3-2020

Energy-Efficient UAV-Assisted Mobile Edge Computing: Resource Allocation and Trajectory Optimization

Mushu Li
University of Waterloo

Nan Cheng
Xidian University

Jie Gao
Marquette University, jie.gao@marquette.edu

Yinlu Wang
Southeast University

Lian Zhao
Ryerson University

See next page for additional authors

IEEE Transactions on Vehicular Technology, Vol. 69, No. 3 (March 2020): 3424-3438. DOI.

Energy-Efficient UAV-Assisted Mobile Edge Computing: Resource Allocation and Trajectory Optimization

Mushu Li
Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada
Nan Cheng
State Key Laboratory of Information Security, School of Telecommunications Engineering, Xidian University, Xian, China
Jie Gao
Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada
Yinlu Wang
National Mobile Communications Research Laboratory, Southeast University, Nanjing, China
Lian Zhao
Department of Electrical, Computer and Biomedical Engineering, Ryerson University, Toronto, Canada
Xuemin Shen

Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada

Abstract:

In this paper, we study unmanned aerial vehicle (UAV) assisted mobile edge computing (MEC) with the objective to optimize computation offloading with minimum UAV energy consumption. In the considered scenario, a UAV plays the role of an aerial cloudlet to collect and process the computation tasks offloaded by ground users. Given the service requirements of users, we aim to maximize UAV energy efficiency by jointly optimizing the UAV trajectory, the user transmit power, and computation load allocation. The resulting optimization problem corresponds to nonconvex fractional programming, and the Dinkelbach algorithm and the successive convex approximation (SCA) technique are adopted to solve it. Furthermore, we decompose the problem into multiple subproblems for distributed and parallel problem solving. To cope with the case when the knowledge of user mobility is limited, we adopt a spatial distribution estimation technique to predict the location of ground users so that the proposed approach can still be applied. Simulation results demonstrate the effectiveness of the proposed approach for maximizing the energy efficiency of UAV.

SECTION I. Introduction

Driven by the visions of Internet of Things (IoT) and 5G communications, mobile edge computing (MEC) is considered as an emerging paradigm that leverages the computing resource and storage space deployed at network edges to perform latency-critical and computation-intensive tasks for mobile users [1]. The computation tasks generated by mobile users can be offloaded to the nearby edge server, such as macro/small cell base station and Wi-Fi access point, to reduce computation delay and computing energy cost at mobile devices. Moreover, by pushing the traffic, computation, and network functions to the network edges, mobile users can enjoy low task offloading time with less backhaul usage [2].

Specifically, in IoT era, MEC is considered as a key enabling technology to support the computing services for billions of IoT nodes to be deployed [3], [4]. Since the most of IoT nodes are power-constrained and have limited computing compatibility, they can offload their computation tasks to network edges to extend their battery life and improve the computing efficiency. However, many IoT nodes are operating in unattended or challenging areas, such as forests, deserts, mountains, or underwater locations [5], to execute some computation-intensive applications, including long pipeline infrastructures monitoring and control [6], underwater infrastructures monitoring [7], and military operations [8]. In these scenarios, the terrestrial communication infrastructures are distributed sparsely and cannot provide reliable communications for the nodes. Therefore, in this paper, we utilize unmanned aerial vehicles (UAVs) to provide ubiquitous communication and computing supports for IoT nodes. Equipped with computing resources, UAV-mounted cloudlet can collect and process the computation tasks of ground IoT nodes that cannot connect to the terrestrial edges. As UAVs are fully controllable and operate at a high altitude, they can be dispatched to the designated places for providing efficient on-demand communication and computing services to IoT nodes in a rapid and flexible manner [9]–[10][11][12].

Despite the advantages of UAV-assisted MEC, there are several challenges in network deployment and operation. Firstly, the onboard energy of a UAV is usually limited. To improve the user experience on the computing service, UAVs should maximize their energy efficiency by optimizing their computing ability in the limited service time. Secondly, planning an energy-aware UAV trajectory is another challenge in UAV-assisted networks. The UAV is required to move to collect the offloaded data from sparsely distributed users for the best channel quality, while a significant portion of UAV energy consumption stems from mechanical actions during flying. Thirdly, the computation load allocation cannot be neglected even though the computing energy consumption in UAV-mounted cloudlet is relatively small compared to its mechanical energy. In the state-of-art MEC server architecture, the dynamic frequency and voltage scaling (DVFS) technique is adopted. The

computing energy for a unit time is growing cubically as the allocated computation load increases [1]. Without proper allocation, the computing energy consumption could blow up, or the offloaded tasks cannot be finished in time. More importantly, UAV trajectory design, computation load allocation, and communication resource management are coupled in the MEC system [13], which makes the system even more complex. To the best of our knowledge, the joint optimization of UAV trajectory, computation load allocation, and communication resource management considering energy efficiency has not been investigated in the UAV-assisted MEC system.

To address the above challenges, we consider an energy constrained UAV-assisted MEC system in this paper. IoT nodes as ground users can access and partially offload their computation tasks to the UAV-mounted cloudlet according to their service requirements. The UAV flies according to a designed trajectory to collect the offloading data, process computation tasks, and send computing results back to the nodes. For each data collection and task execution cycle, we optimize the energy efficiency of the UAV, which is defined as the ratio of the overall offloaded computing data to UAV energy consumption in the cycle, by jointly optimizing the UAV trajectory and resource allocation in communication and computing aspects. The main contributions of the paper are summarized as follows.

1. We develop a model for energy-efficient UAV trajectory design and resource allocation in the MEC system. The model incorporates computing service improvement and energy consumption minimization in a UAV-mounted cloudlet. The communication and computing resources are allocated subject to the user communication energy budget, computation capability, and the mechanical operation constraints of the UAV.
2. We exploit the successive convex approximation (SCA) technique and Dinkelbach algorithm to transform the non-convex fractional programming problem into a solvable form. In order to improve scalability, we further decompose the optimization problem by the alternating direction method of multipliers (ADMM) technique. UAV and ground users solve the optimization problem cooperatively in a distributed manner. By our approach, both users and UAV can obtain the optimal resource allocation results iteratively without sharing local information.
3. We further consider the scenario with limited knowledge of node mobility. A spatial distribution estimation technique, Gaussian kernel density estimation, is applied to predict the location of ground users. Based on the predicted location information, our proposed strategy can determine an energy-efficient UAV trajectory when the user mobility and offloading requests are ambiguous at the beginning of each optimization cycle.

The remainder of the paper is organized as follows. Related works are discussed in Section II. The system model is provided in Section III. Problem formulation and the corresponding approach are presented in Section IV and V, respectively. The extended implementation of the proposed approach are provided in Section VI. Finally, extensive simulation results and conclusions are provided in Sections VII and VIII, respectively.

SECTION II. Related Works

A. Mobile Edge Computing

To improve the user experience on mobile computing in 5G era, the concept of MEC has been proposed in [14] to reduce the transmitting and computing latency by utilizing a vast amount of computation resource located at edge devices. The works [15], [16] consider energy-efficient computing in MEC. In [15], Zhang *et al.* study the total energy consumption minimization in 5G heterogeneous networks. The mobile users make binary offloading decisions to determine where their computation tasks are executed. In [16],

Mao *et al.* investigate the MEC system with energy harvesting devices and propose an online Lyapunov-based method to reduce the computing latency and the probability of task dropping. The works [17]–[18][19] study radio resource allocation for computation offloading in edge computing. In [17], Kuang *et al.* propose a partial offloading scheduling and power allocation approach for single user MEC system and jointly minimize the task execution delay and energy consumption in MEC server while guaranteeing the transmit power constraint of the user. In [18], [19], Rodrigues *et al.* investigate transmit power control and service migration policy to balance the computation load among edge servers and reduce the overall computing delay accordingly. The above works consider resource allocation in MEC with fixed edge infrastructures. To provide on-demand service for remote IoTs, our work studies edge computing supported by UAV-mounted cloudlet, which introduces dynamic channel conditions and mechanical operation constraints.

B. UAV-Assisted Network

The UAV-assisted communication network has been investigated in works [20]–[21][22]. In [20], Wu *et al.* consider trajectory design and communication power control for a multi-UAV multi-user system, in which the objective is to maximize the throughput over ground users in a downlink scenario. In [21], Zeng *et al.* analyze the energy efficiency of the UAV-assisted communication network and design a UAV trajectory strategy for hovering above a single ground communication terminal. In [22], Tang *et al.* investigate a game-based channel assignment scheme for UAVs in D2D-enabled communication networks. UAVs have also been utilized to enhance the flexibility of a MEC system in [23], [24], where UAVs behave as communication relays to participate in the computation offloading process. Moreover, recently, more works utilize UAV as an aerial cloudlet to provide edge computing service [25]–[26][27]. In [25], Jeong *et al.* study UAV path planning to minimize communication energy consumption for task offloading at mobile users, where the energy consumption of UAV-mounted cloudlet is constrained. Both orthogonal and non-orthogonal channel models are considered in the work. In [26], Tang *et al.* propose a UAV-assisted recommendation system in location based social networks (LBSNs), while a UAV-mounted cloudlet is deployed to reduce computing and traffic load of the cloud server. In [27], Cheng *et al.* provide the computation load offloading strategy in an IoT network given the pre-determined UAV trajectories. The work aims to minimize the computing delay, user energy consumption, and server computing cost jointly, where the energy consumption of the UAV-mounted cloudlet has not been investigated. None of the above works discusses the energy efficiency on mobile computing in a UAV-mounted cloudlet, which is considered as a meaningful metric for prolonging the computing service lifetime. Note that although [21] also studies energy-efficient trajectory design, it focuses on a single-ground-terminal scenario, whereas our work focuses on a multi-user scenario with corresponding resource management.

SECTION III. System Model

A. Network Model

The UAV-assisted MEC system is shown in Fig. 1, in which a single UAV-mounted cloudlet is deployed to offer edge computing service for ground users in area $\mathcal{A}$. The UAV periodically collects and processes the computation tasks offloaded from ground users. Each user processes the rest of the computation tasks locally if the task cannot be fully collected by the UAV. Define the computing cycle as a duration of T seconds. Each cycle contains K discrete time slots with equal length. Denote the set of time slots in the cycle by K. Thus, the time length for a slot is T/K, which is denoted by Δ. The list of symbols is given in Table I.

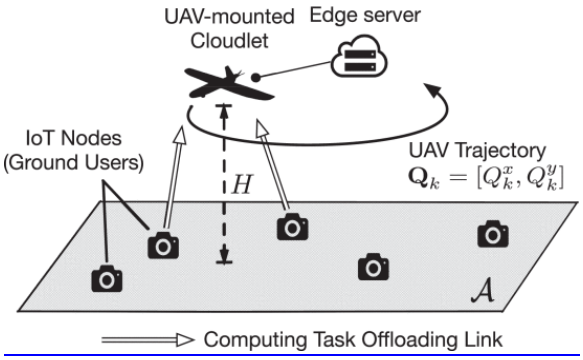

Fig. 1. System model.

TABLE I List of Symbols

k	index of time slot
i	index of ground node/user
$\mathcal{I}$	set of users, where $\mathcal{I} = \{1, \ldots, N\}$
K	set of time slots, where $\mathcal{K} = \{1, \ldots, K\}$
$\mathbf{a}_k(\mathbf{Q})$	average acceleration of the UAV in slot k
$a_{\max}$	maximum acceleration of the UAV
B	channel bandwidth
E_i^T	maximum offloading communication energy of user i
$E_{i,k}^{C,U}(\mathbf{W}_k)$	UAV computing energy for executing tasks from user i in time slot k
$E_k^F(\mathbf{Q})$	UAV propulsion energy consumption in slot k
E_i^M	maximum computing energy consumption of user i
$f_k^U(\mathbf{W}_k)$	CPU-cycle frequency in time slot k
f_i^M	CPU-cycle frequency of user i
$h_{i,k}(\mathbf{Q}_k)$	channel gain for user i in slot k
h_x, h_y	bandwidth of the 2-D Gaussian kernel
H	UAV flying altitude
I_i	overall input data size for computation tasks of user i
$\check{I}_i$	minimum input data amount to be offloaded for user i
K	number of time slots in a time window
N	number of users
P	maximum transmit power of a user
$\mathbf{q}_{i,k}$	horizontal coordinate of user i in slot k
$\mathbf{Q}_k$	horizontal coordinate of the UAV in slot k
$R_{i,k}(\delta_{i,k}, \mathbf{Q}_k)$	data rate for user i in slot k
$S_{i,k}(\delta_{i,k})$	communication energy for user i in slot k
T	time length of a computing cycle
$\mathbf{v}_k(\mathbf{Q})$	average velocity of the UAV in slot k
$v_{\max}$	maximum velocity of the UAV
$W_{i,k}$	amount of data offloaded by i to be processed in slot k at the UAV-mounted cloudlet
γ_1, γ_2	UAV propulsion energy consumption parameters
$\delta_{i,k}$	portion of the maximum power allocated to user i within slot k
Δ	time length of a time slot
σ^2	power spectral density of channel noise
χ_i	number of computation cycles for executing 1 bit

At the beginning of each cycle, ground users with computation tasks in area $\mathcal{A}$ send offloading requests to the UAV-mounted cloudlet. Denote the set of those ground users by $\mathcal{I}$, where $\mathcal{I} = \{1, \dots, N\}$. Assume the ground users in $\mathcal{I}$ can connect to the UAV for all time slots in the cycle. In this work, the UAV and the users cooperatively determine the offloading and resource allocation strategy for this cycle, including the UAV moving trajectory, the transmit power of ground users, and computation load allocation for UAV-mounted cloudlet. Assume that the computation loads on solving the optimization problem are negligible compared to the computation loads of the offloaded tasks. During the cycle, UAV flies over the ground users and offers the computing service according to the designed trajectory and resource allocation strategy. By the end of the cycle, UAV returns to a predetermined final position.

B. Communication Model

The quality of communication links between the UAV and ground users is dependent on their location. To represent their locations, we construct a 3D Cartesian coordinate system. For IoT node i, the horizontal coordinate at time k is denoted by $\mathbf{q}_{i,k} = [q_{i,k}^x, q_{i,k}^y]$. Assume that nodes know their trajectory for the upcoming cycle, *i.e.*, $\{\mathbf{q}_{i,k}, \forall k\}$. For the UAV, the horizontal coordinate at time k is denoted by $\mathbf{Q}_k = [Q_k^x, Q_k^y]$. The UAV moves at a fixed altitude H. The UAV trajectory plan, as an optimization variable, consists of UAV positions in the whole cycle, *i.e.*, $\mathbf{Q} = [\mathbf{Q}_1; \dots; \mathbf{Q}_K]$. The average UAV velocity in slot k is given by

$$\mathbf{v}_k(\mathbf{Q}) = \frac{\mathbf{Q}_k - \mathbf{Q}_{k-1}}{\Delta}, \forall k.$$

(1)

The average acceleration in slot k is given by

$$\mathbf{a}_k(\mathbf{Q}) = \frac{\mathbf{v}_k(\mathbf{Q}) - \mathbf{v}_{k-1}(\mathbf{Q})}{\Delta}, \forall k.$$

(2)

The magnitudes of velocity and acceleration are constrained by the maximum speed and acceleration magnitude, which are denoted by $v_{\max}$ and $a_{\max}$, respectively.

It is assumed that the doppler frequency shift in the communication can be compensated at the receiver. The channel quality depends on the distance between the UAV and users. Due to the high probability of LOS links in UAV communication [21], we assume that the channel gain follows a free-space path loss model. The channel gain for user i in slot k is denoted by $h_{i,k}$, where

$$h_{i,k}(\mathbf{Q}_k) = \frac{g_0}{\|\mathbf{Q}_k - \mathbf{q}_{i,k}\|_2^2 + H^2},$$

(3)

where $\|\cdot\|_2$ is the notation representing the L2 norm. The parameter g_0 denotes the received power at the reference distance (e.g., $d = 1$ m) between the transmitter and the receiver. We consider two channel access schemes: i) orthogonal access, in which the bandwidth is divided into N sub-channels each occupied by one user; and ii) non-orthogonal access, in which the frequency bandwidth is shared among users. Denote the channel bandwidth for the uplink by B. The amount of data that can be offloaded by user i in slot k is

$$R_{i,k}\left(\delta_{i,k}, \mathbf{Q}_k\right) = \frac{B\Delta}{N}\log\left[1 + \frac{\delta_{i,k}h_{i,k}(\mathbf{Q}_k)P}{\sigma^2(B/N)}\right],$$

(4)

under the orthogonal access model, and,

$$R_{i,k}(\boldsymbol{\delta}_k, \mathbf{Q}_k) = B\Delta\log\left[1 + \frac{\delta_{i,k}h_{i,k}(\mathbf{Q}_k)P}{\sigma^2 B + \sum_{j\neq i}\delta_{j,k}h_{j,k}\,(\mathbf{Q}_k)P}\right],$$

(5)

under the non-orthogonal channel model. The parameter P and σ^2 denote the maximum transmit power of ground users and the power spectral density of channel noise, respectively. The variable $\delta_{i,k} \in [0,1]$ represents the portion of the maximum power that is allocated to user i within time slot k, which is a part of the offloading strategy. The symbol $\boldsymbol{\delta}_k$ denotes the vector of $\delta_{i,k}$ for all $i \in \mathcal{I}$ in slot k. The noise power in the transmission is represented by n_0, where $n_0 = \sigma^2 B/N$ for the orthogonal channel access model, and $n_0 = \sigma^2 B$ for the non-orthogonal channel access model. In non-orthogonal model, users share the same channel to offload their tasks. The communication power allocated for a user will interfere the data rate of other users.

C. Computation Model

Due to the limited battery and the computing capability of the UAV, only a part of tasks can be offloaded and executed in the UAV-mounted cloudlet. Full granularity in task partition is considered, where the task-input data can be arbitrarily divided for local and remote executions [25], [28], [29]. Accordingly, a portion of the computation tasks are offloaded to the cloudlet while the rest are executed by the ground users locally. Users upload the input data for their offloaded tasks, and the UAV processes the corresponding computation loads of those tasks. Assume that the computation load can be executed once the input data is received, and the computing data amount is equal to the input data amount of tasks [25]. A task partition technique is considered, where the partition of the computation input bits are utilized to measure the division between the offloaded computation load and local computation load. The overall input data size for computation tasks of user i is denoted by I_i. We set the threshold $\check{I}_i$ as the minimum input data amount required to be offloaded to the cloudlet for user i, where $\check{I}_i \leq I_i$. The threshold represents the part of computation tasks having to be conducted in the cloudlet. Thus, the overall offloaded bits of user i is constrained as follows:

$$\check{I}_i \leq \sum_{k\in\mathcal{K}} R_{i,k}(\boldsymbol{\delta}_k, \mathbf{Q}_k) \leq I_i, \forall i\,.$$

(6)

Under the scenario that the threshold is satisfied, if user's tasks cannot be fully offloaded, the rest of the tasks are processed by IoT nodes locally.

After users upload the input data, the UAV will save the received data to a buffer with enough capacity for further processing. The UAV processes the received data according to the workload allocation results. Let the variable $W_{i,k}$ denote the amount of data, which is from user i's offloaded task, to be processed in slot k. The

UAV can only compute the task which is offloaded and received, and all offloaded tasks should be executed by the end of the cycle. Therefore, the following computation constraints are given:

$$\sum_{t=1}^{k} R_{i,t}(\boldsymbol{\delta}_k, \mathbf{Q}_k) \geq \sum_{t=1}^{k} W_{i,t}, \forall k$$
$$\sum_{t=1}^{K} R_{i,t}(\boldsymbol{\delta}_k, \mathbf{Q}_k) = \sum_{t=1}^{K} W_{i,t}\,.$$

(7a)(7b)

In addition, for the local computing, the CPU-cycle frequency of the IoT node i is fixed as f_i^M. For the UAV-mounted cloudlet, we consider the CPU featured by DVFS technique. The CPU-cycle frequency can step-up or step-down according to the computation workload and is bounded by the maximum CPU-cycle frequency f_{max}^U. As given in [1], [28], the CPU-cycle frequency for the cloudlet can be calculated by

$$f_k^U(\mathbf{W}_k) = \frac{\sum_i \chi_i W_{i,k}}{\Delta} \leq f_{max}^U, \forall k,$$

(8)

where $f_k^U(\mathbf{W}_k)$ represents the CPU-cycle frequency in time slot k, and χ_i denotes the number of computation cycles needed to execute 1 bit of data.

D. Energy Consumption Model

1) Energy Consumption at Nodes

The main energy consumption of nodes are the energy cost from communication and local computing. Firstly, the communication energy for user i offloading tasks in slot k can be formulated as

$$S_{i,k}(\delta_{i,k}) = \delta_{i,k} P \Delta.$$

(9)

The overall offloading communication energy of user i is bounded by E_i^T, *i.e.*,

$$\sum_k S_{i,k}(\delta_{i,k}) \leq E_i^T, \forall i\,.$$

(10)

Therefore, the energy consumption of a user on communication can be reduced if the UAV is closer. On the other hand, for the computing energy consumption, we consider that the lower bound of offloaded bits $\check{I}_i$ guarantees the local computing energy under the user's computing energy requirement, *i.e.*,

$$E_i^M = \kappa \chi_i (I_i - \check{I}_i)(f_i^M)^2 \leq \hat{E}_i^M,$$

(11)

where E_i^M is the maximum computing energy that could be reached by threshold $\check{I}_i$, and $\hat{E}_i^M$ is the parameter representing the constraint of the computing energy consumption. The computing energy model is adopted from [1], [30]. Parameters f_i^M and κ represent the fixed CPU-cycle frequency of user i and a constant related to the hardware architecture, respectively.

2) Energy Consumption at UAV-Mounted Cloudlet

The main energy consumption at the UAV-mounted cloudlet consists of the energy cost from mechanical operation and computing. Although downlink transmission exists in our system, this part of energy consumption is negligible for two reasons: 1) The communication energy is too small compared to the UAV propulsion and computing energy. 2) The output computing results usually have much less data amount compared to the input data amount [31]. We adopt the refined UAV propulsion energy consumption model for fixed-wing UAV following [21].* The propulsion energy consumption in slot k relates to the instantaneous UAV acceleration and velocity, which is given by

$$E_k^F(\mathbf{Q}) = \gamma_1 \|\mathbf{v}_k(\mathbf{Q})\|_2^3 + \frac{\gamma_2}{\|\mathbf{v}_k(\mathbf{Q})\|_2}\left(1 + \frac{\|a_k(\mathbf{Q})\|_2^2}{g^2}\right),$$

(12)

where g denotes the gravitational acceleration. γ_1 and γ_2 are fixed parameters related to the aircraft's weight, wing area, air density, etc. The value of parameters is given in [21], [25]. The computing energy for executing tasks from user i in time slot k is expressed as

$$E_{i,k}^{C,U}(\mathbf{W}_k) = \kappa \chi_i W_{i,k} (f_k^U(\mathbf{W}_k))^2.$$

(13)

SECTION IV. Problem Formulation

In this work, the main objective is to maximize the energy efficiency of the UAV-mounted cloudlet subject to user offloading constraints, UAV computing capabilities, and the mechanical constraints of the UAV. The energy efficiency of the UAV is defined as the ratio between the overall offloaded data and the energy consumption of the UAV in a cycle. The energy efficiency maximization problem is formulated as follows.

$$\begin{aligned}
\max_{\boldsymbol{\delta},\mathbf{W},\mathbf{Q}} \quad & \eta = \frac{\sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{K}} R_{i,k}(\boldsymbol{\delta}_k, \mathbf{Q}_k)}{\sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{I}} E_{i,k}^{C,U}(\mathbf{W}_k) + \sum_{k\in\mathcal{K}} E_k^F(\mathbf{Q})} \\
\text{s.t.} \quad & \|\mathbf{v}_k(\mathbf{Q})\|_2 \le v_{max}, \forall k, \\
& \|\mathbf{a}_k(\mathbf{Q})\|_2 \le a_{max}, \forall k, \\
& \mathbf{Q}_K = \mathbf{Q}_f, \mathbf{v}_K(\mathbf{Q}) = \mathbf{v}_0, \\
& 0 \le \delta_{i,k} \le 1, \\
& (6), (7a), (7b), (8), (10).
\end{aligned}$$

(14)(14a)(14b)(14c)(14d)

The term $\mathbf{Q}_f$ represents the designated final position of the UAV, and $\mathbf{v}_0$ represents the initial velocity at the beginning of the cycle. The constraints can be categorized into three types: 1) user QoS constraints, including (6), (10), and (14d); 2) UAV computing ability constraints, including (7a), (7b), and (8); 3) UAV

mechanical constraints, including (14a), (14b), and (14c). The optimization problem is a non-linear fractional programming. In addition, due to the interference among users in the non-orthogonal channel and the propulsion energy consumption for the fixed-wing UAV, both functions $R_{i,k}(\boldsymbol{\delta}_k, \mathbf{Q}_k)$ and $E_k^F(\mathbf{Q})$ are non-convex. Therefore, solving optimization problem (14) is challenging. To search the global optimizer of a non-convex problem is often slow and may not be feasible. In the following section, we will propose an approach to find a local optima efficiently.

SECTION V. Proposed Optimization Approach

In this section, an optimization approach is introduced to find a solution of problem (14). Firstly, an inner convex approximation method is applied to approximate the non-convex functions $R_{i,k}(\boldsymbol{\delta}_k, \mathbf{Q}_k)$ and $E_k^F(\mathbf{Q})$ by solvable convex functions. The SCA-based algorithm is adopted to achieve the local optimizer of the original problem. After the approximated convex functions are built, the fraction programming in the inner loop of the SCA-based algorithm is handled by the Dinkelbach algorithm. Moreover, in order to improve scalability, the problem is further decomposed into several sub-problems via ADMM technique, in which the power allocation is solved by users in a distributed manner, while the computation load allocation and UAV trajectory planning are determined by UAV itself. The details are presented in following subsections.

A. Successive Convex Approximation

Problem (14) is a non-convex problem due to $R_{i,k}(\boldsymbol{\delta}_k, \mathbf{Q}_k)$ and $E_k^F(\mathbf{Q})$. To construct an approximation that is solvable, we first introduce several auxiliary variables, $\{\xi_{i,k}, \omega_k, l_{i,k}, A_k, \check{R}_{i,k}, \hat{E}_{i,k}^F\}$. For the orthogonal channel access scheme, the new optimization problem is shown as follows:

$$
\begin{aligned}
\max_{\mathcal{V}} \quad & \check{\eta}(\mathcal{V}) = \frac{\sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{K}} \check{R}_{i,k}}{\sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{I}} E_{i,k}^{C,U}(\mathbf{W}_k) + \sum_{k\in\mathcal{K}} \hat{E}_k^F} \\
\text{s.t.} \quad & \check{R}_{i,k} \le \frac{B\Delta}{N}\log(1+\xi_{i,k}), \forall i,k \\
& \xi_{i,k} l_{i,k} \le \delta_{i,k} P, \forall i,k \\
& \frac{(\|\mathbf{Q}_k - \mathbf{q}_{i,k}\|_2^2 + H^2) n_0}{g_0} \le l_{i,k}, \forall i,k \\
& \hat{E}_k^F \ge \gamma_1 \|\mathbf{v}_k(\mathbf{Q})\|_2^3 + \gamma_2 A_k, \forall k \\
& \omega_k^2 \le \|\mathbf{v}_k(\mathbf{Q})\|_2^2, \forall k \\
& \omega_k A_k \ge 1 + \frac{\|a_k(\mathbf{Q})\|_2^2}{g^2}, \forall k \\
& \check{I}_i \le \sum_{k\in\mathcal{K}} \check{R}_{i,k} \le I_i, \forall i, \\
& (6), (7a), (7b), (8), (10), (14a) - (14d).
\end{aligned}
$$

(15)(15a)(15b)(15c)(15d)(15e)(15f)(15g)

Set $\mathcal{V}$ represents the union set of the primary and auxiliary optimization variables, where $\mathcal{V} = \{\boldsymbol{\delta}, \mathbf{W}, \mathbf{Q}, \boldsymbol{\xi}, \boldsymbol{\omega}, \mathbf{l}, \mathbf{A}, \check{\mathbf{R}}, \hat{\mathbf{E}}^F\}$. For the non-orthogonal channel model, constraint (15a) is replaced by the following constraint:

$$\check{R}_{i,k} \leq B\Delta\left[\log\left(1+\sum_{i\in\mathcal{I}}\xi_{i,k}\right)-\log\left(1+\sum_{j\in\frac{\mathcal{I}}{\{i\}}}\xi_{j,k}\right)\right], \quad \forall i,k.$$

(15h)

Lemma 1:

Problem (15) is an equivalent form of problem (14).

Proof:

See Appendix A.■

Problem (15) includes four non-convex constraints, which are (15b), (15e), (15f), and (15h). We approximate those non-convex constraints by their first order Taylor expansions and adopt the successive convex optimization technique to solve the problem. New auxiliary variables, $\{\xi_{i,k}^t, l_{i,k}^t, \omega_k^t, A_k^t, \mathbf{v}_k, z_{i,k}^t\}$, are introduced to represent the corresponding estimated optimizers at the previous iteration of optimization, *i.e.*, iteration t. The SCA-based algorithm iterates until the estimated solution reaches to a local optimizer. Constraint (15b) can be approximated as follows:

$$\|\xi_{i,k}+l_{i,k}, \xi_{i,k}^t - l_{i,k}^t, x_{i,k}-1\|_2 \leq x_{i,k}+1,$$

(16)

where

$$x_{i,k} = \delta_{i,k}P - \frac{(\xi_{i,k}^t - l_{i,k}^t)(\xi_{i,k}-l_{i,k})}{2}.$$

Constraint (15e) can be approximated as follows:

$$\omega_k^2 \leq \|\mathbf{v}_k^t\|_2^2 + 2(\mathbf{v}_k^t)^T(\mathbf{v}_k(\mathbf{Q})-\mathbf{v}_k^t).$$

(17)

Constraint (15f) can be approximated as follows:

$$\|\omega_k - A_k, \omega_k^t + A_k^t, y_k - 1, 2, \frac{2a_k(\mathbf{Q}_k)}{g}\|_2 \leq y_k+1,$$

(18)

where

$$y_k = \frac{(\omega_k^t + A_k^t)(\omega_k + A_k)}{2}.$$

Constraint (15h) can be approximated as follows:

$$\check{R}_{i,k} \leq \frac{B\Delta}{N} \quad [\log\left(1 + \xi_{i,k} + e_{i,k}\right) - \log\left(1 + e_{i,k}^t\right)) - \frac{e_{i,k} - e_{i,k}^t}{\ln 2\left(1 + e_{i,k}^t\right)}],$$

(19)

where $e_{i,k} = \sum_{j \in \mathcal{I}/\{i\}} \quad \xi_{i,k}$.

Lemma 2:

Non-convex constraints (15b), (15e), (15f), and (15h) can be approximated by the convex forms in (16)–(19). The solution of the approximated problem is a local maximizer of problem (14), which provides the lower bound of the maximum energy efficiency that can be achieved.

Proof:

See Appendix B.■

Algorithm 1: SCA-based Algorithm for Solving Problem (15).

1. Initialize the auxiliary variables $\mathbb{A}^0 = \{\xi_{i,k}^0, \omega_k^0, l_{i,k}^0, A_k^0, \check{R}_{i,k}^0, \hat{E}_{i,k}^{F,0}\}$ and loop index $t = 0$.
2. **repeat**
3. Solve the approximated problem (20) for given $\mathbb{A}^t$, and denote the optimal solution for auxiliary variables by $\mathbb{A}^{t+1}$:

$$\begin{aligned} \max_{\mathcal{V}} \quad & \check{\eta}(\mathcal{V}; \mathbb{A}^t) \\ \text{s.t.} \quad & (6), (7a), (7b), (8), (10), (14a) - (14d), \\ & (15c), (15d), (15g), (16) - (17), \\ & (15a), \text{in the case of orthogonal channel}, \\ & (19), \text{in the case of non-orthogonal channel}. \end{aligned} \tag{20}$$

4. Update $t = t + 1$.

5. until The difference of the solutions between two adjacent iterations, *i.e.*, $\|\mathbb{A}^{t+1} - \mathbb{A}^t\|$, is below a threshold θ_1.

Based on Lemma 1 and Lemma 2, the SCA-based algorithm is summarized by Algorithm 1. The term $\check{\eta}(\mathcal{V}; \mathbb{A}^t)$ represents the energy efficiency $\check{\eta}(\mathcal{V})$ in (15) with the given value in auxiliary variable set $\mathbb{A}^t$. Note that the approximated problem inside the loop (Steps 3 and 4 in Algorithm 1) is a fractional programming problem and still non-convex. We will provide the optimal solution of the approximated problem in the remainder of the section. The convergence of SCA has been proven in [32], and the algorithm will stop after finite iterations if the local optimizer exists.

B. Dinkelbach Algorithm

Problem (20) is a fraction programming problem. We can adopt the Dinkelbach algorithm to achieve the optimal solution. The objective function (20) can be rewritten as the following parametric programming form:

$$F^t(\alpha) = \max_{\mathcal{V}} \quad \{\sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{I}} \check{R}_{i,k} - \alpha[\sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{I}} E_{i,k}^{C,U}(\mathbf{W}_k) + \sum_{k\in\mathcal{K}} \hat{E}_k^F] | \mathcal{V} \in \mathcal{F}^t\},$$

(21)

where $\mathcal{F}^t$ represents the feasible set of problem (20) at the t-th iteration in Algorithm 1. The function $F^t(\alpha)$ is a monotonic decreasing function of α. Let the term α^* denote the solution of $F^t(\alpha^*) = 0$. Due to the monotone decreasing property of $F^t(\alpha)$, $F^t(\alpha^*) = 0$ if and only if α^* is equal to the optimal result of problem (20), *i.e.*, $\alpha^* = \check{\eta}(\mathcal{V}^*; \mathbb{A}^t)$ [33]. The algorithm for solving problem (20) is shown in Algorithm 2.

Algorithm 2: Dinkelbach Algorithm for Solving Problem (20).

1. Initialize $\alpha^0 = 0$ if $t = 0$, $\alpha^0 = \alpha^*$ in loop $t-1$ if $t \geq 0$, and the loop index $m = 0$.
2. **repeat**
3. Solve problem (21) for given α^m, and denote the solution for the problem by $\mathcal{V}_d^m$.
4. Update the Dinkelbach auxiliary variable $\alpha^{m+1} = \check{\eta}(\mathcal{V}_d^m; \mathbb{A}^t)$.
5. $m = m + 1$.
6. **until** $F^t(\alpha^{m+1}) \leq \theta_2$.

Due to the nature of the SCA-based algorithm and Dinkelbach algorithm, we can further cut the iteration times based on the following Lemma.

Lemma 3:

Denote the optimal Dinkelbach parameter α^* for two consecutive SCA iterations by $\alpha^*(t-1)$ and $\alpha^*(t)$. We have $\alpha^*(t-1) \leq \alpha^*(t)$, and $F^t(\alpha^*(t-1)) \geq F^t(\alpha^*(t)) = 0$.

Proof:

Denote the optimization results and the corresponding Dinkelbach parameter at iteration $t-1$ by $\mathcal{V}^*(t-1)$ and $\alpha^*(t-1)$, respectively. From Dinkelbach algorithm, we have $\alpha^*(t-1) = \check{\eta}^*(\mathcal{V}^*(t-1); \mathbb{A}^{t-1}) \leq \check{\eta}^*(\mathcal{V}^*)$. As shown in Lemma 2, the approximated function provides the global lower bound of the original optimization function, and the results have to be inside the feasible set of the approximate optimization function for the next iteration. Thus, $\check{\eta}^*(\mathcal{V}^*(t-1); \mathbb{A}^{t-1}) \leq \check{\eta}(\mathcal{V}^*(t-1); \mathbb{A}^t) \leq \check{\eta}^*(\mathcal{V}^*(t); \mathbb{A}^t)$. Therefore, $\alpha^*(t-1) \leq \alpha^*(t)$. Moreover, due to the monotonically decreasing nature of $F(\alpha)$, $F^t(\alpha^*(t-1)) \geq F^t(\alpha^*(t)) = 0$.■

Given Lemma 3, the initial point in iteration t, *i.e.*, $\alpha^0(t)$, in Algorithm 2 can be set at $\alpha^*(t-1)$ rather than 0 so that the computation efficiency of the optimization algorithm can be further improved.

C. Sub-Problem Decomposition by ADMM

By now, the UAV computation energy efficiency maximization problem has been transformed into a solvable form. However, solving problem (21) is time-consuming due to multiple second order cone (SOC) constraints and requires the local information exchange between the UAV and users. Therefore, we propose a distributed solution, in which users maximize their offloaded computation tasks in parallel while the UAV aims to minimize its energy consumption. The original problem is decomposed into several sub-problems without losing optimality, and the UAV and users solve the optimization problem cooperatively. Local information, such as the mobility of users and the propulsion energy consumption function of the UAV, is not required to be shared among users and the UAV.

We adopt ADMM technique to decompose problem (21) [34]. The optimization solution is achieved in an iterative manner. Firstly, we introduce an auxiliary variable, $\mathbb{G}$, which is solved by users:

$$\mathbb{G} = \begin{bmatrix} \ddot{\mathbf{Q}}_{1,1} & \cdots & \ddot{\mathbf{Q}}_{N,1} & W_{1,1} & \cdots & W_{N,1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \ddot{\mathbf{Q}}_{1,K} & \cdots & \ddot{\mathbf{Q}}_{N,K} & W_{1,K} & \cdots & W_{N,K} \end{bmatrix}^T$$

where $\ddot{\mathbf{Q}}_{i,k}$ denotes the UAV location in time slot k expected by user i. Each user solves a part of the matrix $\mathbb{G}_i = [\ddot{\mathbf{Q}}_{i,1}, W_{i,1}; \ldots; \ddot{\mathbf{Q}}_{i,K}, W_{i,K}$, and updates it to the UAV. Then, the UAV generates its trajectory, $\mathbf{Q}$, and overall computation load allocation according to the uploaded matrix $\mathbb{G}$. Denote the overall amount of computation load processed in slot k at UAV by V_k, where $\mathbf{V} = [V_1; \ldots; V_K]$. The results determined by the UAV are summarized by matrix $\mathbb{H}$, where $\mathbb{H} = [\mathbb{I}_{(N\times 1)}\mathbf{Q}; \mathbf{V}]$. $\mathbb{I}_{(N\times 1)}$ is a vector where all N entries are 1. By the end of the ADMM algorithm, the expected UAV trajectories should be unified and follow the flying constraints. The computation load should be allocated under the UAV computing capability. Thus, in the final optimal solution, the following constraint should be satisfied:

$$\mathbb{P}^T\mathbb{G} = \mathbb{H},$$

(22)

where

$$\mathbb{P} = \begin{bmatrix} \mathbb{I}_{(N\times N)} & \mathbf{0}_{(N\times 1)} \\ \mathbf{0}_{(N\times N)} & \boldsymbol{\chi} \end{bmatrix}$$

The vector χ represents the computation intensity for users' tasks, where $\boldsymbol{\chi} = [\chi_1; \ldots; \chi_N]$. The sub-matrices $\mathbb{I}_{(N\times N)}$ and $\mathbf{0}_{(N\times N)}$ denote N-by-N identity matrix and zero matrix, respectively.

In addition, for the non-orthogonal channel model, we introduce another auxiliary variable, $e_{i,k}$, which denotes the summation of $\xi_{j,k}$ in all other users except user i. This variable is used to decouple the correlated $\xi_{j,k}$ in (15h) to facilitate the independent optimization process at each user. At the end of the optimization, $e_{i,k}$ should be equal to $\sum_{j\in\mathcal{I}/\{i\}} \xi_{j,k}$. For simplicity of presentation, we transform this constraint as follows:

$$\frac{1}{N}(e_{i,k} + \xi_{i,k}) = \bar{\xi}_k,$$

(23)

where $\overline{\xi}_k$ is the mean of $\{\xi_{1,k}, \ldots, \xi_{N,k}\}$. Then, the augmented Lagrangian function is formulated as follows:

$$\begin{aligned}\Gamma(\mathcal{V}_{\mathrm{A}}) = & -\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}} \check{R}_{i,k} + \alpha\left[\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}} E_{i,k}^{C,U}(\mathbf{W}) + \sum_{k\in\mathcal{K}} E_k^F\right] \\ & + \mathrm{Tr}\{\mathbf{U}_1^T(\mathbb{P}^T\mathbb{G} - \mathbb{H})\} + \frac{\rho_1}{2}\|\mathbb{P}^T\mathbb{G} - \mathbb{H}\|_F^2 \\ & + \varpi\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{I}}\{U_{2,i,k}[\frac{1}{N}(e_{i,k} + \xi_{i,k}) - \overline{\xi}_k] \\ & + \frac{\rho_2}{2}\left[\frac{1}{N}(e_{i,k} + \xi_{i,k}) - \overline{\xi}_k\right]^2\},\end{aligned}$$

(24)

where $\|\cdot\|_F$ is the notation representing the Frobenius norm. Set VA represents variables $\{\mathcal{V}, \mathbb{G}, \mathbb{H}, \mathbf{U}_1, \mathbf{U}_2\}$. Variables $\mathbf{U}_1 \in \mathbf{R}^{(N+1)\times K}$ and $\mathbf{U}_2 \in \mathbf{R}^{N\times K}$ are Lagrangian multipliers for the two auxiliary constraints, (22) and (23), respectively. Two parameters, ρ_1 and ρ_2, are penalty parameters. The parameter ϖ indicates the channel model. $\varpi = 1$ denotes the case of the non-orthogonal channel access scheme, and $\varpi = 0$ denotes the case of the orthogonal channel access scheme.

Problem (21) can be separated into two sub-problems. The sub-problem solved in user i is organized as follows:

$$\begin{aligned}\min_{\mathcal{V}_1} & -\sum_{k\in\mathcal{K}} \check{R}_{i,k} + \mathrm{Tr}\left\{(\mathbf{U}_{1,i}^{n-1})^T\mathbb{P}_i^T\mathbb{G}_i\right\} \\ & + \frac{\rho_1}{2}\|\mathbb{P}_i^T\mathbb{G}_i - \mathbb{J}_i^{n-1}\|_F^2 \\ & + \varpi\left\{\frac{U_{2,i,k}^{n-1}(e_{i,k})}{N} + \frac{\rho_2}{2}\left(\frac{e_{i,k} - e_{i,k}^{n-1}}{N} + \theta_{i,k}^{n-1}\right)^2\right. \\ & \left. + \sum_{j\in\frac{\mathcal{I}}{\{i\}}} -\frac{U_{2,j,k}^{n-1}\xi_{i,k}}{N} + \frac{\rho_2}{2}\left(\frac{\theta_{j,k}^{n-1}}{N-1} + \frac{\xi_{i,k}^{n-1} - \xi_{i,k}}{N}\right)^2\right\} \\ & \text{s.t.} \frac{(\|\mathbf{Q}_{i.k} - \mathbf{q}_{i,k}\|_2^2 + H^2)n_0}{g_0} \le l_{i,k}, \forall i,k \\ & (7a),(7b),(10),(14d),(15g),(16), \\ & (15a), \text{if } \varpi = 0, \\ & (19), \text{if } \varpi = 1,\end{aligned}$$

(25a)(25b)

and the sub-problem solved in the UAV is organized as follows:

$$\begin{aligned} \min_{\mathcal{V}_2} \quad & \alpha\left[\sum_{k\in\mathcal{K}}\frac{\kappa V_k^3}{\Delta^2}+\sum_{k\in\mathcal{K}}\hat{E}_k^F\right]-\mathrm{Tr}\{(\mathbf{U}_1^n)^T\mathbb{H}\} \\ & +\frac{\rho_1}{2}\|\mathbb{P}^T\mathbb{G}^n-\mathbb{H}\|_F^2 \\ \text{s.t.} \quad & \frac{V_k}{\Delta}\le f_{max}^U,\forall k, \\ & (14a),(14b),(15d),(18),(17). \end{aligned}$$

(26a)(26b)

The term $(x)^{n-1}$ represents the variable x obtained in iteration $n-1$. The Lagrangian multipliers $\mathbf{U}_1$ and $\mathbf{U}_2$ are updated at each iteration as follows:

$$\begin{aligned} \mathbf{U}_1^n &= \mathbf{U}_1^{n-1}+\rho_1(\mathbb{P}^T\mathbb{G}^n-\mathbb{H})^n \\ \mathbf{U}_{2,i,k}^n &= \mathbf{U}_{2,i,k}^{n-1}+\rho_2\theta_{i,k}^n, \end{aligned}$$

(27a)(27b)

where $\theta_{i,k}^n$ is

$$\theta_{i,k}^n=\frac{1}{N}\left(e_{i,k}^n+\xi_{i,k}^n\right)-\overline{\xi}_k^n.$$

(28)

Algorithm 3: ADMM Algorithm for Solving Problem (21).

1. Initialize variables $\{\mathbf{e}^0,\boldsymbol{\xi}^0,\boldsymbol{\theta}^0,\mathbb{H}^0,\mathbb{G}^0\}$ and dual variables $\{\mathbf{U}_1^0,\mathbf{U}_2^0\}$. Loop index $n=0$.
2. **repeat**
3. **For each user** i:
4. *If* $\varpi=0$*:* Wait until receive updated $\mathbb{J}_i^{n-1}$.
5. *If* $\varpi=1$*:* Wait until receive updated $\{\mathbb{J}_i^{n-1},\boldsymbol{\theta}^{n-1}\}$.
6. Calculate the dual variable $\mathbf{U}_{1,i}^{n-1}=\mathbf{U}_{1,i}^{n-2}+\rho_1(\mathbb{P}_i^T\mathbb{G}_i^{n-1}-\mathbb{J}_i^{n-1})$.
7. Calculate the dual variable $\mathbf{U}_2$ for all $i\in\mathcal{I}$ by (27b).
8. Solve problem (25).
9. *If* $\varpi=0$*:* Send $\mathbb{G}_i^n$ to the cloudlet.
10. *If* $\varpi=1$: Send $\{\mathbb{G}_i^n,\mathbf{e}_i^n,\boldsymbol{\xi}_i^n\}$ to the cloudlet.
11. **For the UAV-mounted cloudlet**:
12. Gather information from users to form matrix $\mathbb{G}^n$.
13. Solve problem (26), and update Hn.
14. Update dual variable Un1 by (27a)
15. *If* $\varpi=1$*:* Update variables $\theta_{i,k}^n\forall i,k$ by (28), and send the variables to users.
16. $n=n+1$.
17. **until** $|\Gamma^n(\mathcal{V},\mathbb{G},\mathbf{V},\mathbf{U}_1,\mathbf{U}_2)-\Gamma^{n-1}(\mathcal{V},\mathbb{G},\mathbf{V},\mathbf{U}_1,\mathbf{U}_2)|\le\theta_3$.

$\theta_{i,k}$ represents the difference between the user expected interference and the real interference. At iteration n, problem (25) is solved by each user individually. The optimization variable set $\mathcal{V}_1$ includes $\{\delta_{i,k}, W_{i,k}, \ddot{\mathbf{Q}}_{i,k}, \xi_{i,k}, \check{\mathbf{l}}, \check{\mathbf{R}}, e_{i,k}\}$ for all $k \in \mathcal{K}$. To decompose the auxiliary constraint (22) for each user i, we introduce sub-matrices $\mathbb{P}_i$, $\mathbb{H}_i$, and $\mathbf{U}_{1,i}$, which are defined as follows: The parameter matrix $\mathbb{P}_i$ is the sub-matrix sliced from $\mathbb{P}$, where $\mathbb{P}_i = \mathbf{diag}\{1, \chi_i\}$. The matrix $\mathbb{J}_i$ is obtained by the information from the UAV, where $\mathbb{J}_i^n = \left[\mathbf{Q}^n; \frac{\mathbf{v}^n}{N} + \chi_i {W_i}^n - \sum_{j\in\mathcal{J}} \frac{\chi_j \mathbf{w}_j^n}{N}\right]$. The sub-matrix $\mathbf{U}_{1,i}$ is sliced from the dual variable, where $\mathbf{U}_{1,i} = [\mathbf{U}_1(i,:); \mathbf{U}_1(N+1,:)]$. The detailed decomposition process is omitted due to the space limit. Subsequently, problem (26) is solved by the UAV. The optimization variable set $\mathcal{V}_2$ includes $\{\mathbf{Q}, \boldsymbol{\omega}, \mathbf{A}, \hat{\mathbf{E}}^F\}$.

Lemma 4:

If the initial value of $\{\mathbf{e}^0, \boldsymbol{\xi}^0, \mathbf{U}_1^0, \mathbf{U}_2^0\}$ is shared and unified among all users and the UAV, only information from the UAV required for computing the sub-problem on the user side at each iteration is $\{\mathbb{J}_i^{n-1}, \boldsymbol{\theta}^{n-1}\}$.

Proof:

If the initial value is unified among the UAV and users, the dual variables are not required to be shared and can be computed locally by the UAV and users. For computing the dual variable $\mathbf{U}_{1,i}$ at n, the following knowledge is required: the updated global value $\mathbb{J}_i^{n-1}$, the historical value for the local information $\mathbb{G}_i^{n-1}$, and the historical value of the dual variable $\mathbf{U}_{1,i}^{n-1}$. Therefore, if $\mathbf{U}_{1,i}^0$ is identical to all users and the UAV, $\mathbf{U}_{1,i}^n$ can be synchronized according to the historical value and the value from the global variable. Similarly, $\mathbf{U}_2$ can be updated by users if the initial value is known.■

Consider the condition in Lemma 4, the distributed algorithm is given in Algorithm 3. In each optimization iteration, user side computes and share matrix $\mathbb{G}$ to the UAV, and UAV computes and shares the matrix $\mathbb{J}$ to users. Meanwhile, when $\varpi = 1$, excepting contributing matrix $\mathbb{G}_i$, user i needs the information $e_{j,k}$ and $\xi_{j,k}$ from other users $j \in \mathcal{I}/\{i\}$ to evaluate the interference.

By the problem decomposition, at the user side, each user only aims to maximize its own offloading data given the UAV trajectory computed by the UAV-mounted cloudlet and the interference environment in the previous iteration. At the UAV-mounted cloudlet side, the UAV aims to minimize energy consumption under the users' expected UAV trajectories to collect enough workload. The trade-off between the received offloaded tasks and the energy consumption is controlled by the parameter α which is updated out of the ADMM algorithm loop. Meanwhile, the corresponding variables and constraints are split into two groups. This introduces three main advantages. Firstly, local variables and parameters, such as user location and user offloading constraints, are not required to be uploaded to the UAV. Similarly, UAV's mechanical parameters and settings are not required to be shared to users for offloading optimization. Secondly, less configuration is required when the UAV is replaced. Thirdly, the main computation load in solving the problem is from the SOC programming. The SOC constraints are now decomposed and solved by users in parallel such that the computation efficiency can be improved. For ADMM algorithm, in the orthogonal channel model, there are two main distributed blocks: the user side and the UAV side. The convergence of ADMM is guaranteed when the number of blocks is no more than two. In the non-orthogonal channel model, since each user is required to compute the interference variable $e_{i,k}$ parallelly, convergence is not always guaranteed. Proximal Jacobian ADMM can be adopted to ensure the convergence, in which the proximal term $\frac{\tau}{2}||x_i - x_i^k||^2$ is further combined in the primal problem of the current algorithm [35].

D. Convergence and Complexity Analysis

The convergence for the three loops in Algorithms 1 to 3 is guaranteed. For the SCA-based algorithm, if the problem is feasible and the initial values of the approximate variables are in the feasible set of the original optimization problem (14), the algorithm convergence is ensured [32]. Moreover, the Dinkelbach algorithm can achieve the optimal α^* with a super-linear rate.

The computation complexity of the problem is dominated by the SOC programming [13], [36]. Suppose that Algorithm 3 runs $L_1 \times L_2$ iterations, where the SCA algorithm loop repeats L_1 times, and the loop for the Dinkelbach algorithm repeats L_2 times. The problem before decomposition, *i.e.*, problem (21), has KN SOC constraints in 4 dimensions, K SOC constraints in 7 dimensions, and KN SOC constraints in 2 dimensions, where $6KN + 4K$ variables participates in those constraints. The overall complexity can be $L_1 L_2 O(\sqrt{2KN + K}(6KN + 4K)(20KN + 49K + (6KN + 4K)^2))$. After ADMM decomposition, for the sub-problem on the user side, there are K SOC constraints in 4 dimensions and K SOC constraints in 2 dimensions. Thus, the computation complexity is $L_1 L_2 O(1/\theta_3) O(\sqrt{2K}(5K)(20K + (5K)^2))$ for each user. On the UAV side, the sub-problem contains K SOC constraints in 7 dimensions. The complexity is $L_1 L_2 O(1/\theta_3) O(\sqrt{K}(2K)(49K + (2K)^2))$.

SECTION VI. Proactive Trajectory Design Based on Spatial Distribution Estimation

So far, we have introduced the trajectory design and resource allocation for the scenario that all computation load information and user location are known. However, some IoT nodes have a certain mobility [37]. It is hard for users to know their future positions during the upcoming computation cycle. Moreover, users needs to send the offloading requests at the beginning of the cycle. It means that the user may buffer the computation task until a new cycle begins, which introduces extra delay for waiting to send the request. Thus, the maximum queue delay may reach to T seconds. To deal with the above issues, in this subsection, we introduce an approach to estimate the spatial distribution of user locations in a cycle. The mobility of users is predicted by an unsupervised learning tool, kernel density estimation method [38], and the computation load of each user is considered in a stochastic model correspondingly. The UAV trajectory is optimized via the estimated knowledge about ground users. Thus, UAV can collect the offloaded tasks of users without requesting in advance.

To estimate the location of users, each user need to report its current location periodically. The sampled location of user i is represented by q_i. We use the sampled location to estimate the spatial distribution of users for the cycle, where the probability density function for the user at (x, y) is denoted as $f(x, y)$.

In order to compute $f(x, y)$, consider a small region $\mathbf{R}$ which is a rectangle area with side length of h_x and h_y, *i.e.*, Parzen window. To count the number of users falling within the region, we define the following function to indicate if user i is in the area:

$$C(q_i^x, q_i^y; \mathbf{R}) = \begin{cases} 1, & \text{if } \max\left\{\dfrac{||q_i^x - x||}{h_x}, \dfrac{||q_i^y - y||}{h_y}\right\} \leq \dfrac{1}{2} \\ 0, & \text{otherwise,} \end{cases}$$

(29)

where (x, y) is the central point of the area. Thus, for a large N, the general expression for non-parametric density estimation is [38]

$$f(x,y)=\frac{1}{Nh_xh_y}\sum_{i\in\mathcal{I}}C(q_i^x,q_i^y;\mathbf{R}).$$

(30)

To establish continuous estimation function, a smooth Gaussian kernel is applied, where

$$\hat{f}(x,y)=\frac{1}{N\sqrt{h_xh_y}}\sum_{i\in\mathcal{I}}\frac{1}{2\pi}e^{-[\frac{(q_i^x-x)^2}{2h_x}+\frac{(q_i^y-y)^2}{2h_y}]}.$$

(31)

The term $\hat{f}(x,y)$ is the distribution of Gaussian kernel estimation. In (31), h_x and h_y represent the bandwidth of the Gaussian kernel rather than the side length of the Parzen window. To improve the estimation quality, the proper bandwidth, h_x and h_y, needs to be selected to minimize the error between the estimated density and the true density. In this work, the maximum likelihood cross-validation method [38], [39] is adopted to determine the bandwidth h_x and h_y. The optimal bandwidth is

$$[h_x^*,h_y^*]=\text{argmax}\left\{\frac{1}{N}\sum_{i\in\mathcal{I}}\text{og }\hat{f}_{-i}(q_i^x,q_i^y)\,\text{l}\right\},$$

(32)

where $\hat{f}_{-i}(q_i^x,q_i^y)$ is the estimated distribution in which user i is left out of the estimation. In order to apply the estimated distribution into our proposed approach, we divide the working area of the UAV $\mathcal{A}$ into $G\times G$ sub-areas. For sub-area $\mathcal{A}_i$, there is a virtual user located at the center of the area. The virtual user carries all the computation tasks in the sub-area. It is assumed that the distribution of the task input data size and user spatial location are independent. The expected length of input bits for the tasks generated by a user by $\mathbb{E}[X]$. Thus, expected length of computing bits generated inside sub-area $\mathcal{A}_i$ is

$$\mathbb{E}[I_i]=\mathbb{E}[X]\mathbb{E}[N_i]=\mathbb{E}[X]\int_{(x,y)\in\mathcal{A}_i}\hat{f}(x,y)dxdy,$$

(33)

where $\mathbb{E}[N_i]$ denotes the expected number of users in the sub-area $\mathcal{A}_i$. Our proposed approach can now be adopted to solve the problem: In the new problem, there are G^2 virtual users participating in the computation task offloading, and virtual user i has $\mathbb{E}[I_i]$ computation load to be done in a cycle. The location of user i is fixed at the center of the sub-area. For the orthogonal channel model, the virtual user i shares a portion of $\mathbb{E}[N_i]/N$ of the channel bandwidth. As G increases, the performance of the estimation will be improved correspondingly.

SECTION VII. Numerical Results

In this section, we evaluate the performance of our proposed optimization approach. The parameter settings are given in Table II. The channel gain parameter g_0 is −70 dB. Let the term p represent the percentage of computation tasks that have to be offloaded to the cloudlet, *i.e.*, $(\check{I}_i/I_i) * 100\%$. We consider that users have homogeneous offloading requirements in the simulation, *i.e.*, E_i^T and p are identical for all user. The term "NO" represents the non-orthogonal channel access scheme, and the term "O" represents the orthogonal channel access scheme. We also consider the circular trajectory scheme as the benchmark, where the UAV moves around a circle within a cycle, with the circle center located at (0.5, 0.5) km, and the radius is predefined. Two network scenarios are considered: a three-node scenario and a four-node scenario. In the three-node scenario, there are three users located at (0, 1) km, (1, 1) km, and (1, 0) km, as shown in Fig. 2(a). At the beginning of the cycle, the UAV moves from the location (0, 0) at an initial speed (−10, 0) m/s. By the end of the cycle, the UAV returns to the final designated position at (0.5, 0) km. In the four-node scenario, there are four users located at the randomly generated locations. The users travel at constant speeds which are random selected from [−3, −3] m/s to [3, 3] m/s, as shown in Fig. 2(b). The UAV moves from the location (200, 200) m at an initial speed (−10, 0) m/s and returns to the initial position at the end of the cycle.

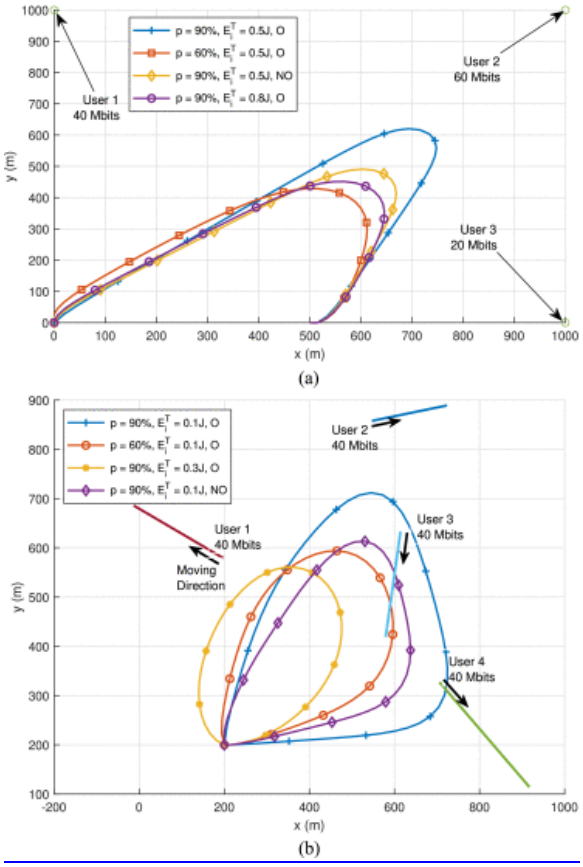

Fig. 2. Optimal UAV trajectories with different parameter settings: (a) the three-node scenario; (b) the four-node scenario with user mobility, where the solid straight lines represent user trajectories, and the arrows represent user moving directions.

TABLE II Parameter Settings for the Three-Node Scenario

Parameter	Value	Parameter	Value
B	3 MHz	κ	10-28

σ^2	-80 dBm/Hz	γ_1	0.0037
χ_i	1550.7	γ_2	500.206
Δ	1.5 s	H	100 m
$\alpha_{\max}$	50 m/s2	P	100 mW
$v_{\max}$	35 m/s	K	50

The UAV trajectory results obtained by the proposed approach are shown in Fig. 2. In the three-node case shown in Fig. 2(a), the UAV takes most of the time moving towards and stays around the location of user 2 due to high computation task loads of the user. With a higher minimum offloading requirement p, the UAV moves closer to users in order to collect more offloading tasks. Similarly, with a lower maximum communication energy requirement E_i^T, the UAV also moves closer to users to reduce the user's offloading communication energy consumption. Moreover, since the non-orthogonal access method has a higher channel capacity, under the same condition, the trajectory of the non-orthogonal case is shrunk to preserve the mechanical energy consumption compared to the orthogonal channel case. Similar results can be obtained in the four-node case, as shown in Fig. 2(b).

The comparisons of the energy efficiency with different settings are shown in Fig. 3. In Figs. 3(a) and 3(b), the x-axis represents the iteration number of the SCA-based algorithm loop. As shown in Fig. 3(a), the energy efficiency converges at $t = 30$ in the three-node scenario, while the number of iterations till convergence is increased in the four-node scenario. Moreover, for both scenarios, with loose user offloading requirements, the energy efficiency is improved due to the expanded optimization feasible set. In contrast, with tight user offloading requirements, the energy efficiency is decreased significantly due to high energy consumption for the UAV to move closer to the users.

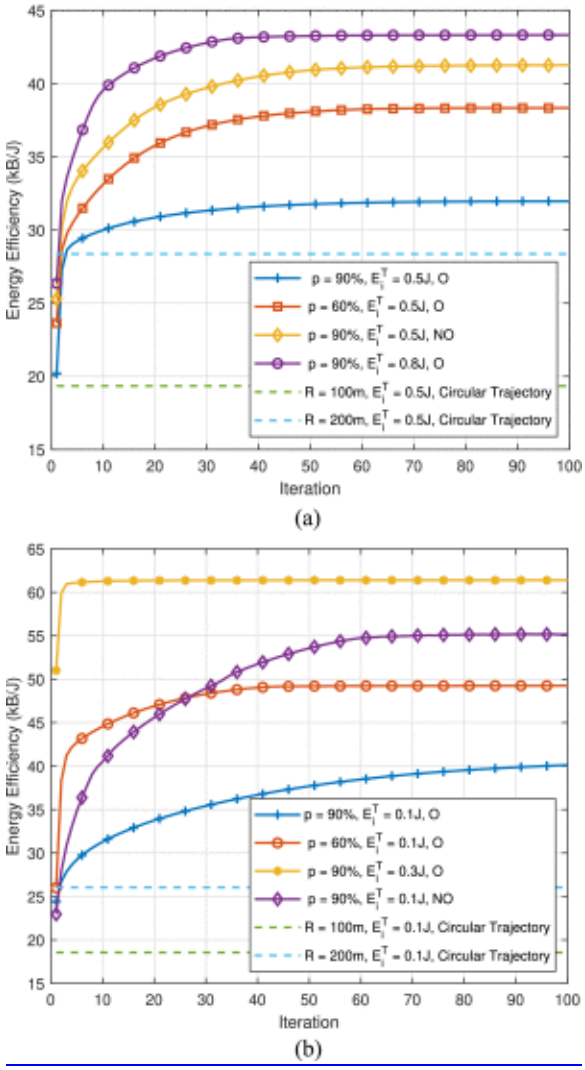

Fig. 3. Energy efficiency versus main loop iteration number with different trajectory designs: (a) the three-node scenario; (b) the four-node scenario with user mobility.

For the three-node case, the ratio of the offloaded data amount to the overall computing data amount is shown in Fig. 4. The parameter setting for the indexes are given in Table III, where the results by the proposed approach are shown in 1–6, and the results by the circular trajectory are shown in 7–9. For all scenarios, the proposed approach can achieve the minimum offloading requirement, while the circular trajectory scheme cannot guarantee to achieve the requirement. Moreover, when the maximum communication energy requirement E_i^T is increased, the UAV can collect more data even though its trajectory is far away from users compared to the case with a lowE_i^T . The UAV also collects the extra offloaded tasks, which is beyond the users' requirement, to improve its energy efficiency.

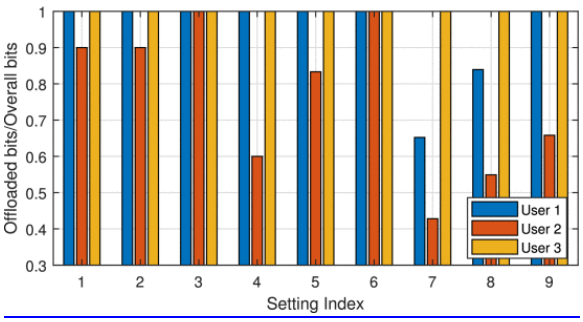

Fig. 4. The ratio of the offloaded task data amount to the overall computation task data among generated by users with different parameter settings.

TABLE III Parameter Setting for Fig. 4

Index	p	E_i^T	Index	p	E_i^T	Index	Radius	E_i^T
I	90%	0.5 J	4	60%	0.5 J	7	200 m	0.5 J
2	90%	0.8 J	5	60%	0.8 J	8	200 m	0.8 J
3	90%	1.1 J	6	60%	1.1 J	9	200 m	1.1 J

The trade-off between the maximum offloading energy, *i.e.*, E_i^T, and the energy efficiency in the three-node case is shown in Fig. 5(a). As E_i^T increases, the energy efficiency of the UAV is increased at first and hits the ceiling in a high E_i^T. At that point, E_i^T is not the factor that limits the energy efficiency performance since all user's computing data is collected as shown in Fig. 5(c). When the energy efficiency reaches the maximum value, the UAV will find a path that has minimum energy consumption given that all tasks are offloaded. Furthermore, our proposed approach can improve the energy efficiency significantly compared to the circular trajectory.

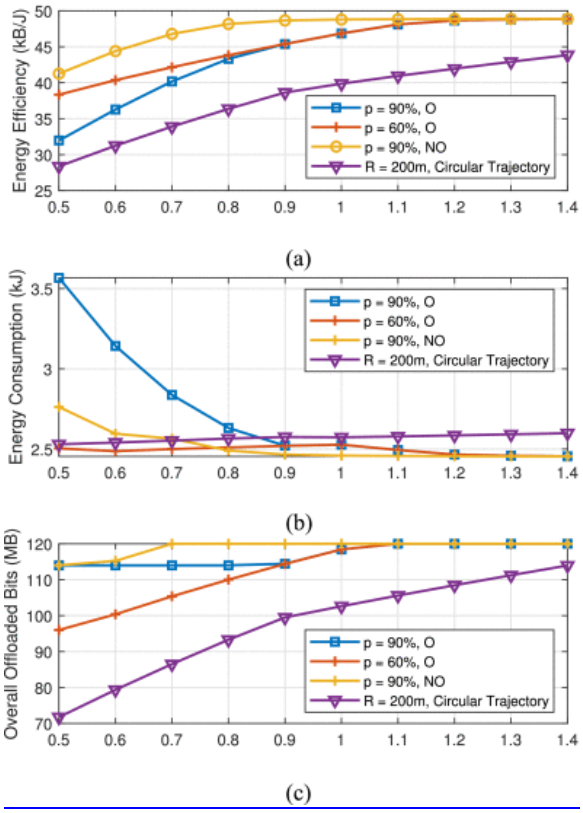

Fig. 5. (a) Energy efficiency versus the maximum offloading communication energy E_i^T with different settings. (b) Overall energy consumption in a cycle versus the maximum offloading communication energy E_i^T. (c) Overall offloaded bits in a cycle versus the maximum offloading communication energy E_i^T.

The magnitudes of the UAV acceleration and velocity in the three-node case are shown in Fig. 6(a) and Fig. 6(b), respectively. The final velocity is constrained to be equal to the initial velocity. Note that the optimal velocity cannot be zero due to the characteristic of fixed-wing UAV. With the lower maximum energy requirement, both magnitudes of acceleration and velocity are increased, such that the UAV can move closer to users. With the higher energy requirement, the fluctuation on velocity and acceleration decreases to reduce the propulsion energy consumption of the UAV.

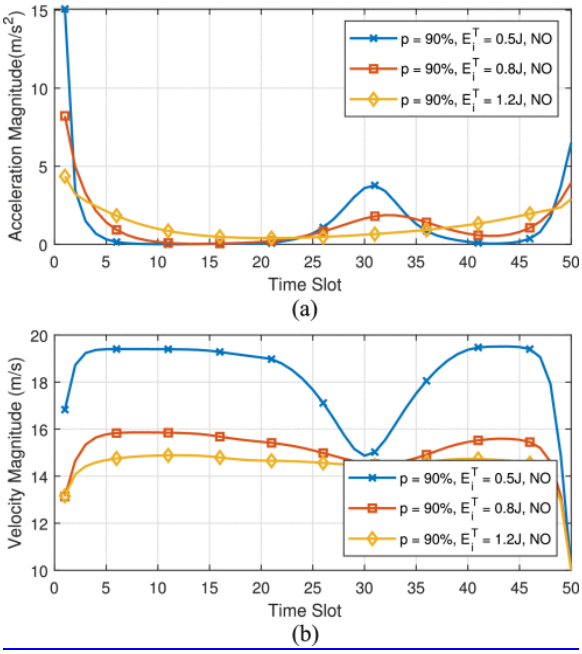

Fig. 6. (a) The acceleration of the UAV in the cycle. (b) The speed of the UAV in the cycle.

The ratio of the actual allocated transmit power to the maximum power, $\delta_{i,k}$, for the three users in a cycle is shown in Fig. 7(a). Note that the overall offloading communication energy is limited. For the user with high offloading demands, *i.e.*, user 2, the ratio is maximized when the UAV moves adjacent to it, while the ratio is minimized when the UAV moves away from it. The user tends to preserve the communication energy and starts the offloading only when the data rate is high. However, for user 3, the transmit power is still allocated when the UAV is far away from the location of the user for two reasons: Firstly, the maximum communication energy of the user allows user uploading the data even though the user transmission efficiency is low. Secondly, the UAV-mounted cloudlet prefers collecting the data in advance such that it can balance the computation load to reduce the computing energy cost. The computation load allocation of the cloudlet in the three-node case is shown in Fig. 7(b). Since the energy consumption is cubically increased as the computation load in a unit time increased (based on (8) and (13)), the computation load is preferred to be balanced among time slots. However, the computation load can only be executed after the corresponding tasks are offloaded into the cloudlet. Therefore, in the case with limited maximum communication energy, the allocated computation load is increased only when the new offloaded tasks are received. In contrast, with the loose maximum communication energy constraint, the workload fluctuation is reduced significantly to minimize the computing energy consumption.

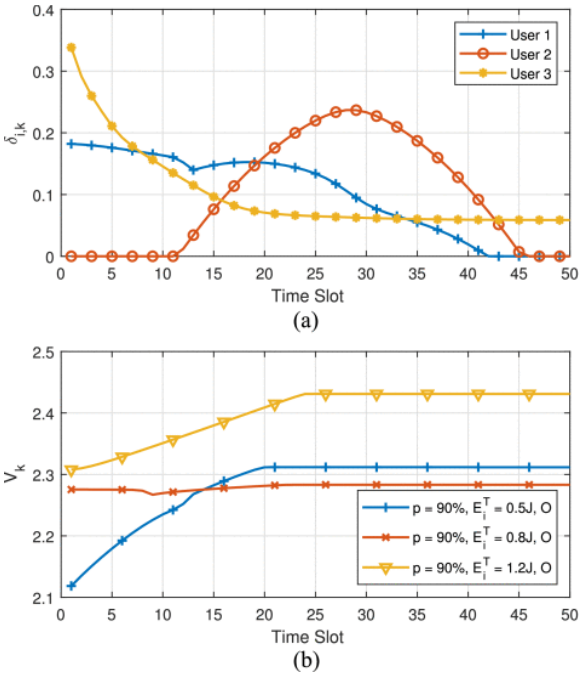

Fig. 7. (a) The transmit power allocation among three users, where p = 90%, and E_i^T = 0.5 J under orthogonal channel scenario. (b) The workload allocation with different settings.

SECTION VIII. Conclusion

In this paper, an optimization approach has been proposed to maximize the energy efficiency of a UAV-assisted MEC system, where the UAV trajectory design and resource allocation have been jointly considered. The non-convex and non-linear energy efficiency maximization problem has been solved in a distributed manner. Moreover, the node mobility estimation has been adopted to design a proactive UAV trajectory when the knowledge of user trajectory is limited. Our work can offer valuable insights on UAV optimal trajectory design for providing on-demand edge computing service for remote IoT nodes. In the future, considering the uncertainty of user mobility and the time-invariant computation demand, we will focus on the online resource management in UAV-assisted MEC system under a dynamic channel environment.

Appendix A Proof of Lemma 1

Firstly, to deal with the non-convex function on the numerator, *i.e.*, $R_{i,k}(\delta_{i,k}, \mathbf{Q}_k)$, we introduce the auxiliary variable $\check{R}_{i,k}$ to indicate the lower bound of the data rate for user i in slot k. Moreover, we introduce two auxiliary variables: the term $\xi_{i,k}$, where $\xi_{i,k} \leq \delta_{i,k}P/l_{i,k}$, and the term $l_{i,k}$, where $l_{i,k} \geq N_0/h_{i,k}$. Thus, the following relation can be established

$$\check{R}_{i,k} \leq \frac{B\Delta}{N}\log\left(1+\xi_{i,k}\right) \leq R_{i,k}(\delta_{i,k}, \mathbf{Q}_k),$$

(34)

where $\check{R}_{i,k}$ is the epigraph form of $R_{i,k}(\delta_{i,k}, \mathbf{Q}_k)$. When (15) is maximized, *i.e.*, the numerator $\check{R}^*_{i,k}$ is maximized, we have $l^*_{i,k} = 1/g^*_{i,k}$, $\xi^*_{i,k} = \delta^*_{i,k}P/l^*_{i,k}$, and $\check{R}^*_{i,k} = R_{i,k}(\delta^*_{i,k}, \mathbf{Q}^*_k)$.

Furthermore, to deal with the non-linear function on the denominator, *i.e.*, $E_k^F(\mathbf{Q})$, we introduce an auxiliary variable $\hat{E}_k^F$ to indicate the upper bound of the UAV propulsion energy in slot k. For the non-linear part of the function, we introduce two auxiliary variables: the term ω_k, where $\omega_k^2 \le \|\mathbf{v}_k(\mathbf{Q})\|_2{}^2$, and the term $A_{i,k}$, where $A_{i,k} \ge (1/\omega_k)(1 + \|a_k(\mathbf{Q})\|_2^2/g^2)$. Thus, we have

$$\begin{aligned}\hat{E}_k^F &\ge \gamma_1\|\mathbf{v}_k(\mathbf{Q})\|_2^3 + \gamma_2 A_k \\ &\ge \gamma_1\|\mathbf{v}_k(\mathbf{Q})\|_2^3 + \gamma_2\frac{1}{\omega_k}\left(1 + \frac{\|a_k(\mathbf{Q})\|_2^2}{g^2}\right) \ge E_k^F(\mathbf{Q}).\end{aligned}$$

(35)

Similarly, when (15) is maximized, *i.e.*, the denominator $E_k^F(\mathbf{Q})$ is minimized, $\hat{E}_k^{F*} = E_k^F(\mathbf{Q}^*)$. Therefore, problem (15) is equivalent to problem (14), and $\eta^* = \check{\eta}^*$.

Appendix B Proof of Lemma 2

Constraint (15b) can be transformed into the following equivalent form:

$$(\xi_{i,k} + l_{i,k})^2 - (\xi_{i,k} - l_{i,k})^2 \le 4\delta_{i,k}P,$$

(36)

which is difference of convex functions [32]. Then, we approximate the second part of the equation by the Taylor expansion:

$$\begin{aligned}(\xi_{i,k} - l_{i,k})^2 &\approx (\xi_{i,k}^t - l_{i,k}^t)^2 \\ &\quad + \begin{bmatrix}2\xi_{i,k}^t - 2l_{i,k}^t \\ 2l_{i,k}^t - 2\xi_{i,k}^t\end{bmatrix}^T \begin{bmatrix}\xi_{i,k} - \xi_{i,k}^t \\ l_{i,k} - l_{i,k}^t\end{bmatrix}\end{aligned}$$

(37)

Then, we further reformulate the approximated equation as the constraints shown in (16) with a cone expression. Moreover, constraint (15f) is approximated by constraint (18) in a similar way. Constraints (15e) and (15h) are approximated by (17) and (19) respectively by first order Taylor expansion to obtain the lower bound on the squared norm and the subtracted term, respectively.

All the approximated constraints (16)–(19) are stricter than their original counterparts, guaranteeing that the solution of the approximated problem is strictly smaller than the original optimum. For example, consider the optimal $\xi_{i,k}$ and $l_{i,k}$ obtained by solving the approximated problem, which is denoted by $\xi_{i,k}^a$ and $l_{i,k}^a$. These two variables are bounded by constraint (16) in the approximated problem. Comparing (16) with the original constraint (15b) and considering the property of the Taylor expansion, we have $\xi_{i,k}^a l_{i,k}^a + \Delta_{approx} \le \delta_{i,k}P$, where $\Delta_{approx} \ge 0$. Thus,

$$\frac{B\Delta}{N}\log\left(1 + \xi_{i,k}^a\right) \le \frac{B\Delta}{N}\log\left(1 + \frac{\delta_{i,k}P}{l_{i,k}^a}\right)$$

(38)

Moreover, due to $l_{i,k}^{a} \geq 1/g_{i,k}$, we have

$$\frac{B\Delta}{N}\log\left(1+\frac{\delta_{i,k}P}{l_{i,k}^{a}}\right) \leq R_{i,k}\left(\delta_{i,k},\mathbf{Q}_{k}\right).$$

(39)

Therefore, the approximation on constraint (15b) will leads to $\check{R}_{i,k}^{*} < R_{i,k}(\delta_{i,k},\mathbf{Q}_{k})$. Other approximated constraints can be proven similarly to show that the proposed approximated objective function provides the global lower bound for original objective function (14). Moreover, due to the gradient consistency in the first order estimation, the SCA algorithm will be stopped when a local optimizer is found.

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, Oct.–Dec. 2017.

2. J. Gao, L. Zhao and X. Shen, "Service offloading in terrestrial-satellite systems: User preference and network utility", *Proc. IEEE Global Commun. Conf.*, pp. 1-6, Dec. 2019.

3. Y. Wu, B. Shi, L. P. Qian, F. Hou, J. Cai and X. Shen, "Energy-efficient multi-task multi-access computation offloading via NOMA transmission for IoTs", *IEEE Trans. Ind. Informat.*.

4. S. Fu, L. Zhao, X. Ling and H. Zhang, "Maximizing the system energy efficiency in the blockchain based Internet of Things", *Proc. IEEE Int. Conf. Commun.*, pp. 1-6, May 2019.

5. N. Mohamed, J. Al-Jaroodi, I. Jawhar, H. Noura and S. Mahmoud, "UAVFog: A UAV-based fog computing for Internet of Things", *Proc. IEEE SmartWorld Ubiquitous Intell. Comput. Adv. Trusted Computed Scalable Comput. Commun. Cloud Big Data Comput. Internet People Smart City Innov*, pp. 1-8, Aug. 2017.

6. W. Z. Khan, M. Y. Aalsalem, M. K. Khan, M. S. Hossain and M. Atiquzzaman, "A reliable Internet of Things based architecture for oil and gas industry", *Proc. 19th Int. Conf. Adv. Commun. Technol.*, pp. 705-710, Feb. 2017.

7. M. C. Domingo, "An overview of the internet of underwater things", *J. Netw. Comput. Appl.*, vol. 35, no. 6, pp. 1879-1890, 2012.

8. T. Samad, J. S. Bay and D. Godbole, "Network-centric systems for military operations in urban terrain: The role of UAVs", *Proc. IEEE*, vol. 95, no. 1, pp. 92-107, Jan. 2007.

9. S. Fu, L. Zhao, Z. Su and X. Jian, "UAV based relay for wireless sensor networks in 5G systems", *Sensors*, vol. 18, 2018.

10. Y. Zhou, N. Cheng, N. Lu and X. Shen, "Multi-UAV-aided networks: Aerial-ground cooperative vehicular networking architecture", *IEEE Veh. Technol. Mag.*, vol. 10, no. 4, pp. 36-44, Dec. 2015.

11. W. Shi et al., "Multi-drone 3-D trajectory planning and scheduling in drone-assisted radio access networks", *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 8145-8158, Aug. 2019.

12. 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.

13. Q. Hu, Y. Cai, G. Yu, Z. Qin, M. Zhao and G. Y. Li, "Joint offloading and trajectory design for UAV-enabled mobile edge computing systems", *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1879-1892, Apr. 2019.

14. Y. C. Hu, M. Patel, D. Sabella, N. Sprecher and V. Young, "Mobile edge computing—A key technology towards 5G", 2014.

15. K. Zhang et al., "Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks", *IEEE Access*, vol. 4, pp. 5896-5907, 2016.

16. Y. Mao, J. Zhang and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices", *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590-3605, Dec. 2016.

17. Z. Kuang, L. Li, J. Gao, L. Zhao and A. Liu, "Partial offloading scheduling and power allocation for mobile edge computing systems", *IEEE Internet Things J.*, vol. 6, no. 4, pp. 6774-6785, Aug. 2019.
18. T. G. Rodrigues, K. Suto, H. Nishiyama, N. Kato and K. Temma, "Cloudlets activation scheme for scalable mobile edge computing with transmission power control and virtual machine migration", *IEEE Trans. Comput.*, vol. 67, no. 9, pp. 1287-1300, Sep. 2018.
19. T. G. Rodrigues, K. Suto, H. Nishiyama and N. Kato, "Hybrid method for minimizing service delay in edge cloud computing through VM migration and transmission power control", *IEEE Trans. Comput.*, vol. 66, no. 5, pp. 810-819, May 2017.
20. Q. Wu, Y. Zeng and R. Zhang, "Joint trajectory and communication design for multi-UAV enabled wireless networks", *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 2109-2121, Mar. 2018.
21. Y. Zeng and R. Zhang, "Energy-efficient UAV communication with trajectory optimization", *IEEE Trans. Wireless Commun.*, vol. 16, no. 6, pp. 3747-3760, Jun. 2017.
22. F. Tang, Z. M. Fadlullah, N. Kato, F. Ono and R. Miura, "AC-POCA: Anticoordination game based partially overlapping channels assignment in combined UAV and D2D-based networks", *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1672-1683, Feb. 2018.
23. S. Garg, A. Singh, S. Batra, N. Kumar and L. T. Yang, "UAV-empowered edge computing environment for cyber-threat detection in smart vehicles", *IEEE Net.*, vol. 32, no. 3, pp. 42-51, May 2018.
24. M. Messous, H. Sedjelmaci, N. Houari and S. Senouci, "Computation offloading game for an UAV network in mobile edge computing", *Proc. IEEE Int. Conf. Commun.*, pp. 1-6, May 2017.
25. S. Jeong, O. Simeone and J. Kang, "Mobile edge computing via a UAV-mounted cloudlet: Optimization of bit allocation and path planning", *IEEE Trans. Veh. Technol.*, vol. 67, no. 3, pp. 2049-2063, Mar. 2018.
26. F. Tang, Z. M. Fadlullah, B. Mao, N. Kato, F. Ono and R. Miura, "On a novel adaptive UAV-mounted cloudlet-aided recommendation system for LBSNs", *IEEE Trans. Emerg. Topics Comput.*, vol. 7, no. 4, pp. 565-577, Oct.–Dec. 2019.
27. 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.
28. Y. Wang, M. Sheng, X. Wang, L. Wang and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling", *IEEE Trans. Commun.*, vol. 64, no. 10, pp. 4268-4282, Oct. 2016.
29. F. Wang, J. Xu, X. Wang and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems", *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 1784-1797, Mar. 2018.
30. W. Yuan and K. Nahrstedt, "Energy-efficient soft real-time CPU scheduling for mobile multimedia systems", *SIGOPS Oper. Syst. Rev.*, vol. 37, no. 5, pp. 149-163, Oct. 2003.
31. H. Li, K. Ota and M. Dong, "Learning IoT in edge: Deep learning for the Internet of Things with edge computing", *IEEE Netw.*, vol. 32, no. 1, pp. 96-101, Jan. 2018.
32. T. Lipp and S. Boyd, "Variations and extension of the convex–concave procedure", *Optim. Eng.*, vol. 17, no. 2, pp. 263-287, 2016.
33. W. Dinkelbach, "On nonlinear fractional programming", *Manage. Sci.*, vol. 13, no. 7, pp. 492-498, 1967.
34. S. Boyd, N. Parikh, E. Chu, B. Peleato and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers", *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1-122, Jan. 2011.
35. W. Deng, M. Lai, Z. Peng and W. Yin, "Parallel multi-block ADMM with O(1/k) convergence", *J. Scientific Comput.*, vol. 71, no. 2, pp. 712-736, 2017.
36. K. Wang, A. M. So, T. Chang, W. Ma and C. Chi, "Outage constrained robust transmit optimization for multiuser MISO downlinks: Tractable approximations by conic optimization", *IEEE Trans. Signal Process.*, vol. 62, no. 21, pp. 5690-5705, Nov. 2014.
37. A. Hakiri, P. Berthou, A. Gokhale and S. Abdellatif, "Publish/subscribe-enabled software defined networking for efficient and scalable IoT communications", *IEEE Commun. Mag.*, vol. 53, no. 9, pp. 48-54, Sep. 2015.

38. R. Cao, A. Cuevas and W. G. Manteiga, "A comparative study of several smoothing methods in density estimation", *Comput. Statist. Data Anal.*, vol. 17, no. 2, pp. 153-176, 1994.

39. M. Mozaffari, A. T. Z. Kasgari, W. Saad, M. Bennis and M. Debbah, "Beyond 5G with UAVs: Foundations of a 3D wireless cellular network", *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 357-372, Jan. 2019.

Footnotes

We deploy the fixed-wing UAV in the proposed system as an example. The proposed approach also can be adapted to the system with a quad-rotor UAV, where only the mechanical energy consumption model is different.