHz)	1
f_j	Allocatable computing CPU frequency of UAV j (GHz)	20
P_i	Transmission power of user i (W)	2
P_j^{max}	Allocatable transmission power of UAV j (W)	20
σ^2	Communication channel noise power (mW)	4^{-18}
k	CPU capacitance coefficient	5×10^{-27}
C_j	Maximum number of users that UAV j can serve	3
η_0	Channel power gain at 1 m distance (dB)	−30
B_{i0}	Communication bandwidth allocated by the BS to user i (MHz)	1.8
B_{j0}	Communication bandwidth allocated by the BS to UAV j (MHz)	20
$B_{jj'}$	Communication bandwidth between two UAVs (MHz)	20
B_j^{max}	Maximum allocatable bandwidth for communication between UAV j and users(Mbps)	80
B_0^{max}	Maximum allocatable bandwidth for communication between BS and users (Mbps)	400

5.2. Experiment Analysis

The experimental results include two parts, verify the convergence of our proposed algorithm EGAC-3MEC and its superiority compared with baseline algorithms.

a) Evaluation of the impact of parameters on algorithm performance. In this part, we will evaluate the influence of different parameters on the algorithm's performance. By systematically changing the parameter values and then evaluating the corresponding algorithm output, we can analyze how these parameters affect both the efficiency and convergence of the algorithm. This will empower us to identify the optimal parameters for our algorithm.

In Figure 5, it can be seen that as the number of iterations increases, the total cost gradually decreases and eventually stabilizes at a small value. A reduced crossover probability directs the algorithm towards a more meticulous local search, resulting in a slower convergence rate but a solution closer to the optimal one. Upon comparing four datasets featuring distinct crossover probabilities, it is observed that the total cost is minimized when the crossover probability is set at 0.2.

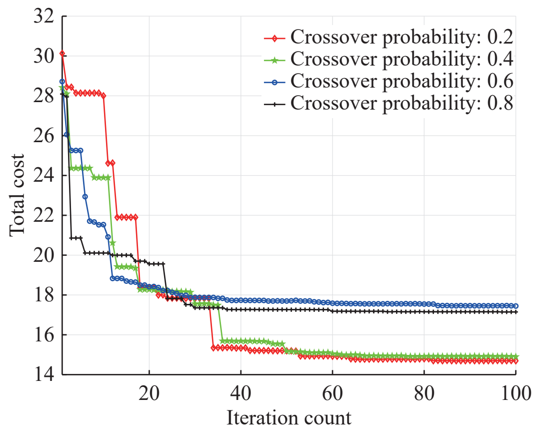

Figure 5. When the population crossover probability differs in the genetic algorithm, the total cost changes with the number of iterations.

In Figure 6, it is evident that a higher mutation probability accelerates system convergence and yields a smaller convergence value. This phenomenon can be attributed to the mechanism within the population mutation process of EGAC-3MEC, where the mutated individual is only substituted if it surpasses the worst individual in the initial population. Upon analyzing four datasets characterized by varying mutation probabilities, it is found that the total cost is minimized when the mutation probability is set at 0.9.

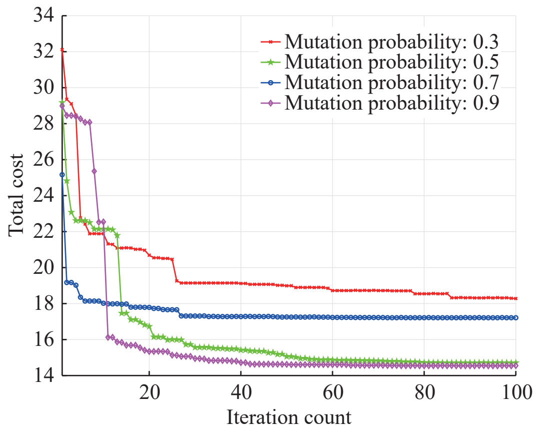

Figure 6. When the population mutation probability differs in the genetic algorithm, the total cost changes with the number of iterations.

In Figure 7, we find that a moderate population size can result in quicker convergence and a reduced total cost. When the population size is too small, there is a risk of the algorithm getting trapped in a local optimal solution. Conversely, an excessively large population can intensify competition among individuals, diminishing the algorithm's diversity and global search capabilities.

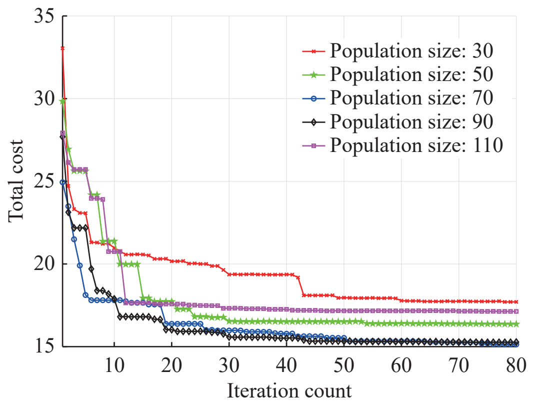

Figure 7. When the population size differs in the genetic algorithm, the total cost changes with the number of iterations.

Figure 8 illustrates that the convergence value depends on different task sizes. This dependency arises from the fact that distinct task sizes correspond to diverse problem spaces, ultimately leading to the attainment of varying optimal solutions throughout the optimization process.

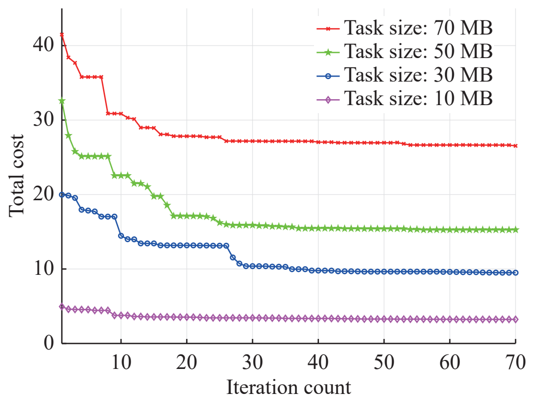

Figure 8. When the task size to be calculated is different, the total cost changes with the number of iterations.

In Figure 9, it can be found that a decrease in the proportion of high-priority users correlates with a reduction in the convergence value. When the objective function incorporates the delay of high-priority users, minimizing the delay for these users necessitates a higher energy investment, thereby elevating the total cost. Consequently, when the proportion of high-priority users is limited, the system is more likely to achieve a lower convergence value.

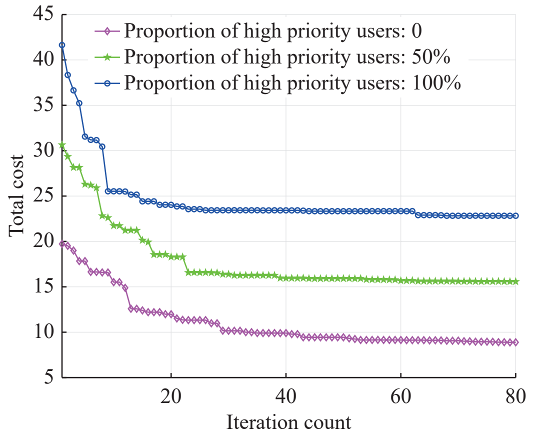

Figure 9. When the proportion of high-priority users differs, the total cost changes with the number of iterations.

b) Comparison with Baseline Algorithms. In what follows, we will compare the proposed EGAC-3MEC with four baselines including the local computing algorithm, proximity offloading algorithm, the minimum incremental task allocation (MITA), and EGAC-3MEC without UAV collaboration.

- *Local computing:* Users execute their computing tasks locally on personal devices, without external dependencies like UAVs or BSs. Local computing diminishes communication delays, bolsters data privacy, and alleviates network congestion. Nonetheless, it might be constrained by the computational capabilities of individual devices, potentially impeding their capacity to efficiently manage intricate or large-scale tasks.
- *Proximity offloading algorithm:* In cases where a user lacks the capacity to handle a task locally, it is wirelessly transmitted to the nearest device capable of executing it. Offloading tasks to UAVs or BSs capitalizes on their superior computing and storage capacities, thereby alleviating the computational burden on user devices and prolonging their battery life.
- *MITA*[42]: With MITA approaches, users assess and select offloading destinations (BS and UAV) for their tasks in a sequential manner. Users evaluate potential offloading locations and calculate the incremental system cost associated with each option. Subsequently, users choose the offloading operation that minimizes the overall cost increase. This method guarantees efficient task allocation by minimizing the constraints on system resources while optimizing performance.
- *EGAC-3MEC (without UAV collaboration):* In the EGAC-3MEC framework without UAV collaboration, tasks are transferred to UAVs, which have the option to either handle them locally or transfer them to BSs. In contrast to collaborative versions, this setup limits UAVs from redistributing tasks to other available UAVs once they have been received.

In Figure 10, it can be observed that EGAC-3MEC exhibits superior performance across various task sizes. The local computing approach incurs the highest total cost as it requires more time to finish tasks due to constrained computing resources. The total costs of the proximity offloading algorithm and MITA are relatively low and comparable because they aim to distribute tasks and resources evenly among the accessible UAVs or BSs. The proximity offloading algorithm simply picks the closest UAV or BS for task offloading, whereas in MITA, users have the flexibility to select the UAV or BS with the least total system cost increase for offloading. This feature aids in identifying efficient offloading choices with reduced total system costs. Furthermore, EGAC-3MEC leverages the collaborative advantages of UAVs to their fullest extent. It not only directly contributes the computing power but also redistributes user tasks to other available idle UAVs or BSs, thereby enhancing system resource utilization and decreasing the total cost. By harnessing UAV collaboration, EGAC-3MEC efficiently optimizes task offloading and resource allocation, leading to a reduction in the overall system cost. This collaborative approach enables UAVs to distribute computing tasks across different nodes, thereby alleviating the load on individual nodes and enhancing overall performance. In comparison to EGAC-3MEC without UAV collaboration, EGAC-3MEC with collaboration achieves a lower total cost, which shows the significance of UAV collaboration in optimizing task offloading and resource utilization.

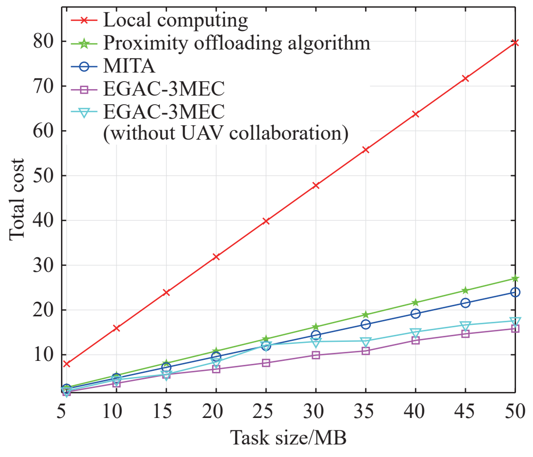

Figure 10. Comparison of the total cost of the EGAC-3MEC algorithm with baseline algorithms including local computing, proximity offloading algorithm, MITA, and EGAC-3MEC without UAV collaboration at different task sizes (MB).

In Figure 11, EGAC-3MEC shows a small total system delay at each task size. By optimizing the collaboration mechanism, EGAC-3MEC can significantly reduce waiting time and transmission delay, thereby improving system responsiveness. If the UAV collaboration feature is removed from EGAC-3MEC, the total delay will increase, which further verifies the effectiveness of UAV collaboration in the entire system.

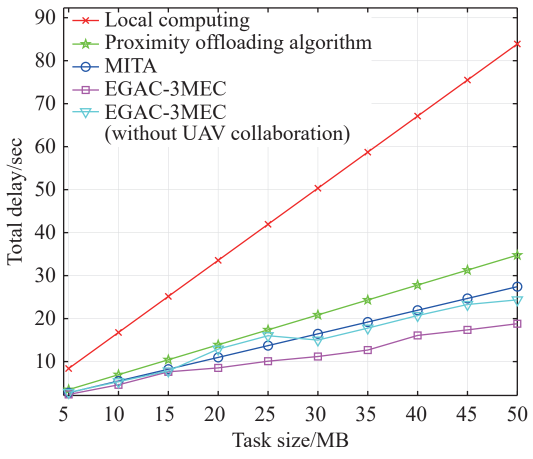

Figure 11. Comparison of the total time delay of the EGAC-3MEC algorithm with baseline algorithms including local computing, proximity offloading algorithm, MITA, and EGAC-3MEC without UAV collaboration at different task sizes (MB).

In Figure 12, we can see that EGAC-3MEC exhibits superior performance at different task sizes. EGAC-3MEC demonstrates superior energy efficiency compared to local computing, proximity offloading, and MITA. However, EGAC-3MEC with UAV collaboration has slightly higher energy consumption than EGAC-3MEC without UAV collaboration. This highlights the necessity of balancing energy consumption against time delays.

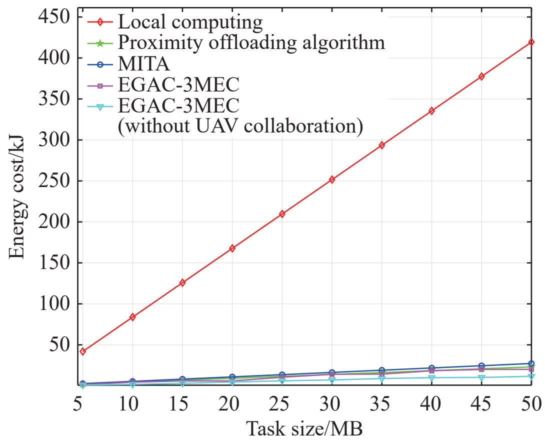

Figure 12. Comparison of the total energy consumption of the EGAC-3MEC algorithm with baseline algorithms including local computing, proximity offloading algorithm, MITA, and EGAC-3MEC without UAV collaboration at different task sizes (MB).

Figure 13 shows the latency of each user for EGAC-3MEC and the baseline algorithms when the task size is 50 MB, where the high-priority users are those with even-numbered IDs. EGAC-3MEC not only supports local computation but also focuses on minimizing latency for these high-priority users, thereby enhancing their service priority. On the other hand, local computation, proximity offloading, and MITA do not explicitly account for this priority distinction in their optimization processes.

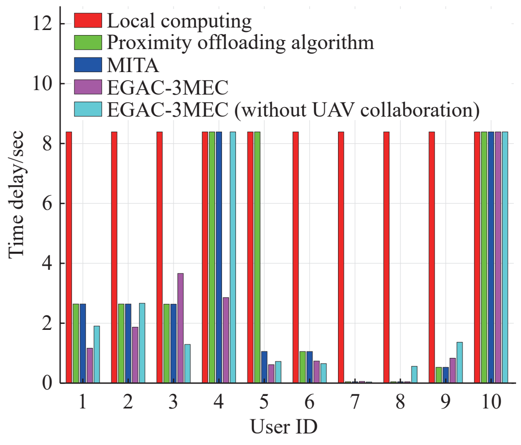

Figure 13. Comparison of time delay of each user of the EGAC-3MEC algorithm with baseline algorithms including local computing, proximity offloading algorithm, MITA, and EGAC-3MEC without UAV collaboration when the task size is 50 MB

6. Conclusion and Future Works

In this paper, a UAV collaborative three-layer MEC system has been investigated where UAVs have been utilized to directly provide computational support to users and act as communication relays for transmitting tasks to the BS. Moreover, UAVs (capable of task offloading to other available UAVs) have been used to achieve load balancing within the UAV network. An optimization problem has been formulated, taking into account system delay, energy consumption, and prioritized user delays. To address this complex problem, the EGAC-3MEC algorithm has been proposed, which first divides the problem into three sub-problems: user task offloading and resource allocation, UAV task offloading, and UAV resource allocation. These sub-problems have been tackled respectively using an improved genetic algorithm, a branch-and-bound method, and an interior point method. The numerical results have demonstrated that EGAC-3MEC exhibits rapid convergence, and outperforms the four baseline algorithms in terms of the overall performance. Additionally, the system under EGAC-3MEC operation has displayed a heightened service priority for high-priority users, showcasing the algorithm's effectiveness in optimizing system efficiency and user satisfaction.

While this paper proposes a comprehensive solution for cooperative MEC systems with UAVs, there are still several promising directions for future research. One potential area of exploration is the further optimization of UAV deployment and trajectory planning to enhance overall network performance. Further research is needed on dynamic and real-time UAV deployment and path optimization, which should take into account varying mission requirements, environmental conditions, and user mobility. The integration of reinforcement learning and adaptive optimization techniques could lead to more effective solutions. Additionally, studying multi-UAV collaboration will be critical for improving both performance of task offloading and resource allocation.

Author Contributions: Jiale Lin: Conceptualization, Methodology, Software, Writing–original draft; **Yao Nie:** Methodology, Software, Validation; **Yue Chen:** Supervision, Conceptualization, Methodology, Writing–review and editing.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

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

References

1. Abrar, M.; Ajmal, U.; Almohaimeed, Z. M.; *et al*. Energy efficient UAV-enabled mobile edge computing for IoT devices: A review. IEEE Access, **2021**, *9*: 127779−127798. doi: 10.1109/ACCESS.2021.3112104
2. Yang, B.; Cao, X. L.; Yuen, C.; *et al*. Offloading optimization in edge computing for deep-learning-enabled target tracking by internet of UAVs. IEEE Internet Things J., **2021**, *8*: 9878−9893. doi: 10.1109/JIOT.2020.3016694
3. Zhang, L.; Ansari, N.. Optimizing the operation cost for UAV-aided mobile edge computing. IEEE Trans. Veh. Technol., **2021**, *70*: 6085−6093. doi: 10.1109/TVT.2021.3076980
4. Zhang, L.; Chakareski, J.. UAV-assisted edge computing and streaming for wireless virtual reality: Analysis, algorithm design, and performance guarantees. IEEE Trans. Veh. Technol., **2022**, *71*: 3267−3275. doi: 10.1109/TVT.2022.3142169
5. Lakhan, A.; Ahmad, M.; Bilal, M.; *et al*. Mobility Aware Blockchain Enabled offloading and scheduling in vehicular fog cloud computing. IEEE Trans. Intell. Transport. Syst., **2021**, *22*: 4212−4223. doi: 10.1109/TITS.2021.3056461
6. Chen, H. W.; Deng, S. G.; Zhu, H. Z.; *et al*. Mobility-aware offloading and resource allocation for distributed services collaboration. IEEE Trans. Parall. Distr. Syst., **2022**, *33*: 2428−2443. doi: 10.1109/TPDS.2022.3142314
7. Wang, Z.; Zhao, Z. W.; Min, G. Y.; *et al*. User mobility aware task assignment for Mobile Edge Computing. Future Gener. Comput. Syst., **2018**, *85*: 1−8. doi: 10.1016/j.future.2018.02.014
8. Chen, X.; Liu, G. Z. Federated deep reinforcement learning-based task offloading and resource allocation for smart cities in a mobile edge network. Sensors, **2022**, *22*: 4738. doi: 10.3390/s22134738
9. Peng, Q. L.; Xia, Y. N.; Feng, Z.; *et al*. Mobility-aware and migration-enabled online edge user allocation in mobile edge computing. In *2019 IEEE International Conference on Web Services (ICWS)*, *Milan*, *Italy*, *8–13 July 2019*; IEEE: New York, 2019; pp. 91–98. doi: 10.1109/ICWS.2019.00026.
10. Saleem, U.; Liu, Y.; Jangsher, S.; *et al*. Mobility-aware joint task scheduling and resource allocation for cooperative mobile edge computing. IEEE Trans. Wirel. Commun., **2021**, *20*: 360−374. doi: 10.1109/TWC.2020.3024538
11. Zhan, W. H.; Luo, C. B.; Min, G. Y.; *et al*. Mobility-aware multi-user offloading optimization for mobile edge computing. IEEE Trans. Veh. Technol., **2020**, *69*: 3341−3356. doi: 10.1109/TVT.2020.2966500
12. Ei, N. N.; Alsenwi, M.; Tun, Y. K.; *et al*. Energy-efficient resource allocation in multi-UAV-assisted two-stage edge computing for beyond 5G networks. IEEE Trans. Intell. Transp. Syst., **2022**, *23*: 16421−16432. doi: 10.1109/TITS.2022.3150176
13. Xiong, K.; Liu, Y.; Zhang, L. T.; *et al*. Joint optimization of trajectory, task offloading, and CPU control in UAV-assisted wireless powered fog computing networks. IEEE Trans. Green Commun. Netw., **2022**, *6*: 1833−1845. doi: 10.1109/TGCN.2022.3157735
14. Lu, W. D.; Ding, Y.; Gao, Y.; *et al*. Resource and trajectory optimization for secure communications in dual unmanned aerial vehicle mobile edge computing systems. IEEE Trans. Ind. Inf., **2022**, *18*: 2704−2713. doi: 10.1109/TII.2021.3087726
15. Dai, B.; Niu, J. W.; Ren, T.; *et al*. Towards energy efficient scheduling of UAV and base station hybrid enabled mobile edge computing. IEEE Trans. Veh. Technol., **2022**, *71*: 915−930. doi: 10.1109/TVT.2021.3129214
16. Zhou, H.; Wang, Z. N.; Min, G. Y.; *et al*. UAV-aided computation offloading in mobile-edge computing networks: A Stackelberg game approach. IEEE Internet Things J., **2023**, *10*: 6622−6633. doi: 10.1109/JIOT.2022.3197155
17. Goudarzi, S.; Soleymani, S. A.; Wang, W. W.; *et al*. UAV-enabled mobile edge computing for resource allocation using cooperative evolutionary computation. IEEE Trans. Aero. Elec. Syst., **2023**, *59*: 5134−5147. doi: 10.1109/TAES.2023.3251967
18. Apostolopoulos, P. A.; Fragkos, G.; Tsiropoulou, E. E.; *et al*. Data offloading in UAV-assisted multi-access edge computing systems under resource uncertainty. IEEE Trans. Mobile Comput., **2023**, *22*: 175−190. doi: 10.1109/TMC.2021.3069911
19. Wang, R.; Cao, Y.; Noor, A.; *et al*. Agent-enabled task offloading in UAV-aided mobile edge computing. Comput. Commun., **2020**, *149*: 324−331. doi: 10.1016/j.comcom.2019.10.021
20. Guo, Y. H.; Zhao, R.; Lai, S. W.; *et al*. Distributed machine learning for multiuser mobile edge computing systems. IEEE J. Sel. Top. Signal Process., **2022**, *16*: 460−473. doi: 10.1109/JSTSP.2022.3140660
21. Liu, W. S.; Li, B.; Xie, W. C.; *et al*. Energy efficient computation offloading in aerial edge networks with multi-agent cooperation. IEEE Trans. Wirel. Commun., **2023**, *22*: 5725−5739. doi: 10.1109/TWC.2023.3235997
22. Zhao, N.; Ye, Z. Y.; Pei, Y. Y.; *et al*. Multi-agent deep reinforcement learning for task offloading in UAV-assisted mobile edge computing. IEEE Trans. Wirel. Commun., **2022**, *21*: 6949−6960. doi: 10.1109/TWC.2022.3153316
23. Zhou, W. Q.; Fan, L. S.; Zhou, F. S.; *et al*. Priority-aware resource scheduling for UAV-mounted mobile edge computing networks. IEEE Trans. Veh. Technol., **2023**, *72*: 9682−9687. doi: 10.1109/TVT.2023.3247431
24. Qin, Z.; Wei, Z. H.; Qu, Y. B.; *et al*. AoI-aware scheduling for air-ground collaborative mobile edge computing. IEEE Trans. Wirel.

Commun., **2023**, *22*: 2989−3005. doi: 10.1109/TWC.2022.3215795
25. Xu, Y.; Zhang, T. K.; Loo, J.; *et al*. Completion time minimization for UAV-assisted mobile-edge computing systems. IEEE Trans. Veh. Technol., **2021**, *70*: 12253−12259. doi: 10.1109/TVT.2021.3112853
26. Hu, J. W.; Jiang, M.; Zhang, Q.; *et al*. Joint optimization of UAV position, time slot allocation, and computation task partition in multiuser aerial mobile-edge computing systems. IEEE Trans. Veh. Technol., **2019**, *68*: 7231−7235. doi: 10.1109/TVT.2019.2915836
27. Gao, N.; Jin, S.; Li, X. 3D deployment of UAV swarm for massive MIMO communications. In *Proceedings of the ACM MobiArch 2020 The 15th Workshop on Mobility in the Evolving Internet Architecture*, *New York*, *NY*, *USA*, *21 September 2020*; Association for Computing Machinery: New York, 2020; pp. 24–29. doi: 10.1145/3411043.3412502.
28. Seid, A. M.; Boateng, G. O.; Anokye, S.; *et al*. Collaborative computation offloading and resource allocation in multi-UAV-assisted IoT networks: A deep reinforcement learning approach. IEEE Internet Things J., **2021**, *8*: 12203−12218. doi: 10.1109/JIOT.2021.3063188
29. Liu, Y.; Xie, S. L.; Zhang, Y.. Cooperative offloading and resource management for UAV-enabled mobile edge computing in power IoT system. IEEE Trans. Veh. Technol., **2020**, *69*: 12229−12239. doi: 10.1109/TVT.2020.3016840
30. Chai, F. R.; Zhang, Q.; Yao, H. P.; *et al*. Joint multi-task offloading and resource allocation for mobile edge computing systems in satellite IoT. IEEE Trans. Veh. Technol., **2023**, *72*: 7783−7795. doi: 10.1109/TVT.2023.3238771
31. Li, J. L. Y.; Yi, C. Y.; Chen, J. Y.; *et al*. Joint trajectory planning, application placement, and energy renewal for UAV-assisted MEC: A triple-learner-based approach. IEEE Internet Things J., **2023**, *10*: 13622−13636. doi: 10.1109/JIOT.2023.3262687
32. He, X. F.; Jin, R. C.; Dai, H. Y.. Multi-hop task offloading with on-the-fly computation for multi-UAV remote edge computing. IEEE Trans. Commun., **2022**, *70*: 1332−1344. doi: 10.1109/TCOMM.2021.3129902
33. Kong, P.; Li, B.; Wang, Y. H.; *et al* Multi-UAV cooperative computational delay and energy consumption modeling and DDPG optimization. In *2022 IEEE 8th International Conference on Computer and Communications (ICCC)*, *Chengdu*, *China*, *9–12 December 2022*; IEEE: New York, 2022; pp. 719–724. doi: 10.1109/ICCC56324.2022.10065948.
34. Tun, Y. K.; Dang, T. N.; Kim, K.; *et al*. Collaboration in the sky: A distributed framework for task offloading and resource allocation in multi-access edge computing. IEEE Internet Things J., **2022**, *9*: 24221−24235. doi: 10.1109/JIOT.2022.3189000
35. Qi, X. H.; Chong, J. Z.; Zhang, Q. Y.; *et al*. Collaborative computation offloading in the multi-UAV fleeted mobile edge computing network via connected dominating set. IEEE Trans. Veh. Technol., **2022**, *71*: 10832−10848. doi: 10.1109/TVT.2022.3188554
36. Xia, J. M.; Wang, P.; Li, B.; *et al*. Intelligent task offloading and collaborative computation in multi-UAV-enabled mobile edge computing. China Commun., **2022**, *19*: 244−256. doi: 10.23919/JCC.2022.04.018
37. Zhang, H. X.; Yang, Y. J.; Shang, B. D.; *et al*. Joint resource allocation and multi-part collaborative task offloading in MEC systems. IEEE Trans. Veh. Technol., **2022**, *71*: 8877−8890. doi: 10.1109/TVT.2022.3174530
38. Hu, H.; Chen, Z.; Zhou, F. H.; *et al*. Joint resource and trajectory optimization for heterogeneous-UAVs enabled aerial-ground cooperative computing networks. IEEE Trans. Veh. Technol., **2023**, *72*: 8812−8826. doi: 10.1109/TVT.2023.3244812
39. Meng, K. T.; He, X. F.; Wu, Q. Q.; *et al*. Multi-UAV collaborative sensing and communication: Joint task allocation and power optimization. IEEE Trans. Wirel. Commun., **2023**, *22*: 4232−4246. doi: 10.1109/TWC.2022.3224143
40. Guo, H. Z.; Wang, Y. T.; Liu, J. J.; *et al*. Multi-UAV cooperative task offloading and resource allocation in 5G advanced and beyond. IEEE Trans. Wirel. Commun., **2024**, *23*: 347−359. doi: 10.1109/TWC.2023.3277801
41. Boyd, S. P.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, 2004.
42. Zhao, L.; Yang, K. Q.; Tan, Z. Y.; *et al*. Vehicular computation offloading for industrial mobile edge computing. IEEE Trans. Ind. Inf., **2021**, *17*: 7871−7881. doi: 10.1109/TII.2021.3059640

Citation: Lin, J; Nie, Y.; Chen, Y. Extended Genetic Algorithm for Unmanned Aerial Vehicle Collaboration in Three-Layer Mobile Edge Computing Task Allocation and Optimization. *International Journal of Network Dynamics and Intelligence*. 2025, 4(3), 100015. doi: 10.53941/ijndi.2025.100015