Dependency Tasks Offloading and Communication Resource Allocation in Collaborative UAV Networks

A Metaheuristic Approach

Nguyen, Loc X.; Tun, Yan Kyaw; Dang, Tri Nguyen; Park, Yu Min; Han, Zhu; Hong, Choong Seon

Published in:
IEEE Internet of Things Journal

DOI (link to publication from Publisher):
10.1109/JIOT.2022.3233667

Publication date:
2023

Document Version
Accepted author manuscript, peer reviewed version

Citation for published version (APA):
Nguyen, L. X., Tun, Y. K., Dang, T. N., Park, Y. M., Han, Z., & Hong, C. S. (2023). Dependency Tasks Offloading and Communication Resource Allocation in Collaborative UAV Networks: A Metaheuristic Approach. *IEEE Internet of Things Journal*, *10*(10), 9062-9076. https://doi.org/10.1109/JIOT.2022.3233667

Dependency Tasks Offloading and Communication Resource Allocation in Collaborative UAV Networks: A Metaheuristic Approach

Loc X. Nguyen, Yan Kyaw Tun, *Member, IEEE*, Tri Nguyen Dang, Yu Min Park, Zhu Han, *Fellow, IEEE*, and Choong Seon Hong, *Senior Member, IEEE*

***Abstract*—Nowadays, unmanned aerial vehicles (UAVs)-assisted mobile-edge computing (MEC) systems have been exploited as a promising solution for providing computation services to mobile users outside of terrestrial networks. However, it remains challenging for standalone UAVs to meet the computation requirement of numerous users due to their limited computation capacity and battery lives. Therefore, we propose a collaborative scheme among UAVs to share the workload between them. Furthermore, this work is the first to consider the task topology of offloading in MEC-enabled UAVs networks while restricting their power consumption. We study the task topology, in which a task consists of a set of subtasks, and each subtask has dependencies upon other subtasks. In the real world, subtasks with dependencies must wait for their preceding subtasks to complete before being executed, and this affects the offloading strategy. Next, we formulate an optimization problem to minimize the average latency of users by jointly controlling the offloading decision for dependent tasks and allocating the communication resources of UAVs. The formulated problem is NP-hard and cannot be solved in polynomial time. Therefore, we divide the problem into two subproblems: 1) offloading decision problem and 2) communication resource allocation problem. Then, a metaheuristic method is proposed to find the suboptimal solution to the former problem, while the latter problem is solved by using convex optimization. Finally, we conduct simulation experiments to prove that our proposed offloading technique outperforms several benchmark schemes in minimizing the average latency of users for dependency tasks and achieving higher uplink transmission rates.**

***Index Terms*—Collaborative unmanned aerial vehicles (UAVs) network, communication resource allocation, directed acyclic graph (DAG) tasks, discrete whale optimization algorithm (D-WOA), offloading dependency subtasks.**

Manuscript received 26 August 2022; revised 6 December 2022; accepted 23 December 2022. Date of publication 2 January 2023; date of current version 9 May 2023. This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korea Government (MSIT) (No. 2020R1A4A1018607), Institute of Information & Communications Technology Planning & Evaluation (IITP) Grant funded by MSIT (Artificial Intelligence Innovation Hub) under Grant 2021-0-02068, IITP grant funded by MSIT (No. 2019-0-01287, Evolvable Deep Learning Model Generation Platform for Edge Computing), and IITP grant funded by MSIT (No. RS-2022-00155911, Artificial Intelligence Convergence Innovation Human Resources Development (Kyung Hee University). *(Corresponding author: Choong Seon Hong.)*

Loc X. Nguyen, Tri Nguyen Dang, Yu Min Park, and Choong Seon Hong are with the Department of Computer Science and Engineering, Kyung Hee University, Yongin 17104, Gyeonggi, Republic of Korea (e-mail: xuanloc088@khu.ac.kr; trind@khu.ac.kr; yumin0906@khu.ac.kr; cshong@khu.ac.kr).

Yan Kyaw Tun is with the Division of Network and Systems Engineering, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, 114 28 Stockholm, Sweden, and also with the Department of Computer Science and Engineering, Kyung Hee University, Yongin 17104, Gyeonggi, Republic of Korea (e-mail: yktun@kth.se).

Zhu Han is with the Electrical and Computer Engineering Department, University of Houston, Houston, TX 77004 USA, and also with the Department of Computer Science and Engineering, Kyung Hee University, Yongin 17104, Gyeonggi, Republic of Korea (e-mail: hanzhu22@gmail.com).

Digital Object Identifier 10.1109/JIOT.2022.3233667

I. Introduction

Over the last decade, the number of mobile users has grown exponentially, which eventually promotes the development of wireless communication and networking management to guarantee the Quality of Services (QoS) for all users [1]. Along with that growth, computation-intensive applications, such as virtual reality (VR), natural language processing, and fast navigation have become necessary components of our lives. Nonetheless, mobile users need help catching up with data processing due to the restrictions on computing capacity and battery life. Mobile cloud computing (MCC), with large storage space, high processing speed, and unlimited energy resources, relieves the pressure on mobile users by offering its resources [2], [3]. Nevertheless, as more and more mobile users appear, MCC is running into various problems, including high latency, poor coverage, weak security, and sluggish data transmission [4]. Therefore, a new concept in the computing landscape has been invented to replace the outdated one: multiaccess edge computing (MEC), which can resolve the above issues in the MCC system. The term "edge computing" refers to data processing and storage that takes place at the "edge" of a network close to the user [5]. By doing this, we not only drastically lower the transmission latency since MEC servers are substantially closer to the users than cloud servers but also avoid the peak in traffic flows. Normally, we deploy the computing servers at the network's edge, such as cellular base stations (BSs) or wireless access points (APs).

In the case of user devices operating in rural areas, such as mountains, forests, deserts, or underwater, and in temporary events, e.g., rescue operations in disaster locations or military operations, installing a new BS or AP requires lots of time and is a waste of money for one-time use. Therefore, Zhang et al. [6] and Zhou et al. [7] have proposed ideas that use unmanned aerial vehicles (UAVs), such as drones, high-altitude platforms, and balloons, as communication platforms for providing services to mobile users. These vehicles are flexible to deploy in any special area and can be reused for different scenarios, for instance, MEC-enabled UAVs being deployed to serve IoT devices in [8] and [9]. Moreover, MEC-enabled UAVs can collect data and perform computing tasks for those devices that do not have any direct access to terrestrial BSs or APs [10]. In addition, MEC-enabled UAV servers can be used as relay stations for efficiently expanding the communication coverage.

Even though the MEC-enabled UAV system has great potential and provides better and faster computation services than conventional fixed-location MEC systems, some challenges still stand in the way of adopting this technique in real world. First, the MEC-enabled UAV system has lower computing capacities and energy resources than the cloud server and the traditional MEC system. Therefore, it is eager to find an optimal offloading strategy for MEC-enabled UAV systems to lower the task latency under such limited conditions [11], [12]. Second, each MEC-enabled UAV system has to support numerous devices over a broad coverage area while having fewer bandwidth resources to allocate, compared to the communication resources of the MEC-enabled BS system. Therefore, an optimization algorithm is required to allocate the bandwidth resources among users instead of distributing them equally. Another approach to guarantee the data rate requirement of users is optimizing the UAV trajectory [13]. Moreover, the server attached to the UAV has much lower computing resources compared to the cloud server and the servers attached to the terrestrial BSs. As a consequence, the task from users may fail to finish on time even though only part of it is offloaded. Therefore, the collaboration among UAVs has been proposed as a key for overcoming the task's deadline and also boosting the energy efficiency.

Task offloading has been one of the main features in edge computing scenarios, which decides how the task will be executed. In most cases, the task from a user is parameterized by two criteria: 1) the input size and 2) the number of CPU cycles required to finish the task. By considering this way, they can easily decompose the task into small subtasks, offload them to edge devices, and enable parallel execution on both edge and mobile devices. However, the dependency of the subtasks might have a significant influence on this parallelism [14]. The dependency among subtasks can be simply understood as the start of a subtask happens only after the completion of all the predecessor subtasks as it requires the output information from the predecessor subtasks [15]. As a result, these dependent subtasks must be executed sequentially rather than in parallel. A directed acyclic graph (DAG) is often used to demonstrate the dependencies among subtasks. Each circle denotes a subtask of the whole task, and the directed edge represents the topology among them, and each subtask has a different input length, computing requirement, and result requirements. Hence, the scheduling decision becomes much more complicated, and appears to be an NP-hard problem.

Recently, most user applications consist of a series of subtasks. For example, Li and Jain [16] discussed the process of a face recognition task, comprising four subtasks: 1) face detection; 2) face alignment; 3) feature extraction; and 4) feature matching, in which these subtasks have to be performed sequentially due to the dependency among them. The outputs from face detection (i.e., face location, size, and pose) will be used as the inputs of the face alignment subtask. The aligned face enters the feature extraction module to get a feature vector. Then, that feature vector will be compared with the features in the database to find the best match and output the face ID in the final subtask. This face recognition task is just a simple example of the dependency inside the task, and there are still a lot of tasks that we have to consider the dependency inside.

To the best of our knowledge, this is the first study that considers the task dependency for the offloading problem in a MEC-enabled UAVs network. On top of that, we utilize the computing resource of idle UAVs in the network by proposing collaboration among UAVs to reduce the latency of the tasks. In addition, to make our system model more practical, we consider the energy limitation of UAVs. The main contributions of this article are summarized as follows.

1) We propose a task model, which includes multiple subtasks and the dependency among them to capture the task topology in the real scenario. Then, those tasks from users will be offloaded to MEC-enabled UAVs networks. The BS, which is the central controller, will decide the offloading strategy based on its knowledge of the tasks and networks.
2) We mathematically formulate an optimization problem of the proposed system model to minimize the average latency of the mobile users by controlling the offloading decision and the communication resources allocated to each user. The formulated problem has been proven to be NP-hard.
3) We decompose the problem into two subproblems: a) dependent task offloading decision problem and b) bandwidth allocation problem for users. Then, we propose discrete whale optimization algorithm (D-WOA), a metaheuristic approach to solve the decision problem, which is hard to be solved in polynomial time by traditional methods. The problem of allocating communication resources is convex; thus, we can use the splitting conic solver (SCS) in CVXPY to solve it.
4) Finally, we conduct a series of simulations to evaluate the efficiency of the proposed method and compare it against benchmark schemes, such as the exhaustive search algorithm (ESA) and associated UAVs. We also show how the restricted energy of the UAVs affects the execution latency of the tasks.

The remainder of this article will be organized as follows. We will briefly discuss related works in Section II. Then, Section III describes the system model and problem formulation in more detail. Our proposed solution will be demonstrated in-depth in Section IV. We show the simulation results and, based on those results, we draw a conclusion in Sections V and VI, respectively.

II. Related Works

A. Multiaccess Edge Computing

This section will discuss a summary of the literature on multiaccess edge computing. Yang et al. [17] proposed an artificial fish swarm algorithm (AFSA) to solve the task offloading problem with the assistance of femto relay BSs. Their objective is to find the access and offloading decisions that return the least energy consumption. Sun et al. [18] maximized the sum of computing efficiency by a traditional optimization approach: the iterative gradient descent method. A two-stage heuristic optimization algorithm was proposed in [19] to allocate the communication resources and offload the computing tasks from multiple users to multiple servers using the least amount of energy. Xia et al. [20] proposed a DUPA^3 game to allocate data, users, and power for caching problems in MEC to maximize the total data rate and the number of served users. The maximization of the total data rate leads to a low latency network, which is beneficial for mobile users. Liu et al. [21] considered the data caching problem from the service provider's view. They proposed an approximation approach to solve the data caching problem with the goal of optimizing service providers' revenue. Most recently, Zhu et al. [22] minimized the task computation delay by determining the NOMA-based transmission duration and workload offloading allocation among edge devices. While Fan et al. [23] formulated the task offloading and resource allocation problem for MEC-enabled vehicles, the difference is that they considered both moving and parked vehicles in the scenario.

B. MEC-Enabled UAVs Networks

The MEC-enabled UAV has received lots of attention from research scientists for its many advantages. Tun et al. [24] proposed a block successive upper bound minimization (BSUM) approach to minimize the energy consumed by IoT devices and UAV-aided MEC systems by finding the optimal offloading decision, resource allocation, and UAV route. Yu et al. [25] formulated a problem for minimizing the weighted sum of service delay for IoT devices and the energy consumption of UAV by controlling the UAV location, computation resource, communication resource, and the splitting ratio of the task. Then, the successive convex approximation-based algorithm was proposed to find a suboptimal solution. Zhang et al. [26] optimized the offloading decision to minimize the cost of time and energy consumed by the system while considering the limited energy of the UAV-aided MEC, and proposed a game-theory-based solution to find the offloading decision.

The above-mentioned studies considered UAVs to operate independently from each other in the MEC network. Tun et al. [27] and Seid et al. [28] proposed the collaboration among MEC-enabled UAVs in task computing and proved its effectiveness. Tun et al. [27] optimized the offloading decision, the computing and communication resources under the energy limitation of participating users and UAVs to obtain the lowest latency of all tasks. Compared with the previous paper, the collaboration among MEC-enabled UAVs provides more destinations for the task to be offloaded, taking the pressure off the associated UAV. Seid et al. [28] proposed CCORA-DRL for each MEC-enabled UAV to learn an efficient computation offloading strategy and, therefore, acquired minimum service latency. Besides the resource optimization problem, researchers recently considered secure transmission in MEC-enabled UAVs networks [29], [30]. Lu et al. [29] proposed successive convex approximation and block coordinate descent algorithms to maximize the user's minimum secure calculation capacity, while [30] proposed multiagent reinforcement learning to optimize the secure offloading. Pang et al. [31] proposed an intelligent reflecting surface (IRS) to facilitate security in communication between the user and the UAV. The formulated problem is to maximize the average secrecy rate by jointly optimizing the trajectory of UAV, transmit beamforming and the phase shift of IRS. However, these works do not consider the task topology in the offloading scheme, which can degrade the system's performance.

C. Offloading Dependent Tasks

This section will provide a summary of the literature on offloading dependent tasks. Sundar and Liang [32] were the first to study scheduling and offloading decisions for tasks comprising dependent subtasks in the MCC system. They aimed to minimize the task execution cost subject to the task completion deadline by using individual time allocation with greedy scheduling (ITAGS) to effectively solve the proposed NP-hard problem. In comparison, the work in [33] considered the edge servers to execute the dependent tasks. The formulated problem became more complicated than the MCC system due to the heterogeneous computing resources of the edge servers and the requirement for information exchange among subtasks. They came up with the distributed earliest finish-time offloading algorithm to effectively reduce the latency of the dependent task from IoT devices. In [34], the dependent task was generated by vehicular edge computing (VEC), and they considered minimizing the execution latency of the tasks by optimizing the offloading strategy for each subtask: local execution at the VEC or offloading to the roadside unit. Then, they proposed multiple applications multiple tasks scheduling (MAMTS) algorithm to calculate the subtask prioritization and schedule the subtask according to the priority.

Fan et al. [15] introduced a new metric to evaluate the efficiency of offloading strategy for dependent tasks and then proposed a heuristic offloading solution to reduce the cost of the process. Zhao et al. [35] formulated offloading dependent tasks with service catching problems to minimize the makespan of the tasks and then designed a convex programming-based approach to tackle the problem. Sahni et al. [36] jointly optimized the offloading decisions and the network flow in the collaborative edge computing systems to minimize the completion time of the tasks. Most currently, Wang et al. [37] proposed deep reinforcement learning with the assistance of a Sequence-to-Sequence neural network to solve the offloading problem for dependent tasks. However, most of these works assumed that the edge server had a large

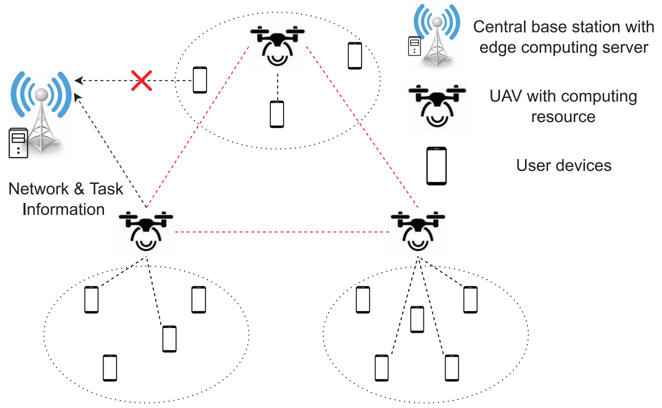

Fig. 1. Network model.

computing capacity and infinite power resources; therefore, this assumption will not hold if we employ MEC attached to the UAVs. These works did not consider the reduction in computation resource for one subtask when multiple subtasks were being offloaded simultaneously, or they assumed the edge executed one subtask at a time [36].

III. System Model and Problem Formulation

A. Overview of System Model

Network Model: As shown in Fig. 1, we have a set $\mathcal{V}$ of V MEC-enabled UAVs to provide communication and computation services to users in a certain area, and these UAVs can connect with the terrestrial BS by both Line-of-Sight (LoS) and non-LoS (NLoS) networks. In this article, we assume that there is no direct communication link between mobile users and the BS or that users are not within the BS's service region. The BS is equipped with a computing server and serves as a centralized controller with collective knowledge about the network. It is in charge of selecting UAVs to offload subtasks and resource allocation. The BS must gather information about each user's task through the third-party UAVs because there is no direct connection between BS and the users. To reduce the complexity of the problem, we assume that users are already assigned to the serving UAVs, determined by the channel quality. In particular, each UAV $v \in \mathcal{V}$ knows its connected set of users, $\mathcal{U}_v$. The total number of mobile users in our network is defined as follows $\bigcup_{v=1}^{V} \mathcal{U}_v$, where there is no user repetition among different sets.

Task Model: Each user can generate only one task at a time. Hence, we consider a task generated by the user u as task u. User $u \in \mathcal{U}_v$ has different requirements and structure D_u that can be modeled as DAG $D_u = (T_u, P_u)$, where T_u is a set of dependent subtasks in task u, $T_u = \{j|1 \leq j \leq N_u\}$, P_u is a set of dependencies between subtasks in task u, and N_u is the total number of subtasks in task u. The computation requirement to finish the subtask j of task u depends on its input size, which is denoted as $H_{u,j}$. The amount of data dependency between subtasks p and j is denoted as $D^u_{p,j}$. If the preceding subtasks of j are located in different devices, we have to transmit $D^u_{p,j}$ to the responsible UAV before we start the subtask j. From now on, the set of predecessors and successors of subtask j

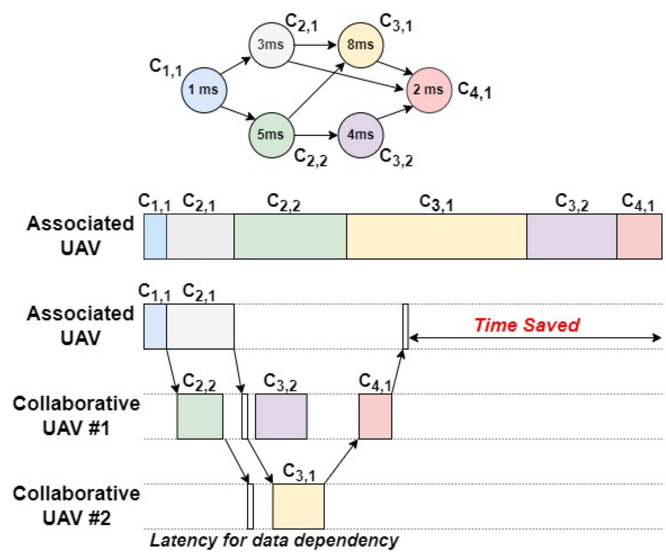

Fig. 2. Case study for task model in the proposed scenario.

are defined as $Pd_{u,j}$ and $Sc_{u,j}$, respectively. Besides, there are some subtasks that we can do simultaneously because they are independent of each other.

We illustrate a case study for offloading a dependent task in a single UAV compared with multicollaborative UAVs. As shown in Fig. 2, the DAG represents our dependent task, which has six nodes indicating six subtasks, while the directed edges express the dependency between two subtasks. For example, the subtask $c_{1,1}$ can be executed without any conditions, while the subtasks $c_{2,1}$ and $c_{2,2}$ only be executed when receiving the result (dependency data) from the subtask $c_{1,1}$, the same explanation can be applied to other subtasks. Furthermore, we also compare the latency of the dependent task when UAVs in the network operate independently and collaboratively. UAVs that working independently of each other have to execute all the subtasks sequentially. In the collaborative scheme, we can utilize the idle computing resource and execute some subtasks parallelly, such as subtask $c_{2,1}$ and $c_{2,2}$. As shown in Fig. 2, instead of executing subtask $c_{2,2}$ in the associated UAV with subtask $c_{2,1}$, we transmit $c_{2,2}$ along with the result from subtask $c_{1,1}$ to the collaborative UAV number one with more powerful computing resources and execute the subtask there. As a result, we can lower the latency of the dependent task. However, one notice here is that we have to transfer the dependency data between two subtasks if they are not executed at the same UAV.

In this article, we assume that the task is created by a local user, and to capture this assumption, we add one dummy subtask at the start of the task. As the dummy subtask has zero execution time and zero communication cost, it does not make any changes to the original task. Thus, the number of subtask is increased to N'_u, where $N'_u = N_u + 1$. The devices of mobile users have low computing capacity and energy, and so these limitations yield challenges for them to process the task quickly. Therefore, users offload the task to the associated UAVs in collaborative UAVs networks, which have a larger computing capacity to finish the task via wireless links within

TABLE I
SUMMARY OF NOTATIONS

Notation	Definition	Notation	Definition
$\mathcal{V}$	Set of UAVs, $\|\mathcal{V}\|$= V	$\mathcal{U}$	Set of users, $\|\mathcal{U}\|$= U
$\mathcal{D}_u$= ($\mathcal{T}_u$,$\mathcal{P}_u$)	Directed acyclic graph of task u, where $\mathcal{T}_u$ is the set of dependent sub-tasks and $\mathcal{P}_u$ is the set of edges	$\mathcal{D}^u_{p,j}$	The size of dependent data between sub-task p and j of task u
N_u	Total number of sub-tasks in task u	I_v	Set of tasks information at UAV v
$S_{u,j}$	Set of successor sub-tasks of sub-task j in task u	$P_{u,j}$	Set of predecessor sub-tasks of sub-task j in task u
$H_{u,j}$	Input size of sub-task j of task u	T^{dis}_u	Starting time for distributing and executing task u
t^{load}_u	Transmission latency of task u from the user to its associate UAV	$t^{\text{dep}}_{u,p,j}$	Latency for transmitting dependent data from UAV execute sub-task p to UAV execute sub-task j
$FT_{u,j}$	Finish time of sub-task j of task u	$ST_{u,j}$	Start time of sub-task j of task u
$t^{\text{exe}}_{u,j}$	Execution latency of sub-task j of task u	C_u	Number of CPU cycles required to execute one bit data
$\Gamma^{u\to v}$	Spectrum efficiency from user u to its associate UAV	P_u	Transmit power of user u
$G_{v,u}$	Achievable channel gain between the user and UAV	σ^2	Gaussian noise power n
$d^{u,v}$	Distance from user to its associated UAV	$R^{u\to v}$	Achievable uplink transmission rate
β^v_u	Percentage of bandwidth that will be allocated to user u	B^v	Total accessible bandwidth that UAV v can allocate to its users
$t^{u\to v}$	Data transmission latency from user to UAV	$E^{v,\text{up}}$	Energy consumed by user to transmit task u to UAV v
$\theta^{u\to v}_{u,j}$	Decision variable whether sub-task j of user u is offloaded to its associate UAV v or not	$\gamma^{v\to w}_{u,j}$	Binary variable which decides sub-task j will be offloaded to the collaborated UAV w in the network
f^v_u	Computing capacity of the MEC server of associated UAV	$F^{\max}_v$	Maximum computing resource of UAV v
$t^{v,\text{exe}}_{u,j}$	The amount of latency to execute the sub-task at UAV v	$E^{v,\text{exe}}_{u,j}$	Energy consumption for execute sub-task j
$R^{v\to w}$	Achievable data rate from UAVs v to w	P_v	The amount of power required by the UAV v
$G^{v,w}$	Channel gain between two collaborative UAVs	$d^{v,w}$	Distance between UAVs v and w
$E^{v\to w}_{u,j}$	Energy consumption required to distribute sub-task j from UAV v to UAV w	E^{tol}_v	Total energy consumption of UAV v
$P^{v,\text{hov}}$	Hovering power of UAV v	$t^{\text{exe}}_{u,j}$	Execution latency of sub-task j of task u
$E^{v,\text{hov}}$	Hovering energy of UAV v	$t^{v,\text{hov}}$	Hovering time of UAV v
$t^{v\to b}$	Transmit latency from UAV v to the BS	$R^{v\to b}$	Communication data rate from the user to the BS
$B^{v\to b}_{\text{mm}}$	The amount of bandwidth in mmWave frequency band allocated to UAV v	$P_{b,v}$	The energy received at the BS
$P^{v\to b}$	The amount of power required to transmit information from UAV v to the BS	G^{tx}_v	Antenna gain of UAV v
G^{rx}_b	Antenna gain of the BS	d^b_v	Distance from the BS to UAV v
B^{mm}_c	Carrier freqency of the mmWave link	$E^{v\to b}$	Energy consumption for transmitting information from UAV v to the BS

the minimum amount of time. In the next section, we will explain our proposed idea by discussing the task and network models. The key notations are presented in Table I.

B. Constraints of Task and Network Models

1) Task Model: As mentioned before, a dummy subtask is added to ensure the task is created at user u. The start time of the dummy subtask is set as the time when the UAV receives the offloading decision and starts to distribute the task. Since it does not require any computation resource, its finish time of it is defined in [36] as follows:

$$X_{uN'_u} = 1 \ \forall u \in \mathcal{U}_v \tag{1}$$

$$ST_{uN'_u} = T^{\text{dis}}_u \ \forall u \in \mathcal{U}_v \tag{2}$$

$$FT_{uN'_u} = T^{\text{dis}}_u \ \forall u \in \mathcal{U}_v \tag{3}$$

where T^{dis}_u is the start time for distributing and executing task u. Since the dummy subtask does not require any computing resources, it is the only subtask executed at the local site. After the decision is made at the BS, the subtasks from the user have to be transferred to the responsible UAV, and this process takes time. We define the latency to transmit the input data of subtask j as $t^{\text{load}}_{u,j}$. The arrival time of subtask j of user u will be given by the following equation:

$$AT_{u,j} = T^{\text{dis}}_u + t^{\text{load}}_{u,j} \ \forall j \in N_u \ \forall u \in \mathcal{U}_v. \tag{4}$$

The start time of a subtask depends not only on the finish time of preceding subtasks $Pd_{u,j}$ and the transmit latency of result data $D^u_{p,j}$ but also on the arrival time of subtask j. The relationship between the start time of the subtask and those time variables can be expressed by the following equation:

$$ST_{u,j} \geq \max\left\{ \max_{\forall p \in Pd_{u,j}} \left(FT_{u,p} + t^{\text{dep}}_{u,p,j}\right);\ AT_{u,j} \right\} \\ \forall p \in Pd_{u,j} \ \forall j \in N_u \ \forall u \in \mathcal{U}_v. \tag{5}$$

The second way to include the transmitting time of the subtask into our problem is to distribute the subtask right after the decision is made at the BS. In this case, the start time can be given by

$$ST_{u,j} \geq \max_{\forall p \in Pd_{u,j}} \left(FT_{u,p} + t^{\text{dep}}_{u,p,j}\right) + AT_{u,j} \\ \forall p \in Pd_{u,j} \ \forall j \in N_u \ \forall u \in \mathcal{U}_v \tag{6}$$

where $t^{\text{dep}}_{u,p,j}$ is the latency for transmitting dependent data from other devices, which is the result of the preceding subtask. In case subtask j needs multiple result data, we will have to wait until the last one is finished and transmitted to the UAV that is responsible for executing subtask j. However, when all dependent subtasks of subtask j are located at the same device with subtasks j, this latency becomes zero. The transmission latency of the dependent data $D^u_{p,j}$ between subtasks p and j of task u depends on the amount of data we need to send and the channel quality between UAVs, and it can be determined as

$$t^{\text{dep}}_{u,p,j} = \frac{D^u_{p,j}}{R_{v,w}} \ \forall p \in Pd_{u,j} \ \forall j \in N_u \ \forall u \in \mathcal{U}_v \tag{7}$$

where $R_{v,w}$ is the achievable data rate between UAVs. Then, the finishing time for subtask j will be calculated by taking the sum of its start time and the execution time

$$FT_{u,j} = ST_{u,j} + t^{\text{exe}}_{u,j} \ \forall j \in N_u \ \forall u \in \mathcal{U}_v \tag{8}$$

where $t^{\text{exe}}_{u,j}$ is the execution time of the subtask j, which will be defined in later. The completion time of a task will be represented by (9), which is the difference between the time instance when the last subtask is finished and the start time of the dummy subtask

$$F_u = \max_{j=1,\ldots,N_u} FT_{u,j} - ST_{uN'}. \tag{9}$$

2) Offloading to the Associated UAV: A number of subtasks will be offloaded to the associated UAV $v \in \mathcal{V}$ through the wireless communication link. According to [38], the achievable spectrum efficiency from user u to UAV v can be determined by the following equation:

$$\Gamma^{u\to v} = \log_2\left(1 + \frac{P_u G^v_u}{\sigma^2}\right) \ \forall u \in \mathcal{U}_v \ \forall v \in \mathcal{V} \tag{10}$$

in which P_u is the power of user u and G^v_u is the channel gain between user u and UAV v, while σ^2 is the white Gaussian noise power. Generally, the channel gain between mobile device u and UAV v is acquired by

$$G^v_u = 10^{-\delta^v_u/10} \ \forall u \in \mathcal{U}_v \ \forall v \in \mathcal{V} \tag{11}$$

where δ^v_u is the path loss between UAV and user u. Furthermore, in this work, we consider not only the LoS but also the NLoS links for air-to-ground communication: in particular, from users to UAVs and backward. Then, the path loss δ^v_u is the combination of two types of path losses: 1) LoS path loss, $\delta^{v,\text{LoS}}_u$ and 2) N-LoS path loss, $\delta^{v,\text{NLoS}}_u$. The formulation of these path losses is defined in [39] as follows:

$$\delta^{v,\text{LoS}}_u = 10n\log\left(\frac{4\pi d^{u,v} B^{\text{lte}}_c}{c}\right) + L_{\text{LoS}} \tag{12}$$

$$\delta^{v,\text{NLoS}}_u = 10n\log\left(\frac{4\pi d^{u,v} B^{\text{lte}}_c}{c}\right) + L_{\text{NLoS}} \tag{13}$$

where n $\geq$ 2 is the path-loss exponent, B^{lte}_c denotes the carrier frequency (i.e., 2 GHz), c is the speed of light, L_{LoS} and L_{NLoS} are the average added losses for LoS and NLoS links. Additionally, $d^{u,v}$ is the distance from mobile user to UAV, and can be acquired by the following equation:

$$d^{u,v} = \sqrt{(x_v - x_u)^2 + \left(y_v - y_y\right)^2 + z_v^2}. \tag{14}$$

The next thing we consider is the probability of existing LoS connectivity between UAV v and user u, which is denoted in [40] as follows:

$$Pr^{v,\text{LoS}}_u = \frac{1}{1 + C.\exp\left[-D\left(\frac{180}{\pi}\sin^{-1}\left(\frac{z_v}{d^{u,v}}\right) - C\right)\right]} \tag{15}$$

where z_v is the hovering altitude of UAV v. While C and D are variables that depend on the operating environment, such as city or countryside and others. As a result, the likelihood of the existing NLoS link from the mobile user to the associated UAV is acquired by the following equation:

$$Pr^{v,\text{NLoS}}_u = 1 - Pr^{v,\text{LoS}}_u. \tag{16}$$

Then, the path loss from user u to the associated UAV is calculated by the following equation:

$$\delta^v_u = Pr^{v,\text{LoS}}_u \delta^{v,\text{LoS}}_u + Pr^{v,\text{NLoS}}_u \delta^{v,\text{NLoS}}_u \ \forall u \in \mathcal{U}_v \ \forall v \in \mathcal{V}. \tag{17}$$

Back to the spectrum efficiency between user u and UAV v, the achievable uplink transmission rate of device u is determined in [27] as follows:

$$R^{u\to v} = \beta^v_u B^v \Gamma^{u\to v} \ \forall u \in \mathcal{U}_v \ \forall v \in \mathcal{V} \tag{18}$$

where β^v_u is the percentage of total available bandwidth of UAV v (denoted as B^v) that will be allocated to user u.

Thus, the transmission latency from user u to UAV v depends on the uplink transmission rate and can be formulated as follows:

$$t^{u\to v} = \frac{\sum_{j=1}^{\mathcal{T}_u} H_{u,j}}{R^{u\to v}} \tag{19}$$

where $H_{u,j}$ is the input size of subtask j belong to task u. Then, the energy consumed by up-link transmitting from user to UAV is denoted as follows:

$$E^{v,\text{up}}_u = \frac{P_u \sum_{j=1}^{\mathcal{T}_u} H_{u,j}}{R^{u\to v}}. \tag{20}$$

The offloaded subtasks can be executed at the associated UAV if its computing resource is available. And the decision variable used to represent a subtask will be offloaded to the associated UAV v is defined as follows:

$$\theta^v_{u,j} = \begin{cases} 1, & \text{if subtask } j \text{ of user } u \text{ is offloaded to} \\ & \quad \text{associated UAV } v \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

In this case, the latency for transmitting the input data of subtask j is given as follows:

$$t^{\text{load}}_{u,j} = t^{u\to v}_{u,j}. \tag{22}$$

When the subtask is executed at UAV v, the computing time of that subtask can be given as

$$t^{\text{exe}}_{u,j} = \frac{C_u H_{u,j}}{f^v_{u,j}} \ \forall u \in \mathcal{U}_v \ \forall v \in \mathcal{V} \tag{23}$$

where $f^v_{u,j}$ is the computing resource (i.e., cycles/s) of the MEC server mounted on UAV v that is allocated to execute subtask j of user u. This computing resource will be determined by the input size of subtask j over all the executed subtasks at UAV v as defined in [41]

$$f^v_{u,j} = \frac{H_{u,j}}{\sum_{q\in U_v} H_{u,j}} F^{\max}_v \tag{24}$$

where $F_v^{\max}$ is the maximum available computing resource of UAV v. We have to ensure that the total computing capacity of UAVs across a number of subtasks from different users is less than or at least equal to the maximum capacity of the UAV, i.e.,

$$\sum_{q\in U_v} \theta_{u,j}^v f_{u,j}^v \le F_v^{\max} \ \forall v \in \mathcal{V}. \tag{25}$$

The following equation will be used to calculate how much energy of UAV v will be used to complete subtask j

$$E_{u,j}^{v,\text{exe}} = \mathrm{k}_v \left(f_{u,j}^v\right)^2 C_u Z_{u,j} \ \forall u \in \mathcal{U}_v \tag{26}$$

where $\mathrm{k}_v = 5 \times 10^{-27}$ is a constant that only changes according to the server's chip architecture mounted on the UAV.

3) Offloading to Collaborative UAVs: UAV v will pass on a part or whole of the task to other UAVs ($w \in \mathcal{V}$, $v \neq w$) in the network by a wireless link if its computing capacity is not sufficient to execute the task alone. Therefore, we need to define another binary variable $\gamma_{u,j}^w$ to indicate whether to forward the subtask j of user u from UAV v to UAV w or not. The definition of this binary variable is given as follows:

$$\gamma_{u,j}^w = \begin{cases} 1, & \text{if sub-task } j \text{ of user } u \text{ is forwarded to} \\ & \text{UAV } w \\ 0, & \text{otherwise.} \end{cases} \tag{27}$$

The constraint used to control each subtask being offloaded to only one UAV in the network is described by

$$\theta_{u,j}^v + \gamma_{u,j}^w = 1 \ \forall j \in N_u' \ \forall u \in \mathcal{U}_v \ \forall w \in \mathcal{V}. \tag{28}$$

The latency for transmitting subtask j from UAV v to UAV W is given as follows:

$$t_{u,j}^{v\to w} = \frac{H_{u,j}}{R^{v\to w}} \tag{29}$$

where $R^{v\to w}$ is the achievable rate from UAV v to UAV w and can be defined as

$$R^{v\to w} = B^{v,w} \log_2\left(1 + \frac{P_v G_{v,w}}{\sigma^2}\right) \tag{30}$$

where $B^{v,w}$ is the available bandwidth for direct communication between UAVs v and w, P_v is the power that UAV v requires for transmission, and $G^{v,w}$ is the achievable channel gain between two UAVs. Because the UAVs fly over the sky, we assume LoS link for UAV-to-UAV communication. Then, the channel gain between UAVs, which is defined in [42], as follows:

$$G^{v,w} = 10^{-(L_{v,w}/10)} \tag{31}$$

where $L_{v,w} = \Theta_{v,w} + \Gamma_{LoS}$ is the path loss between UAVs v and w. Here, Γ_{LoS} is the additional attenuation factor for the LoS link [42], and $\Theta_{v,w}$ is defined as follows:

$$\Theta_{v,w}(\text{dB}) = 20\log_{10}\left(d^{v,w}\right) + 20\log_{10}(f_c) + 10\log_{10}\left[\left(\frac{4\pi}{c}\right)^2\right] \tag{32}$$

where c is the speed of light and f_c is the carrier frequency (i.e., 2 GHz). $d^{v,w}$ is the distance between UAVs v and w, which can be expressed as follows:

$$d^{v,w} = \sqrt{(x_w - x_v)^2 + (y_w - y_v)^2 + (z_w - z_v)^2} \tag{33}$$

where $[x_w, y_w, z_w]$ is coordinate vector of UAV w in 3-D area.

If subtask j belongs to user u is offloaded to the collaborative UAV w, the latency for transmitting input data is considered the sum over two components: 1) the transfer time from the user to the associated UAV v and 2) the forward time to the collaborative UAV w. Therefore, it can be given by

$$t_{u,j}^{\text{load}} = t_{u,j}^{u\to v} + t_{u,j}^{v\to w}. \tag{34}$$

The energy consumed for transmitting input data from UAVs v to w in the network is given by

$$E_{u,j}^{v\to w} = P_v\left(\frac{H_{u,j}}{R^{v\to w}}\right) \ \forall u \in \mathcal{U}_v \ \forall v \in \mathcal{V}. \tag{35}$$

Once UAV w receives subtask j of user u, the latency $t_{u,j}^{\text{exe}}$, and the energy consumption $E_{u,j}^{w,\text{exe}}$ for executing it can be easily calculated based on (23) and (26).

4) Information Gathering at the BS: As mentioned before, the BS operates as a centralized controller and is responsible for offloading decisions. To do this, the MEC-enabled UAVs have to send their information: maximum computing capacity, available energy, along with the DAG of the tasks, which is denoted as I_v to the BS. The communication latency from UAV v to the ground BS for transmitting that information can be acquired by

$$t^{v\to b} = \frac{I_v}{R^{v\to b}} \ \forall v \in \mathcal{V} \tag{36}$$

where $R^{v\to b}$ is the achievable data rate of the mmWave link from UAV v to the ground BS, which can be calculated by [43]

$$R^{v\to b} = B_{\text{mm}}^{v\to b} \log_2\left(1 + \frac{P_{b,v}}{B_{\text{mm}}^{v\to b}\sigma^2}\right) \ \forall v \in \mathcal{V} \tag{37}$$

where $B_{\text{mm}}^{v\to b}$ is the bandwidth in the mmWave frequency band that the BS allocated to UAV v. $P_{b,v}$ is the received power at the BS, and calculated as follows:

$$P_{b,v} = P^{v\to b} G_v^{tx} G_b^{rx} \frac{c}{4\pi d_v^b B_c^{\text{mm}}} \tag{38}$$

where $P^{v\to b}$ is the transmit power to the BS and B_c^{mm} is the carrier frequency of the mmWave link. The distance d_v^b from UAV v to the BS is determined by

$$d_v^b = \sqrt{(x_b - x_v)^2 + (y_b - y_v)^2 + (z_v)^2} \tag{39}$$

where $[x_b, y_b]$ is the coordinate of the BS. From a couple of previous denotations, the energy consumption to transmit information from UAV v to the BS is defined as follows:

$$E^{v\to b} = P^{v\to b} \frac{I_v}{R^{v\to b}} \ \forall v \in \mathcal{V}. \tag{40}$$

5) Energy Consumption of the Associated UAV: The total energy usage of UAV v is the sum of the computation energy to execute the subtasks, the energy to forward the input data of subtasks to neighboring UAVs, the energy to transmit information to BS, and the energy consumption to hover at a fixed altitude. It is worth noting that since the length of the input data is much longer than the dependent data, we omit the energy for transmitting the dependent data. Thus, the total energy consumption can be presented as

$$E_v^{\mathrm{tol}} = \sum_{u \in U_v} \theta_{u,j}^u E_{u,j}^{v,\mathrm{exe}} + \sum_{v \in W, w \neq v} \gamma_{u,j}^w E^{v \to w} + E^{v \to b} + E^{v,\mathrm{hov}} \tag{41}$$

where $E^{v,\mathrm{hov}}$ is the hovering energy of UAV v and calculated as follows [44]:

$$\begin{aligned} E^{v,\mathrm{hov}} &= P^{v,\mathrm{hov}} t^{v,\mathrm{hov}} \\ &= \frac{\eta\sqrt{\eta}}{\varphi_v \sqrt{2\pi q r^2 \varkappa}} t^{v,\mathrm{hov}} \end{aligned} \tag{42}$$

where η is the trust that is proportional to the UAV's mass, φ_v is the power efficiency of UAV v, q denotes the number of rotors belonging to a single UAV, r is the diameter of each rotor, and $\varkappa$ is the air density. Finally, $t^{v,\mathrm{hov}}$ is the maximum hovering time of the UAV, and it is given by

$$\begin{aligned} t^{v,\mathrm{hov}} = \max_{u \in U_v} \Bigg[& t_u^{v,\mathrm{up}} + t^{v \to b} \\ & + \max\left(t_u^{v,\mathrm{exe}}; t_u^{v \to w,\mathrm{com}} + t_u^{v \to w,\mathrm{exe}} \right) \Bigg]. \end{aligned} \tag{43}$$

C. Problem Formulation

In this article, we investigate a realistic joint offloading decision and communication resource allocation problem with the objective of minimizing the average latency of all users in the network. Therefore, the objective function will be defined as

$$\boldsymbol{M}(\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\beta}) = \frac{1}{U_v} \sum_{u \in U_v} \left\{ F_u(\boldsymbol{\theta}, \boldsymbol{\gamma}) + t^{u \to v}(\boldsymbol{\beta}) \right\}. \tag{44}$$

At the time we are working on this article, there is no paper that considers both finding the offloading decision for the dependent task in a collaborative UAVs network and solving the optimal resource allocation for the associated users. The problem can be formulated as follows:

$$\begin{aligned} \underset{\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\beta}}{\text{minimize}} \quad & \boldsymbol{M}(\boldsymbol{\theta}, \boldsymbol{\gamma}, \boldsymbol{\beta}) && \text{(45a)} \\ \text{subject to} \quad & \theta_{u,j}^v + \gamma_{u,j}^w = 1 \\ & \qquad \forall j \in N_u' \ \forall u \in \mathcal{U}_v \ \forall w \in \mathcal{V} && \text{(45b)} \\ & \theta_{u,j}^v \in \{0, 1\} \ \forall j \in N_u' \ \forall u \in \mathcal{U}_v && \text{(45c)} \\ & \gamma_{u,j}^w \in \{0, 1\} \ \forall j \in N_u' \ \forall u \in \mathcal{U}_v \ \forall w \in \mathcal{V} && \text{(45d)} \\ & \sum_{u \in U_v} \beta_u^v \le 1 \ \forall v \in \mathcal{V} && \text{(45e)} \\ & \beta_u^v \in [0, 1] \ \forall u \in \mathcal{U}_v && \text{(45f)} \\ & E_v^{\mathrm{tol}} \le E_v^{\max} \ \forall v \in \mathcal{V} && \text{(45g)} \end{aligned}$$

where $\boldsymbol{\theta}$ is the offloading decision vector which has $\sum_{u=1}^{K} \sum_{j=1}^{N_u} 1$ elements and each element represents whether subtask j of task u offloads to the associated UAV or not, while $\boldsymbol{\gamma}$ is offloading decision vector to the collaborative UAVs in our network and the size of this vector will be $\sum_{w \in \mathcal{V}, w \neq v} \sum_{u=1}^{K} \sum_{j=1}^{N_u} 1$. $\boldsymbol{\beta}$ is the communication resource (i.e., bandwidth) allocation vector with each element β_u^v represents the fraction of bandwidth allocated to user $u \in \mathcal{U}_v$ at UAV $v \in \mathcal{V}$. The constraint (45b) ensures that the subtask has to be executed at one device and constraints (45c) and (45d) indicate the binary decision variables. Constraints (45e) and (45f) guarantee that the total fraction of bandwidth allocated to users should be less than the maximum available bandwidth at the UAV. Finally, constraint (45g) represents the energy constraint of each UAV.

In this problem, we have to assign each subtask of users' tasks to one MEC-enabled UAV, which can be simplified as a generalized assignment problem (GAP). GAP is an NP-hard problem. Therefore, our problem is also an NP-hard problem that we cannot find the optimal solution in polynomial time. Hence, a heuristic solution will be the best way to solve the problem.

IV. Proposed Solution

In order to address the formulated NP-hard problem, we first decompose the problem into two subproblems: 1) the offloading-dependent task problem and 2) the communication resource allocation problem. Then, we propose a metaheuristic approach so-called D-WOA to solve the subproblem in Section IV-A, and a standard optimization tool is applied to solve the subproblem in Section IV-B, respectively.

A. Offloading Dependent Task Problem

With the task information received from the MEC-enabled UAVs in the network, the BS is responsible for optimizing the offloading decisions. The first subproblem is formulated as follows:

$$\begin{aligned} \underset{\boldsymbol{\theta}, \boldsymbol{\gamma}}{\text{minimize}} \quad & \boldsymbol{M}(\boldsymbol{\theta}, \boldsymbol{\gamma}) && \text{(46a)} \\ \text{subject to} \quad & \theta_{u,j}^v + \gamma_{u,j}^w = 1 \\ & \qquad \forall j \in N_u' \ \forall u \in \mathcal{U}_v \ \forall w \in \mathcal{V} && \text{(46b)} \\ & \theta_{u,j}^v \in \{0, 1\} \ \forall j \in N_u' \ \forall u \in \mathcal{U}_v && \text{(46c)} \\ & \gamma_{u,j}^w \in \{0, 1\} \ \forall j \in N_u' \ \forall u \in \mathcal{U}_v \ \forall w \in \mathcal{V} && \text{(46d)} \\ & E_v^{\mathrm{tol}} \le E_v^{\max} \ \forall v \in \mathcal{V}. && \text{(46e)} \end{aligned}$$

By observing, we can see the coupling between $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$ in (46b), and beyond that, they are binary variables. With these statements, we can conclude that the proposed problem is an NP-hard problem. Normally, we can relax these binary variables into continuous and then apply the convex optimization method to tackle the problem. However, with $\mathcal{N}$ subtasks in a single task, the number of offloading decisions will increase exponentially and become impossible to solve in polynomial time. Therefore, we use a metaheuristic solution, the whale optimization algorithm (WOA), to mimic humpback whales' social hunting behavior. Before deploying the algorithm, we need to merge the offloading variable $\boldsymbol{\theta}$ and the offloading vector $\boldsymbol{\gamma}$ into a single vector, $X_u = [x_{u1}, x_{u2}, \ldots, x_{un}]$, where

X_u and x_{uj} are the offloading vectors for task u and offloading decision for subtask j of task u, respectively. The value of x_{uj} is discrete and it indicates which UAV is responsible for executing the subtask. The reformulated problem is given as follows:

$$\underset{\boldsymbol{\chi}}{\text{minimize}} \quad \boldsymbol{M(X)} \tag{47a}$$

$$\text{subject to} \quad x_{uj} \in [1, V] \ \forall j \in N'_u \ \forall u \in \mathcal{U}_v \tag{47b}$$

$$E_v^{\text{tol}} \le E_v^{\max} \ \forall v \in \mathcal{V}. \tag{47c}$$

WOA: In metaheuristic algorithms, they are divided into four classes: 1) evolutionary algorithms; 2) Physics-based algorithms; 3) swarm-based algorithms; and 4) human-based algorithms. WOA belongs to the swarm-based algorithm class, which has two phases in its searching process: 1) *exploration phase* and 2) *exploitation phase*. The former phase will help the optimizer explore the search space globally and, therefore, the movement of searching agents needs to be as randomized as possible in this phase. In contrast to this, the exploitation phase can be described as a process of examining the promising regions of the search space. The algorithm has been mathematically modeled based on the search for prey of humpback whales by [45]. Humpback whales have the ability to detect the position of their prey. When they find out the prey, they swim around and surround them by creating a bubble in a spiral shape [46]. The algorithm has three main mechanisms: 1) encircling; 2) bubble-net hunting; and 3) searching.

1) Encircling Prey: At the beginning of the search, the optimal solution is unknown to agents, and the algorithm considers the current best solution as the target prey so that other search agents will update their solutions toward the best searching agent. The following equations illustrate this behavior:

$$\vec{D} = |\vec{C} \cdot \vec{X}^*(t) - \vec{X}(t)| \tag{48a}$$

$$\vec{X}(t+1) = \vec{X}^*(t) - \vec{A} \cdot \vec{D} \tag{48b}$$

where $\vec{X}^*(t)$ denotes the current best solution, and it will be updated in each iteration if any agents find out better objective, $\vec{C}$ and $\vec{A}$ are coefficient vectors, t represents the number of the current iteration, the math operation $\cdot$ indicates element by element multiplication. Vectors $\vec{A}$ and $\vec{C}$ can be given by the following equations:

$$\vec{A} = 2\vec{a} \cdot \vec{r} - \vec{a} \tag{49a}$$

$$\vec{C} = 2 \cdot \vec{r} \tag{49b}$$

where $\vec{a}$ is linearly decreased from 2 to 0 over the course of iterations, and $\vec{r}$ is a random vector in [0,1]. This mechanism is called shrinking encircling in the original WOA paper [45].

2) Bubble-Net Hunting: As mentioned before, searching agents can locate the position of the prey and swim to the prey along a spiral path. This hunting mechanism can prevent prey from escaping and can be mathematically modeled as follows:

$$\vec{X}(t+1) = \vec{D}' \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}^*(t) \tag{50a}$$

$$\vec{D}' = |\vec{X}^*(t) - \vec{X}(t)| \tag{50b}$$

where $\vec{D}'$ is the distance between the ith whale and the optimal solution obtained so far, b is the constant for defining the shape of the logarithmic spiral, and l is a random number between $[-1, 1]$.

The humpback whales swim not only around the prey within a shrinking circle but also along a spiral-shaped path at the same time. In order to capture this simultaneous behavior, a random variable p is introduced. Its value falls in [0,1], and we will choose an action based on its value. The mathematical model is given as follows:

$$\vec{X}(t+1) = \begin{cases} \vec{X}^*(t) - \vec{A} \cdot \vec{D}, & \text{p} < 0.5 \\ \vec{D}' \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}^*(t), & \text{p} \ge 0.5. \end{cases} \tag{51}$$

The above equation will give both behaviors an equal probability of being picked.

3) Searching for Prey: The approach for the shrinking encircling mechanism will be reused for the search (exploration stage). When agents are in this stage, they search randomly according to the position of each other. The mechanism will only be active when $|\vec{A}| \ge 1$. The search agent updates its position far away from the reference whale, which is different from the encircling mechanism when $|\vec{A}| < 1$. The mathematical model of the searching phase will be as follows:

$$\vec{D} = |\vec{C} \cdot \vec{X}_{rand}(t) - \vec{X}(t)| \tag{52a}$$

$$\vec{X}(t+1) = \vec{X}_{rand}(t) - \vec{A} \cdot \vec{D}. \tag{52b}$$

4) Objective Function: With each decision matrix we obtain from the searching agents, we calculate the total latency, which will be the sum of distributed latency and computing latency. The distributed latency is the time to transmit the subtasks from the associated MEC-enabled UAV to the other MEC-enabled UAVs in the network. However, one important detail is that not all the solution given by searching agents is feasible due to energy constraint, while the original WOA is designed to address the unconstrained optimization problem. Therefore, to deal with the constraints, Pham et al. [47] introduced some constraint-handling techniques, such as the penalty method, feasibility rules, stochastic ranking, and split them into categories. In this article, we adopt the most simple and well-known technique: the penalty method in our objective. Therefore, the reformulated problem in (42) is transformed into penalty form as follows:

$$\underset{\boldsymbol{\chi}}{\text{minimize}} \quad M(\boldsymbol{X}) + \lambda \cdot G\left(E_v^{\text{tol}}, E_v^{\mathbf{max}}\right) \cdot \left(E_v^{\text{tol}} - E_v^{\mathbf{max}}\right)^2 \tag{53a}$$

$$\text{subject to} \quad x_{uj} \in [1, V] \ \forall j \in N'_u \ \forall u \in \mathcal{U}_v \tag{53b}$$

where $\text{G}(E^{\text{tol}}, E^{\max})$ is a step function conditioned by E^{tol} and $E^{\max}$. If the energy consumption given by offloading decision exceeds the maximum energy of any UAV in the network, the function $\text{G}(\cdot)$ returns the value 1; otherwise, it equals 0. The right term in our objective function is denoted as the penalty term, and λ is the penalty coefficient, which is normally adopted to manipulate the penalty value. If the penalty's

Algorithm 1 D-WOA for Offloading Decisions

Input: N: the number of searching agents, v: the number of UAVs in the network, n: the total number of sub-tasks and the maximum number of iterations $MaxIT$.

Output: The optimal decision X* and the optimal latency value f(X*).

Initialize the whale population, $X_u = [x_{u1}, x_{u2}, ..., x_{un}] \in [1, V]^n$ $(u = 1, 2, ..., N)$,
Calculate the latency of each agent decision and determine the optimal decision X*, $t \leftarrow 0$.
while $t \leq MaxIT$ **do**
    for $u \leftarrow 1$ to N (each searching agent) **do**
        Update vector a, A, C, l and p.
        if $p < 0.5$ **then**
            if $|A| < 1$ **then**
                Update the offloading decision by (48b).
            else
                Update the offloading decision by (52b).
            end if
        else if $p \geq 0.5$ **then**
            Update the offloading decision by (52a).
        end if
    end for
    Calculate the latency of each agent and update X* if there is a better offloading decision.
    $t \leftarrow t + 1$.
end while
Return(X*, f(X*)).

value is too small, then the proposed algorithm may converge to an infeasible solution and accept the punishment. On the other hand, the severe penalty prevents the agent from exploring more promising regions and staying in their comfort zone. The pseudo code of the proposed algorithm (WOA) is shown in Algorithm 1.

B. Communication Resource Allocation

In the beginning, users use a nonoptimal bandwidth to send their task information to the associated UAVs. When the UAVs have the task information of users in the coverage area, they not only send it to the BS for the optimal offloading decision but also use it to reallocate the communication bandwidth for their associated users. The second subproblem is presented below:

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \quad M(\beta) \tag{54a}$$

$$\text{subject to} \quad \sum_{u \in U_v} \beta_u^v \leq 1 \ \forall v \in \mathcal{V} \tag{54b}$$

$$\beta_u^v \in [0, 1] \ \forall u \in \mathcal{U}_v \ \forall v \in \mathcal{V}. \tag{54c}$$

As shown in (44), the offloading decision and the communication resource vectors are not coupled. Therefore, the resource allocation is not dependent on the offloading decisions from the BS. We can prove the convexity of the above problem and solve it using the CVXPY toolkit.

Algorithm 2 Sequence of the Proposed Scheme

Step 1 Tasks are generated at the user sites, their information are uploaded to the associated UAVs and the BS.
Step 2 With the task information:
The UAVs optimize the communication resource to the associated users for uploading the offloading task.
The BS solves the offloading decision problem by running WOA.
Step 3 The tasks stored at the associated UAVs are distributed and executed according to the offloading decisions.

C. Complexity Analysis of the Proposed Algorithm

The detail of the proposed scheme is given in Algorithm 2. Pham et al. [47] and Aung et al. [48] stated that the solution to each subproblem can be used to determine the complexity of the proposed algorithm. Here, we have two subproblems: offloading decisions and communication resource allocation problems. For the first subproblem, the computation complexity of the D-WOA normally depends on the number of searching agents N, the number of iterations $MaxIT$, and finally, the dimension of the searching agents. Particularly in our case, it depends on the number of subtasks $\mathcal{O}_1(N \cdot MaxIT \cdot M)$. However, due to the power constraint, the complexity of D-WOA increases and becomes $\mathcal{O}_1(N \cdot MaxIT \cdot (M + m))$, where m is the number of inequality constraints. In the resource allocation problem, the complexity is quite simple $\mathcal{O}_2(V \cdot U_v)$. Therefore, the computation complexity of our proposed approach to solve the problem is $\mathcal{O}(NMaxIT \cdot \ (M + m) + V \cdot U_v)$.

V. Performance Evaluation

In this section, we conduct a series of simulations to evaluate the performance of our algorithm and compare it with other benchmark solutions.

A. Simulation Settings

1) Settings for Network Model: We consider a rectangular region with the size of 1 km × 1 km that will be divided into four nonoverlapping areas, and each of them will be covered by a UAV at an altitude of around 50 m. The number of users in each subarea being served by the MEC-enabled UAV obeys a uniform distribution, ranging from [2, 10] users. Notice that not all of them have a task to execute at the same time. The BS is located at the center of the area within the communication range of all the UAVs. Table II provides detailed information on the simulation parameters.

2) Settings for Task Model: The topology of the task will be generated layer-by-layer, and the number of subtasks in each layer follows a normal distribution with $\mu = 2$ and $\sigma^2 = 1$. Instead of randomizing the input size of the whole task, we control the size of subtasks, which are selected from Gaussian distribution with a mean value of 6 MB (Mega Byte), and the variance value is 1 MB. The dependency information among subtasks is drawn from a uniform distribution [150, 250] kb. Finally, for the default case, each task has ten subtasks, and the number of active users (have a task for offloading) is 3.

TABLE II
SIMULATION PARAMETERS

Parameter	Value	Parameter	Value
B^v	3 MHz	$B^{v,w}$	8 MHz
P_v	30 dBm	P_u	23 dBm
f_c	2 GHz	$F_v^{\max}$	[800, 1000] MHz
σ^2	-174 dBm	C	11.9 [40]
D	0.136 [40]	η	30 N [44]
j	4 [44]	r	0.254 [44]
φ_v	70% [44]	N	100 agents
$MaxIT$	50	V	4

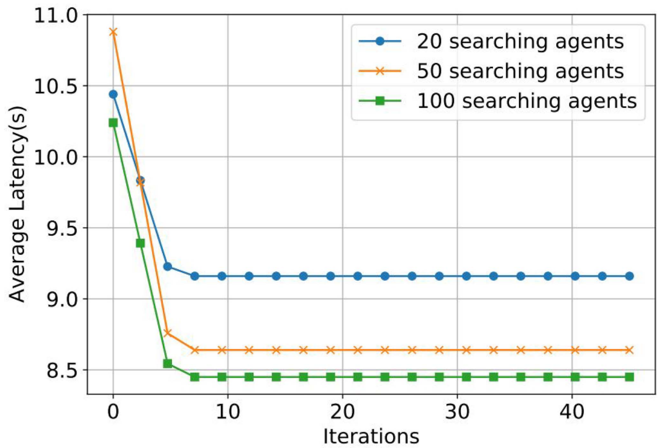

Fig. 3. Convergence of the proposed algorithm with different numbers of searching agents.

B. Benchmark Solutions

To assess the performance of our proposal, we compare our solution approach with the following schemes in terms of total latency.

1) *Associated UAV:* As the name it sounds, in this solution approach, the task generated by users will be transmitted to the corresponding associated UAVs. It should be pointed out that one user is only associated with one UAV, and there is no collaboration among MEC-enabled UAVs in this case. Therefore, each MEC-enabled UAV has to execute the tasks from the associated users by itself.
2) *ESA:* The offloading problem in (46a) can be considered as a knapsack problem. However, instead of a single knapsack, we have a set of knapsacks (UAVs). A subtask only needs to be executed by one UAV, thus the offloading decision matrix is restricted by this and the energy constraint. We attempt to generate all feasible solutions. Among them, we find the optimal solution with the lowest latency.

Fig. 3 illustrates the convergence of the D-WOA under the different numbers of searching agents. As mentioned before, the heuristic solution can solve the NP-hard problem in polynomial time, but the solution is just a local optimal. As shown in Fig. 3, when the number of searching agents increases, the optimal value also increases (i.e., achieving lower average latency). This phenomenon can be easily understood that the more agents participate in the search, the more likely they encounter a better solution. This average latency is calculated by using the offloading decision belonging to the best searching agent. In addition, most of the improvements are made in the early iterations, in which the value of $|\vec{A}|$ is large, indicating that the agents are in the exploration stage.

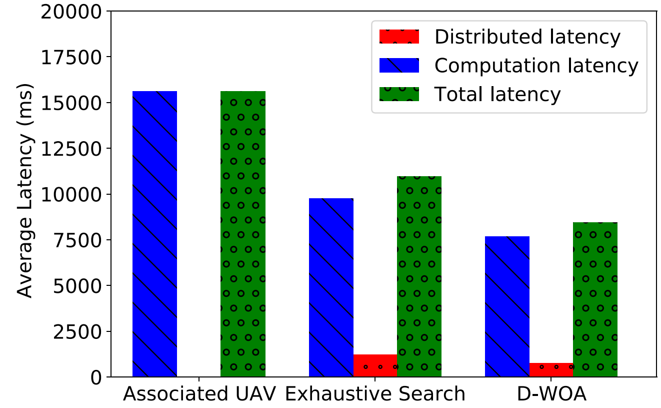

Fig. 4. Total average latency of the proposed algorithm compared with the benchmark scheme.

In Fig. 4, we compare our solution (D-WOA) for the offloading decision problem with the associated UAV and the exhaustive search approaches under different kinds of latency, such as distributed latency, computation latency, and total latency. As the name sounds, the computation latency is the amount of time MEC-enabled UAVs need to execute the subtasks. On the other hand, the distributed latency is the time for the associated UAV to distribute the subtasks to the other UAVs responsible for executing the subtasks. Therefore, in the associated UAV solution, the distributed latency equals 0. The distributed latency depends on the offloading decision of the algorithm. Finally, the total latency is the sum of two previous latencies and our objective function in (46a). In this scenario, the total latency under our proposed solution, exhaustive search, and associated UAV scheme are 8,446 ms, 10 968 ms, and 15 599 ms, respectively. Based on the outcomes above, our proposed D-WOA method outperforms other schemes. Particularly, 45.86% better than the associated scheme and 22.994% better than the exhaustive search. As we can see, the associated UAV scheme, in which there is no cooperation among UAVs, performs poorly compared to the ESA and proposed D-WOA. This outcome indicates the effectiveness of the collaboration of UAVs in the wireless network.

Fig. 5 shows the energy consumption of each UAV for ESA and D-WOA. As we can see, the D-WOA forces the UAVs to utilize their available resource effectively, while the solution provided by ESA only exploits 50% of the available energy. This result shows the tradeoff between the energy consumption of UAVs and the latency experienced by users. However, in this article, we focus on minimizing the latency while keeping the energy consumption under the threshold. Normally, the maximum energy of a UAV is around 200 kJ. Nevertheless, in this simulation, we focus on evaluating the proposed scheme under the limited energy so that the maximum energy of a MEC-enabled UAV will be set based on the number of subtasks as follows: $E_v^{\max} = N_u \times 2000$. As mentioned above, each task of a user has ten subtasks, so $E_v^{\max} = 20$ kJ.

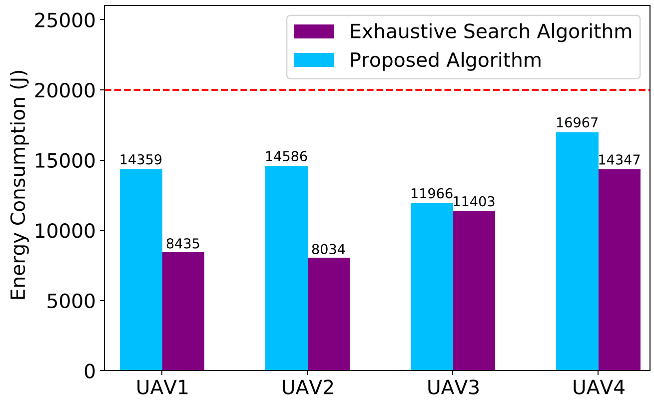

Fig. 5. Energy consumption by each UAV in the proposed algorithm and ESA.

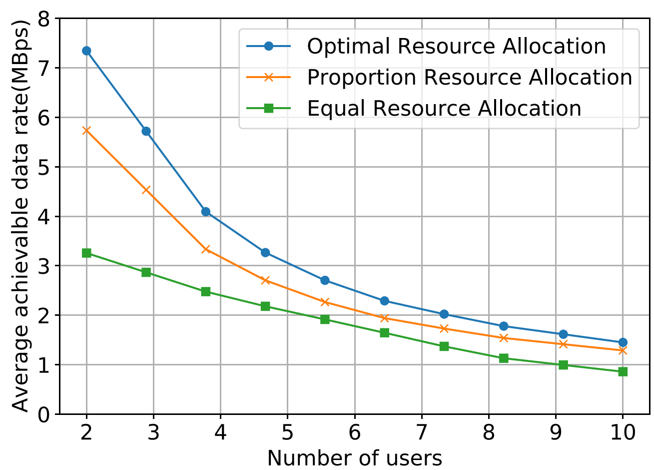

Fig. 6. Average data rate under different numbers of users.

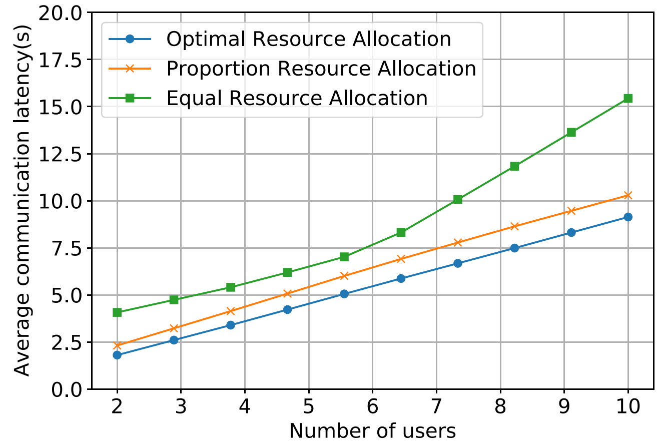

Fig. 7. Average communication latency under different number of users.

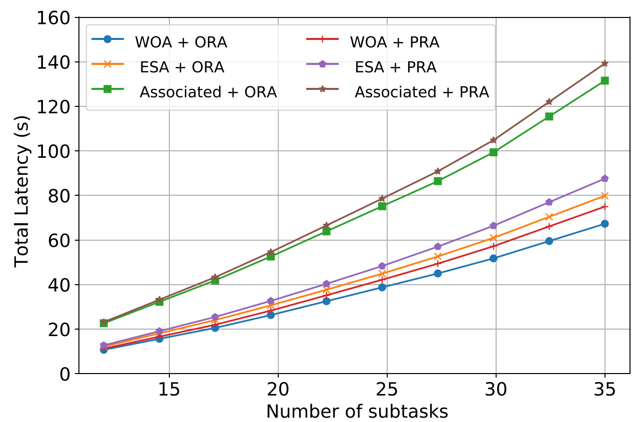

Fig. 8. Total latency of different algorithms versus the number of subtasks (unlimited energy).

When the number of subtasks per task increases, the maximum energy also increases to guarantee a feasible solution.

In Fig. 6, we show the average achievable data rate of the users communicating with its associated UAV in a single subarea, and the same way can be applied to other subareas. In the equal resource allocation scheme, all communication resources (i.e., bandwidth) available at the UAV are equally allocated among its associated users within the area, whether they need it or not in a fair manner. While in the proportion resource scheme, the MEC-enabled UAV allocates communication resources to the users based on the network information, i.e., the active users and the size of offloading tasks, as the following equation:

$$\beta^{v\to u} = \frac{H_{u,v}}{\sum_{q\in v} H_{u,v}} B^{v} \ \forall v \in \mathcal{V} \ \forall u \in \mathcal{U}_{v}. \tag{55}$$

As shown in Fig. 6, we compare the average uplink data rate of our proposed solution with proportional resource allocation (PRA) and equal resource allocation schemes under the different numbers of users in the subarea. Generally, when the number of users increases, the average data rate of all schemes decreases. This happens because of the limitation of bandwidth, and more users have to share a fixed amount of bandwidth. As we can see in Fig. 6, our resource allocation solution yields a higher data rate compared to other schemes in every circumstance. Additionally, in Fig. 7, we display the average communication latency for various network sizes. It is easy to understand why this happens, the communication latency is calculated by dividing the input size of the task by the achievable data rate, which means communication latency and achievable data rate are inversely proportional. Thus, when the data rate decreases dramatically (shown in Fig. 6), the communication latency grows significantly (shown in Fig. 7). Fig. 7 also compares the average latency under different schemes, and we can see that the optimal resource allocation gives the lowest latency among the three of them. Specifically, the optimal resource scheme is 11.20% better than the PRA approach and 40.80% better than the equal resource allocation approach when the subarea has ten mobile users. These percentages are even more when the network has fewer users. With these results, we can conclude that the proposed method for the resource allocation problem outperforms other schemes.

In Fig. 8, we compare the result of our proposal for the joint offloading decision and resource allocation problem with other benchmarks in total latency. Our approach, as far as we can tell, outperforms any combination of ESA, associated UAVs, and PRA. When the number of subtasks in each task increases, the total latency of all algorithms also increases rapidly. However, their slopes differ: associated schemes have the

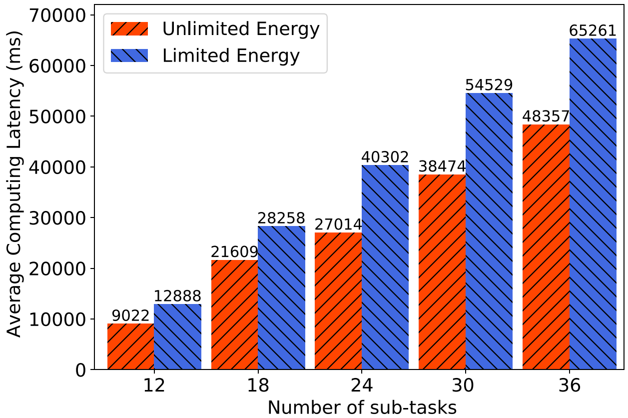

Fig. 9. Average computing latency between limited and unlimited cases under different numbers of subtasks.

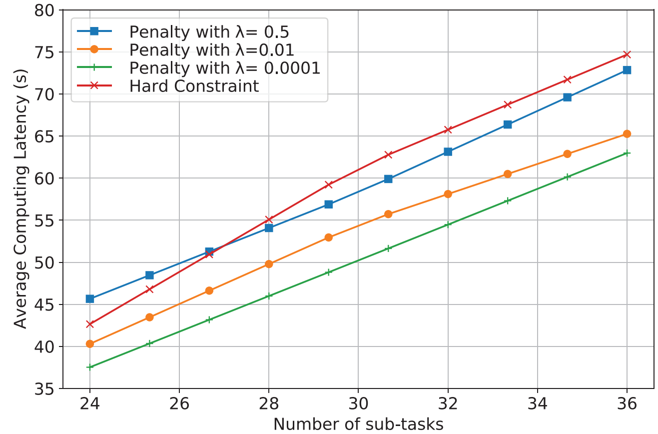

Fig. 10. Average computing latency among different penalty factors of subtasks.

steepest slope, followed by the ESA scheme and our proposed solution. This indicates that our method works effectively in a large dimension space (more subtasks, more offloading variables), while the other schemes perform poorly. For example, when there are 35 subtasks for a single task, the total latency is 68.354 s (WOA+ORA), 80.068 s (ESA+ORA), and 133.644 s (Associate + ORA). These results indicate our proposed approach is 14.63% better than the greedy approach and 48.85% better than the associate scheme. The explanation for the improvement is that the associated UAV scheme executes the tasks alone. There are cases when two subtasks can be executed in parallel, but the associated UAV only executes one of them or divides the computing resources to execute both of them at the same time. In either case, computing latency increases. For the ESA case, the collaboration among UAVs lowers the latency, but it is extremely hard to find the best answer among all feasible solutions without an appropriate search strategy. The integration of different searching mechanisms in D-WOA has proved its effectiveness by providing the lowest latency in the simulation.

In Fig. 9, we compare the average computing latency of D-WOA in two cases: unlimited and limited energy. From the latency result, we can easily see the effect of energy constraint here: the unlimited energy case achieves better latency than the limited case. The gap between the two schemes gets wider with the increased number of subtasks. When the energy of UAVs is limited, they have to lower their computing resources to execute the task, as (26) leads to reducing the computing energy. With the lower computing resource and the inversely proportional relationship between computing latency and computing resources in (23), the execution time of the task increases, which is obviously appropriate.

In Fig. 10, we do experiments to show the impact of the penalty factor on computing latency. Various values of impact factors have been evaluated. However, we only show some worth noticed values in the figure. Before going into detail, we need to explain the hard constraint case: if the offloading decision violates the energy constraint, we omit the solution by assigning a large number to the objective function. As we observe from Fig. 10, the penalty technique offers a better solution than the hard constraint scheme. The penalty factor λ is small, which implies that the searching agents are encouraged to explore more promising regions to find the best solution, even if it violates the constraint. We can get a lower computing latency when we decrease the penalty factor from 0.5 to 0.01. However, as the phenomenon that the smaller the better is not always right; the scheme with the penalty factor $\lambda = 0.0001$ returns an infeasible solution, meaning that the objective function is not penalized enough. To ensure the algorithm returns a feasible solution, the penalty factor has to stay in the range [0.01–0.5].

VI. Conclusion

This article has studied the topology of tasks in the real world. To be more specific, we observed how the dependency affects the start time of successor subtasks and the finish time of the task. We then considered the scheme where dependent tasks from mobile users are offloaded to collaborative MEC-enabled UAVs networks. Then, we formulated an optimization problem with the goal of minimizing the average latency experienced by the mobile users through optimizing the offloading decision for each subtask of users' tasks and communication resource allocation. The designed problem was mixed-integer, nonlinear, and nonconvex when considering it as a whole. Therefore, to solve the problem in polynomial time and make it easier, we decomposed the formulated problem into two subproblems: 1) the offloading decision problem and 2) the communication resource allocation problem. Then, we proposed an appropriate heuristic solution, D-WOA, to acquire the optimal offloading matrix for offloading decision problem. Next, we used SCS solver in the library CVXPY to solve the communication resource allocation problem. Finally, we conducted extensive simulations to demonstrate the superior performance of our algorithm compared to existing benchmark schemes in terms of the total delay encountered by users. Our proposed scheme achieved the lowest latency and outperformed other benchmark schemes in the literature.

References

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.

[2] M. A. Marotta, L. R. Faganello, M. A. K. Schimuneck, L. Z. Granville, J. Rochol, and C. B. Both, “Managing mobile cloud computing considering objective and subjective perspectives,” *Comput. Netw.*, vol. 93, no. 3, pp. 531–542, Dec. 2015.

[3] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, “A survey of mobile cloud computing: Architecture, applications, and approaches,” *Wireless Commun. Mobile Comput.*, vol. 13, no. 18, pp. 1587–1611, Dec. 2013.

[4] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, “Mobile edge computing: A survey,” *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.

[5] W. Yu et al., “A survey on the edge computing for the Internet of Things,” *IEEE Access*, vol. 6, pp. 6900–6919, Nov. 2017.

[6] J. Zhang et al., “Stochastic computation offloading and trajectory scheduling for UAV-assisted mobile edge computing,” *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3688–3699, Apr. 2019.

[7] F. Zhou, Y. Wu, H. Sun, and Z. Chu, “UAV-enabled mobile edge computing: Offloading optimization and trajectory design,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.

[8] N. Cheng et al., “Space/aerial-assisted computing offloading for IoT applications: A learning-based approach,” *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1117–1129, May 2019.

[9] T. Zhang, Y. Xu, J. Loo, D. Yang, and L. Xiao, “Joint computation and communication design for UAV-assisted mobile edge computing in IoT,” *IEEE Trans. Ind. Informat.*, vol. 16, no. 8, pp. 5505–5516, Aug. 2020.

[10] N. Cheng et al., “Air-ground integrated mobile edge networks: Architecture, challenges, and opportunities,” *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 26–32, Aug. 2018.

[11] C. You, K. Huang, H. Chae, and B.-H. Kim, “Energy-efficient resource allocation for mobile-edge computation offloading,” *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[12] Y. K. Tun, M. Alsenwi, S. R. Pandey, C. W. Zaw, and C. S. Hong, “Energy efficient multi-tenant resource slicing in virtualized multi-access edge computing,” in *Proc. Asia-Pacific Netw. Operat. Manage. Symp. (APNOMS)*, Matsue, Japan, Sep. 2019, pp. 1–4.

[13] F. Cheng et al., “UAV trajectory optimization for data offloading at the edge of multiple cells,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 6732–6736, Jul. 2018.

[14] Y. Han, Z. Zhao, J. Mo, C. Shu, and G. Min, “Efficient task offloading with dependency guarantees in ultra-dense edge networks,” in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Waikoloa Village, HI, USA, Feb. 2019, pp. 1–6.

[15] Y. Fan, L. Zhai, and H. Wang, “Cost-efficient dependent task offloading for multiusers,” *IEEE Access*, vol. 7, pp. 115843–115856, Aug. 2019.

[16] S. Z. Li and A. K. Jain, *Handbook of Face Recognition*. London, U.K.: Springer, 2011.

[17] L. Yang, H. Zhang, M. Li, J. Guo, and H. Ji, “Mobile edge computing empowered energy efficient task offloading in 5G,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 6398–6409, Jul. 2018.

[18] H. Sun, F. Zhou, and R. Q. Hu, “Joint offloading and computation energy efficiency maximization in a mobile edge computing system,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 3052–3056, Mar. 2019.

[19] H. Li, H. Xu, C. Zhou, X. Lü, and Z. Han, “Joint optimization strategy of computation offloading and resource allocation in multi-access edge computing environment,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 10214–10226, Sep. 2020.

[20] X. Xia et al., “Data, user and power allocations for caching in multi-access edge computing,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 5, pp. 1144–1155, May 2022.

[21] Y. Liu, Q. He, D. Zheng, X. Xia, F. Chen, and B. Zhang, “Data caching optimization in the edge computing environment,” *IEEE Trans. Services Comput.*, vol. 15, no. 4, pp. 2074–2085, Jul./Aug. 2022.

[22] B. Zhu, K. Chi, J. Liu, K. Yu, and S. Mumtaz, “Efficient offloading for minimizing task computation delay of NOMA-based multiaccess edge computing,” *IEEE Trans. Commun.*, vol. 70, no. 5, pp. 3186–3203, May 2022.

[23] W. Fan, J. Liu, M. Hua, F. Wu, and Y. Liu, “Joint task offloading and resource allocation for multi-access edge computing assisted by parked and moving vehicles,” *IEEE Trans. Veh. Technol.*, vol. 71, no. 5, pp. 5314–5330, May 2022.

[24] Y. K. Tun, Y. M. Park, N. H. Tran, W. Saad, S. R. Pandey, and C. S. Hong, “Energy-efficient resource management in UAV-assisted mobile edge computing,” *IEEE Commun. Lett.*, vol. 25, no. 1, pp. 249–253, Jan. 2021.

[25] Z. Yu, Y. Gong, S. Gong, and Y. Guo, “Joint task offloading and resource allocation in UAV-enabled mobile edge computing,” *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3147–3159, Apr. 2020.

[26] K. Zhang, X. Gui, D. Ren, and D. Li, “Energy–Latency tradeoff for computation offloading in UAV-assisted multiaccess edge computing system,” *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6709–6719, Apr. 2021.

[27] Y. K. Tun, T. N. Dang, K. Kim, M. Alsenwi, W. Saad, and C. S. Hong, “Collaboration in the sky: A distributed framework for task offloading and resource allocation in multi-access edge computing,” *IEEE Internet Things J.*, vol. 9, no. 23, pp. 24221–24235, Dec. 2022.

[28] A. M. Seid, G. O. Boateng, S. Anokye, T. Kwantwi, G. Sun, and G. Liu, “Collaborative computation offloading and resource allocation in multi-UAV-assisted IoT networks: A deep reinforcement learning approach,” *IEEE Internet Things J.*, vol. 8, no. 15, pp. 12203–12218, Aug. 2021.

[29] W. Lu et al., “Resource and trajectory optimization for secure communications in dual unmanned aerial vehicle mobile edge computing systems,” *IEEE Trans. Ind. Informat.*, vol. 18, no. 4, pp. 2704–2713, Apr. 2022.

[30] W. Lu et al., “Secure transmission for multi-UAV-assisted mobile edge computing based on reinforcement learning,” *IEEE Trans. Netw. Sci. Eng.*, early access, Jun. 22, 2022, doi: 10.1109/TNSE.2022.3185130.

[31] X. Pang, N. Zhao, J. Tang, C. Wu, D. Niyato, and K.-K. Wong, “IRS-assisted secure UAV transmission via joint trajectory and beamforming design,” *IEEE Trans. Commun.*, vol. 70, no. 2, pp. 1140–1152, Feb. 2022.

[32] S. Sundar and B. Liang, “Offloading dependent tasks with communication delay and deadline constraint,” in *Proc. IEEE INFOCOM IEEE Conf. Comput. Commun.*, Honolulu, HI, USA, Apr. 2018, pp. 37–45.

[33] C. Shu, Z. Zhao, Y. Han, G. Min, and H. Duan, “Multi-user offloading for edge computing networks: A dependency-aware and latency-optimal approach,” *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1678–1689, Mar. 2020.

[34] Y. Liu et al., “Dependency-aware task scheduling in vehicular edge computing,” *IEEE Internet Things J.*, vol. 7, no. 6, pp. 4961–4971, Jun. 2020.

[35] G. Zhao, H. Xu, Y. Zhao, C. Qiao, and L. Huang, “Offloading tasks with dependency and service caching in mobile edge computing,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 11, pp. 2777–2792, Nov. 2021.

[36] Y. Sahni, J. Cao, L. Yang, and Y. Ji, “Multihop offloading of multiple DAG tasks in collaborative edge computing,” *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4893–4905, Mar. 2021.

[37] J. Wang, J. Hu, G. Min, W. Zhan, A. Y. Zomaya, and N. Georgalas, “Dependent task offloading for edge computing based on deep reinforcement learning,” *IEEE Trans. Comput.*, vol. 71, no. 10, pp. 2449–2461, Oct. 2022.

[38] Y. Zhou, F. R. Yu, J. Chen, and Y. Kuo, “Resource allocation for information-centric virtualized heterogeneous networks with in-network caching and mobile edge computing,” *IEEE Trans. Veh. Technol.*, vol. 66, no. 12, pp. 11339–11351, Dec. 2017.

[39] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, “Efficient deployment of multiple unmanned aerial vehicles for optimal wireless coverage,” *IEEE Commun. Lett.*, vol. 20, no. 8, pp. 1647–1650, Aug. 2016.

[40] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, “Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs,” *IEEE Trans. Wireless Commun.*, vol. 15, no. 6, pp. 3949–3963, Jun. 2016.

[41] Y. K. Tun, N. H. Tran, D. T. Ngo, S. R. Pandey, Z. Han, and C. S. Hong, “Wireless network slicing: Generalized Kelly mechanism-based resource allocation,” *IEEE J. Sel. Areas Commun.*, vol. 37, no. 8, pp. 1794–1807, Aug. 2019.

[42] U. Challita and W. Saad, “Network formation in the sky: Unmanned aerial vehicles for multi-hop wireless backhauling,” in *Proc. IEEE Global Commun. Conf.*, Singapore, Dec. 2017, pp. 1–6.

[43] C.-C. Lai, L.-C. Wang, and Z. Han, “Data-driven 3D placement of UAV base stations for arbitrarily distributed crowds,” in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Waikoloa Village, HI, USA, Feb. 2019, pp. 1–6.

[44] M. Monwar, O. Semiari, and W. Saad, “Optimized path planning for inspection by unmanned aerial vehicles swarm with energy constraints,” in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Abu Dhabi, United Arab Emirates, Feb. 2018, pp. 1–6.

[45] S. Mirjalili and A. Lewis, “The whale optimization algorithm,” *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0965997816300163

[46] Y. Li, Y. He, X. Liu, X. Guo, and Z. Li, “A novel discrete whale optimization algorithm for solving knapsack problems,” *Appl. Intell.*, vol. 50, no. 10, pp. 3350–3366, Jun. 2020.

[47] Q.-V. Pham, S. Mirjalili, N. Kumar, M. Alazab, and W.-J. Hwang, “Whale optimization algorithm with applications to resource allocation in wireless networks,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4285–4297, Apr. 2020.

[48] P. S. Aung, Y. M. Park, Y. K. Tun, Z. Han, and C. S. Hong, “Energy-efficient communication networks via multiple aerial reconfigurable intelligent surfaces: DRL and optimization approach,” Jul. 2022, *arXiv:2207.03149*.

Loc X. Nguyen received the B.Eng. degree in control engineering and automation from the Ho Chi Minh City of Technology, Ho Chi Minh City, Vietnam, in 2019. He is currently pursuing the Ph.D. degree in computer science with Kyung Hee University, Seoul, South Korea.

His research interests include task offloading, network resource optimization, semantic communication, federated learning, and beyond 5G.

Yan Kyaw Tun (Member, IEEE) received the B.E. degree in marine electrical systems and electronics engineering from Myanmar Maritime University, Thanlyin, Myanmar, in 2014, and the Ph.D. degree in computer science and engineering from Kyung Hee University, Seoul, South Korea, in 2021.

He worked as a Postdoctoral Fellow with the Intelligent Networking Laboratory. He is currently working as a Postdoctoral Fellow with the School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Stockholm, Sweden, and he is also with Kyung Hee University, Yongin, South Korea. His research interests include network economics, game theory, network optimization, wireless communication, wireless network virtualization, mobile-edge computing, and wireless resource slicing for 5G.

Dr. Tun received the Best Ph.D. Thesis Award in Engineering in 2020.

Tri Nguyen Dang received the B.S. degree in information teaching from Hue University’s College of Education, Hue, Vietnam in 2014. He is currently pursuing the Ph.D. degree in computer science and engineering under fully funded scholarship with Kyung Hee University, Seoul, South Korea.

His professional experience includes Mobile Application Developer, middleware programming. His research interest includes network optimization, and mobile cloud computing, multiaccess edge computing, and Internet of Things.

Mr. Dang had won the Consolation Prize of Olympiad in ACM ICPC Vietnam in 2013.

Yu Min Park received the B.S. degree in applied mathematics and computer engineering and the M.S. degree in computer engineering from Kyung Hee University, Seoul, South Korea, in 2019 and 2021, respectively, where he is currently pursuing the Ph.D. degree in computer engineering.

His research interests include reinforcement learning, intelligent networking management system, and network resource optimization.

Zhu Han (Fellow, IEEE) received the B.S. degree in electronic engineering from Tsinghua University, Beijing, China, in 1997, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Maryland, College Park, MD, USA, in 1999 and 2003, respectively.

He was an Research and Development Engineer with JDSU, Germantown, MD, USA, from 2000 to 2002. He was a Research Associate with the University of Maryland from 2003 to 2006. He was an Assistant Professor with Boise State University, Boise, ID, USA, from 2006 to 2008. He is currently a John and Rebecca Moores Professor with the Electrical and Computer Engineering Department as well as the Computer Science Department, University of Houston, Houston, TX, USA, and he is also with Kyung Hee University, Yongin, South Korea. His main research targets on the novel game-theory-related concepts critical to enabling efficient and distributive use of wireless networks with limited resources. His other research interests include wireless resource allocation and management, wireless communications and networking, quantum computing, data science, smart grid, and security and privacy.

Dr. Han received the NSF Career Award in 2010, the Fred W. Ellersick Prize of the IEEE Communication Society in 2011, the EURASIP Best Paper Award for the *Journal on Advances in Signal Processing* in 2015, IEEE Leonard G. Abraham Prize in the field of Communications Systems (Best Paper Award in IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS) in 2016, and several best paper awards in IEEE conferences. He is also the Winner of the 2021 IEEE Kiyo Tomiyasu Award, for outstanding early to mid-career contributions to technologies holding the promise of innovative applications, with the following citation: “for contributions to game theory and distributed management of autonomous communication networks.” He has been a 1% Highly Cited Researcher since 2017 according to the Web of Science. He was an IEEE Communications Society Distinguished Lecturer from 2015 to 2018, and has been an AAAS Fellow since 2019 and an ACM Distinguished Member since 2019.

Choong Seon Hong (Senior Member, IEEE) received the B.S. and M.S. degrees in electronic engineering from Kyung Hee University, Seoul, South Korea, in 1983 and 1985, respectively, and the Ph.D. degree from Keio University, Tokyo, Japan, in 1997.

He joined KT, Seongnam, Gyeonggi, South Korea, in 1988, where he was involved in broadband networks as a member of the Technical Staff. He has been with Keio University since 1993. He was with the Telecommunications Network Laboratory, KT, as a Senior Member of Technical Staff and as the Director of the Networking Research Team until 1999. He has been a Professor with the Department of Computer Science and Engineering, Kyung Hee University since 1999. His research interests include future Internet, intelligent edge computing, network management, and network security.

Dr. Hong is a member of the Association for Computing Machinery, Institute of Electronics, Information and Communication Engineers, Information Processing Society of Japan, Korean Institute of Information Scientists and Engineers, Korean Institute of Communications and Information Sciences, Korean Information Processing Society, and Open Standards and ICT Association. He has served as the General Chair, the TPC Chair/Member, or an Organizing Committee Member for international conferences, such as the Network Operations and Management Symposium (NOMS), International Symposium on Integrated Network Management, Asia–Pacific NOMS, End-to-End Monitoring Techniques and Services, IEEE Consumer Communications and Networking Conference, Assurance in Distributed Systems and Networks, International Conference on Parallel Processing, Data Integration and Mining, World Conference on Information Security Applications, Broadband Convergence Network, Telecommunication Information Networking Architecture, International Symposium on Applications and the Internet, and International Conference on Information Networking. He was an Associate Editor of the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT and IEEE JOURNAL OF COMMUNICATIONS AND NETWORKS. He was also a Guest Editor of the *IEEE Network Magazine*. He currently serves as an Associate Editor for the *International Journal of Network Management* and *Future Internet*.