Distributed Computation Offloading and Trajectory Optimization in Multi-UAV-Enabled Edge Computing

Xiangyi Chen, Yuanguo Bi, *Member, IEEE,* Guangjie Han, *Senior Member, IEEE,* Dongyu Zhang, Minghan Liu, Han Shi, Hai Zhao, and Fengyun Li

***Abstract*—The Internet of Things (IoT) technology has expanded network space by inter-connected devices, which has been widely used in various fields, such as environmental monitoring, object tracking, risk warning, etc. Due to insufficient computing capacity, limited battery life, and unreliable communication environment in IoT, Unmanned Aerial Vehicle (UAV) -enabled edge computing has been recently utilized to provide enhanced coverage and efficient computational support in the scenarios with sparse or unreliable ground infrastructure, such as disaster rescue, emergency response, military fields, etc. However, UAV-enabled edge computing faces many challenges, such as low offloading efficiency, high energy consumption, high complexity, etc. In this paper, a distributed computation offloading scheme is proposed to provide computational support to large-scale IoT nodes and optimize the energy efficiency of multiple UAVs. Firstly, to provide accurate and efficient computational support, a real-time intelligent positioning algorithm is designed to obtain the precise location information of IoT nodes. Then, a distributed computation offloading and path planning algorithm is presented, which jointly optimizes the computation offloading of large-scale IoT nodes and trajectory planning of multiple UAVs to reduce the energy consumption of UAVs. Furthermore, we develop a closed-form theoretical analysis model to demonstrate that the algorithm enables a performance guarantee related to energy efficiency. Finally, extensive simulations have been conducted and show that the proposed scheme can greatly improve the system utility and energy efficiency.**

***Index Terms*—Edge computing, Computation offloading, Trajectory optimization, Internet of Things.**

Manuscript received October 26, 2021; revised January 24, 2022; accepted April 28, 2022; date of current version May 10, 2022. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFE0125300 and Grant 2019JSJ12ZDYF01, in part by the National Natural Science Foundation of China under Grant U1808207 and Grant 62171113, in part by the Fundamental Research Funds for the Central Universities of China under Grant N2116002, Grant 2020GFZD014, and Grant N2124006-1. (*Corresponding author: Yuanguo Bi.*)

Xiangyi Chen, Yuanguo Bi, Dongyu Zhang, Minghan Liu, Han Shi, Hai Zhao, and Fengyun Li are with the School of Computer Science and Engineering, Northeastern University, Shenyang 110169, China, and also with the Engineering Research Center of Security Technology of Complex Network System, Ministry of Education, Shenyang 110169, China (e-mail: xiangyichen3746@gmail.com, biyuanguo@mail.neu.edu.cn, zdy116116@163.com, lmh248944264@163.com, HanShi.edu@outlook.com, haizhao310@163.com, lifengyun@mail.neu.edu.cn).

Guangjie Han is with the Department of Information and Communication Systems, Hohai University, Changzhou 213022, and State Key Laboratory of Acoustics, Institute of Acoustics, Chinese Academy of Sciences, Beijing 100190, China (e-mail: hanguangjie@gmail.com).

I. Introduction

DRIVEN by the Internet of Things (IoT) technology, more and more devices have connected to the Internet to support various IoT applications, such as environmental monitoring, object tracking, risk warning, etc [1]. Cisco estimates that 12.3 billion mobile devices will connect to the Internet by 2022, forming a huge network [2]. The expansion of IoT will result in a substantial increasing number of task requests and a large volume of data, which imposes great pressure on the processing capacity of the existing IoT network [3], [4].

On the one hand, IoT nodes face the issues of insufficient computing capacity, limited battery life, poor computing compatibility, etc. For example, some IoT nodes may need to execute tasks beyond their computing capacities, and the life spans of the IoT nodes deployed in the outdoor environment are greatly limited by the carried batteries, which forces the IoT nodes to offload these tasks for extending their life spans. On the other hand, most of these tasks are computation-intensive and delay-sensitive. Offloading these tasks to a cloud center may face huge challenges [5], [6]. For example, transmitting all task data to the cloud center for processing will consume a lot of communication resources (backhaul links usage) and bring a high end-to-end delay, which greatly wastes network resources and degrades the quality of IoT services [7].

To handle large volume and widely distributed data, the emerging Mobile Edge Computing (MEC) extends the computing and storage capacities to the network edge, which reduces the non-negligible delay caused by data transmissions [8]–[14]. Furthermore, Unmanned Aerial Vehicle-mounted Edge Servers (UAV-ES) have been utilized as promising approaches to provide enhanced coverage and efficient computational support in the areas with sparse or unreliable ground infrastructure, such as disaster rescue, emergency response, and military fields [15], [16]. For example, some IoT nodes are deployed in unattended or even hostile environments to execute tasks including detecting dangerous terrain, monitoring natural resources, executing danger warning, etc [17], [18]. In these scenarios, the ground infrastructure is sparsely distributed, and the computational support based on fixed base stations is infeasible. Therefore, UAV-enabled edge computing has emerged as a promising technique to provide dynamic and efficient computational support for IoT nodes, which improves the system flexibility and extends the operational life of the system [19], [20].

However, due to the limited onboard power of UAVs and the mobility of IoT nodes, UAV-enabled edge computing faces many challenges, such as low offloading efficiency, high energy consumption, high complexity, etc. First, IoT nodes may need to constantly move to execute predetermined tasks, such as object tracking, and many low-cost IoT nodes cannot afford the cost of global navigation satellite systems receivers [21], [22]. To reduce the cost of IoT nodes and provide efficient computational support in complex terrain environments, it is necessary to build an intelligent positioning solution to obtain precise location information of IoT nodes. Second, due to the widespread distribution of large-scale IoT nodes, a single UAV can hardly cope with a large number of task requests. Furthermore, due to the lack of global information in multi-UAV-enabled edge computing, traditional centralized optimization solutions are not applicable [23]. Therefore, based on the local information, how to make computation offloading decisions for large-scale IoT nodes, and how to plan multiple UAV trajectories to improve the effectiveness of the system are still open issues.

To address the above issues, we study the online distributed computation offloading problem in the scenario of multiple UAVs and large-scale IoT nodes taking into account the mobility of IoT nodes and time-varying computing demands. First, we design a Three-dimensional Space Intelligent Positioning (TSIP) algorithm to obtain the mobility information of IoT nodes. Then, to cope with the complexity brought by large-scale IoT nodes and multiple UAVs, we propose a Distributed Computation Offloading and Path planning (DCOP) algorithm to perform computation offloading of IoT nodes and trajectory optimization of multiple UAVs. Our goal is to maximize the system utility and minimize the UAVs energy consumption. The contributions of this paper are four-fold, which are summarized as follows.

1) An intelligent positioning algorithm TSIP is designed to obtain the real-time location information of IoT nodes, which is utilized to assist in computation offloading and trajectory planning. Furthermore, we analyze the positioning error of TSIP in the process of anchor node selection.
2) A distributed computation offloading algorithm DCOP is proposed, which performs the computing offloading of IoT nodes and path planning of UAVs. The maximum utility coverage set for each UAV is constructed through communications and consensus agreements among IoT nodes to improve energy efficiency.
3) We analyze the performance of DCOP based on the submodularity of the gain function, and prove that DCOP enables a performance guarantee related to energy efficiency through a closed-form theoretical analysis.
4) Finally, extensive simulations are conducted to evaluate the performance of our proposed algorithm, which show that our algorithm effectively improves the system utility and significantly reduces the energy consumption of UAVs.

The rest of the paper is organized as follows. In Section II, we briefly review the related works. We introduce the system model and give the problem formulation in Section III. In Section IV, TSIP and DCOP are proposed, and a closed-form theoretical model is developed to analyze the performance of DCOP in Section V. We evaluate the performance of the proposed algorithms through extensive simulations in Section VI, followed by concluding remarks in Section VII.

II. Related Work

In recent years, computation offloading in MEC has attracted widespread attention from researchers, and some proposals have been presented for various research fields. Specifically, according to the mobility of users and edge servers, the related works can be divided into three categories: i) Static optimization without considering mobility [24]–[28]; ii) Considering the mobility of users [29]–[33]; iii) Considering the mobility of edge servers [16], [34]–[37]. In MEC, mobile edge servers (e.g., vehicle-mounted edge servers or UAV-ES) may move to users and proactively provide computational support.

The optimization researches of computation offloading in MEC under static situations have become more and more mature, which mainly include computation offloading and joint optimization of computation offloading and service deployment [24]–[27]. For instance, Li *et al.* [24] proposed an incentive-aware offloading framework to reduce cost by offloading jobs to MEC servers. Poularakis *et al.* [27] studied the joint optimization of request routing and service deployment in dense MEC networks to minimize the load of the centralized cloud. In addition, some related works investigated user mobility and proposed solutions for computation offloading in MEC networks. Specifically, Ouyang *et al.* [32] proposed a mobility-aware framework based on Lyapunov optimization, achieving a balance between system performance and operating cost for MEC networks. Considering the mobility of users, Wang *et al.* [33] presented a dynamic computation coordination method to minimize the overall service delay. These works aim to offload computing tasks to static edge servers that are permanently deployed, without taking into account some scenarios with sparse or unreliable ground infrastructure.

Due to the limitations of static edge mechanisms in environments where the ground communication infrastructure is sparsely deployed and the communications are unreliable, the mobile edge mechanism has become an urgent need and attracted considerable attention from researchers. In order to provide computational support to end users, Liu *et al.* [34] studied the joint optimization problem of path planning and resources allocation in vehicle-mounted edge computing, which was formulated as a mixed integer nonlinear program. They proposed an algorithm based on piecewise linear approximation and linear relaxation to solve this problem, and improved the task completion rate of IoT nodes. In the UAV-ES situation, Guo *et al.* [35] studied the joint optimization of computation offloading and trajectory of one UAV in MEC to minimize the total delay of all users. Jeong *et al.* [36] jointly optimized the bit allocation for communications and UAV trajectory to reduce a UAV energy consumption. Another related work [37] focused on user association, user uploading

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3175050, IEEE Internet of Things Journal

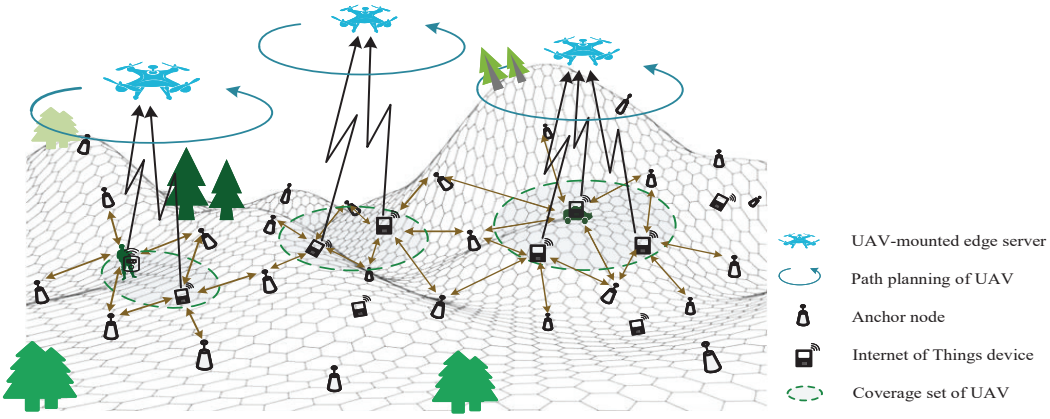


Fig. 1. The architecture of distributed computation offloading and trajectory optimization in multi-UAV-enabled edge computing for Internet of Things.

power, and UAV trajectory to maximize the total bits offloaded to a UAV from all users. In [16], Li *et al.* studied the issues of UAV trajectory, users transmit power and computation load allocation to improve the energy efficiency of a UAV. The above-mentioned approaches assume that the locations of users are fixed or pre-known, and they have not fully taken into account the complexity of the existing multi-UAV system [16], [37].

Although some UAV-assisted MEC solutions have been proposed, there are still some open issues to be addressed. Specifically, most related works assume that the locations of users are fixed or pre-known, and have not fully considered how to obtain users' locations accurately in different time slots. Some works study the optimization of UAV-assisted MEC networks based on the global information that is difficult to be obtained. In addition, they mainly focus on single UAV, which is not suitable for large-scale IoT scenarios. In a complex UAVs system, optimization research on multiple UAVs considering time-varying computing demand and user mobility needs further investigation. This paper proposes a real-time positioning algorithm TSIP in three-dimensional space to obtain the location information of IoT nodes, and designs a distributed computation offloading algorithm DCOP based on local information to optimize system energy efficiency.

III. System Model

A. System Architecture

The system architecture is shown in Fig. 1, which includes the ground layer and the edge layer. The edge layer contains multiple UAVs that constantly move and provide computational support to IoT nodes nearby. In the ground layer, multiple fixed anchor nodes for IoT node positioning are deployed in the three-dimensional space. According to different task demands (such as threat detection, object tracking, etc.), IoT nodes constantly move and generate task requests. Due to the computing capacity and power constraints of IoT nodes, task requests need to be offloaded to the UAVs for execution as much as possible. Otherwise, they can only be executed locally. Therefore, we need to jointly determine the computation offloading decision of each IoT node and the trajectory of each UAV to maximize the energy efficiency of the system.

UAV is denoted by $v_j \in V$, indexed by $j \in \{1, 2, 3, ..., |V|\}$, anchor node is denoted by $r_m \in R$, indexed by $m \in \{1, 2, 3, ..., |R|\}$, and IoT node is denoted by $u_i \in U$, indexed by $i \in \{1, 2, 3, ..., |U|\}$. In order to analyze the dynamic changes of the system, we model the system based on a quasi-static network scenario [34], [38], [39], and discretize the system time into multiple time slots, $t \in T$, indexed by $\{1, 2, 3, ..., |T|\}$. The length of each time slot is ς. The notations and variables used in our formulations are shown in Table I.

B. Computation Model

Denote the computation task model as $s_i^k(t) = (\rho_i^k(t), \sigma_i^k(t), w_i^k(t))$, indexed by $k \in \{1, 2, \cdots, |K|\}$, which respectively represent task type, task size of input-data (in bits), and computational intensity of the task, that is the required CPU frequency (in CPU cycles per bit). $s_i^k(t) \in S(t)$, where $S(t)$ is the set of tasks in time slot t. Denote the offloading decision variable of IoT node u_i in time slot t as a binary variable $x_{ij}(t)$, and $x_{ij}(t) = 1$ means that the node u_i offloads the task to UAV v_j for execution, otherwise $x_{ij}(t) = 0$. If all $x_{ij}(t) = 0, \forall v_j \in V$, then we set $x_{i\phi}(t) = 1$ to indicate that the task is executed locally.

Each edge server is equipped with multiple CPU cores. It can provide parallel computation for multiple IoT tasks simultaneously by processor sharing [40]. Therefore, if UAV v_j moves to the location of IoT node u_i, it can provide computational support to a user set Π_{ij} which is centered on u_i with radius d_j, where d_j is the coverage radius of v_j. Note that the maximum CPU cycle frequency for UAV v_j to perform tasks is $f_j^{\max}$.

C. Communication Model

Since the communication efficiency is closely related to the locations of IoT nodes and UAVs, we consider both the

mobility of IoT nodes and UAVs in the research architecture. To represent the locations of IoT nodes and UAVs, we build a three-dimensional cartesian coordinate system. The location of IoT node u_i in time slot t is $p_i(t) = (u_{i,x}(t), u_{i,y}(t), u_{i,z}(t))$, and the fixed location of anchor node r_m is $p_m = (r_{m,x}, r_{m,y}, r_{m,z})$. Similarly, the location of UAV v_j in time slot t is $p_j(t) = (v_{j,x}(t), v_{j,y}(t), v_{j,h}(t))$. Therefore, the average velocity vector of v_j in time slot t is expressed as [16]

$$\mathbf{v_j}(t) = \frac{p_j(t) - p_j(t-1)}{\varsigma}. \tag{1}$$

Similarly, the average acceleration vector of v_j in time slot t is calculated as

$$\mathbf{a_j}(t) = \frac{\mathbf{v_j}(t) - \mathbf{v_j}(t-1)}{\varsigma}. \tag{2}$$

And the path P_j of v_j can be denoted as

$$P_j = \{p_j(t) \,|t \in T\}. \tag{3}$$

The channel between UAV and IoT node is considered as line-of-sight (LoS) [37], [41]. According to a free-space path loss model [16], [42], the channel power gain between v_j and u_i in time slot t is calculated by

$$h_{ij}(t) = \frac{\rho_0}{\|p_j(t) - p_i(t)\|^2}, \tag{4}$$

where $\|*\|$ is the notation representing the Euclidean norm, ρ_0 is the received channel power gain at the reference distance $d_0 = 1$ m between v_j and u_i. In non-orthogonal channel access model [16], the transmission rate of IoT node u_i in communication with UAV v_j in time slot t is calculated by [43], [44]

$$R_{ij}(t) = W \log_2 \left[1 + \frac{\beta_i h_{ij}(t)}{\sum_{i' \neq i} \beta_{i'} h_{i'j}(t) + \sigma^2}\right], \tag{5}$$

where W is the bandwidth, β_i is the transmission power of u_i, $\sum_{i' \neq i} \beta_{i'} h_{i'j}(t)$ is the signal interference, and σ^2 is the noise power. Therefore, in time slot t, the size of data offloaded from u_i to v_j is given by

$$\delta_{ij}^k(t) = \min\left[x_{ij}(t) R_{ij}(t) \varsigma, \sigma_i^k(t)\right]. \tag{6}$$

The size of data offloaded is $\sigma_i^k(t)$ if the task data can be entirely offloaded in time slot t. Otherwise, (6) calculates the amount of task data that can be offloaded.

D. Energy Consumption Model

1) UAV Energy Consumption Model: For UAV propulsion energy consumption, we adopt an improved UAV propulsion energy consumption model [36], [45]. The propulsion energy consumption of UAV v_j in time slot t is related to its instantaneous acceleration and velocity, which can be calculated as

$$e_j^f(t) = \eta_1 \|\mathbf{v_j}(t)\|^3 + \frac{\eta_2}{\|\mathbf{v_j}(t)\|}\left(1 + \frac{\|\mathbf{a_j}(t)\|^2}{g^2}\right), \tag{7}$$

where η_1 and η_2 are fixed parameters mainly related to UAV's weight and wing areas, and g represents the gravitational acceleration [16], [36], [45].

TABLE I
COMMONLY USED NOTATIONS AND VARIABLES

Notation	Description
v_j,V	UAV and the set of UAVs
u_i,U	IoT node and the set of IoT nodes
r_m,R	Anchor node and the set of anchor nodes
t,T,ς	Time slot, the set of time slots, and the length of each time slot
$s_i^k(t)$,$S(t)$	Computation task of u_i and the set of tasks in time slot t
$\sigma_i^k(t)$	Task size of input-data of s_i^k
$w_i^k(t)$	Task computational intensity of s_i^k
$x_{ij}(t)$	Binary variable indicates whether the task s_i^k of IoT node u_i is offloaded to UAV v_j in time slot t
$x_{i\phi}(t)$	Binary variable indicates whether the task s_i^k of IoT node u_i is executed locally in time slot t
d_j	Coverage radius of UAV v_j
$f_j^{\max}$	Maximum CPU cycle frequency of UAV v_j
$\varpi_{ij}^k(t)$	Execution frequency of task s_i^k
$p_i(t)$	Location of IoT node u_i in time slot t
$p_j(t)$	Location of UAV v_j in time slot t
p_m	Location of anchor node r_m
$\mathbf{v_j}(t)$,$\mathbf{a_j}(t)$	Velocity and acceleration of UAV v_j in time slot t
P_j	Path planning of UAV v_j
$R_{ij}(t)$	Transmission rate of IoT node u_i offloading task to UAV v_j
$e_j^f(t)$	Propulsion energy consumption of UAV v_j in time slot t
$e_{ij}^k(t)$	Computational energy consumption of task s_i^k
$e_m(t)$	Energy of anchor node r_m in time slot t
$\delta_{ij}^k(t)$	Size of data offloaded from u_i to v_j in time slot t
$\Re$, $\hat{e}$	Communication range and energy threshold of anchor node
ϑ_i, Θ_i	Candidate positioning unit and the set of positioning unit of IoT node u_i
ξ_{ij}^k	Scoring function of UAV v_j processing the task s_i^k
$U_j(t)$	User set in the coverage of UAV v_j in time slot t
$U_{ij}(t)$	User set in the coverage of UAV v_j centered on u_i in time slot t
$\Pi_j^{\max}(t)$	Maximum utility coverage set of v_j in time slot t
$\Pi_{ij}(t)$	Utility coverage set of v_j centered on u_i in time slot t

The computational energy consumption of a UAV is mainly related to the amount of data offloaded and the frequency of actual execution. In time slot t, the computational energy consumption of v_j in executing the task s_i^k for u_i can be calculated as

$$e_{ij}^k(t) = \kappa_j \delta_{ij}^k(t) w_i^k(t) (\varpi_{ij}^k(t))^2, \tag{8}$$

where κ_j is the effective switched capacitance of UAV v_j processor [36], [37], and $\varpi_{ij}^k(t) = \frac{\delta_{ij}^k(t) w_i^k(t)}{\varsigma}$ is the actual CPU execution frequency of task s_i^k. Since the energy consumption of a UAV caused by data transmissions is smaller than those of flight and computation, we ignore the UAV energy consumption due to communications [43].

2) Anchor Energy Consumption Model: We define the initial battery energy of anchor node r_m as E_m, and the battery energy at the beginning of time slot t is expressed as $e_m(t)$, where $e_m(1) = E_m$. Therefore, the energy of anchor node r_m in time slot $t+1$ is expressed as

$$e_m(t+1) = e_m(t) - \sum_{a=1}^{A} e_m^a(t), \tag{9}$$

where A represents the number of times that anchor node r_m is selected as the positioning unit in time slot t, and $\sum_{a=1}^{A} e_m^a(t)$ is its total energy consumed in positioning IoT nodes in time slot t.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3175050, IEEE Internet of Things Journal

E. Problem Formulation

UAVs providing computational support to IoT nodes can bring utility to the system. Meanwhile, the movement and task execution of a UAV will incur energy cost. Our goal is to maximize the system utility and minimize the energy consumption cost. This process involves two essential variables, including IoT node offloading decision $x_{ij}(t)$ and UAV path planning $p_j(t)$. We define the energy efficiency of UAV v_j as the ratio of the amount of task data offloaded to the consumed energy, which can be denoted as

$$z_j(t) = \frac{\sum\limits_{u_i \in U} \delta_{ij}^k(x_{ij}(t))}{e_j^f(p_j(t)) + \sum\limits_{u_i \in U} e_{ij}^k(t)}, \tag{10}$$

where δ_{ij}^k is the amount of data offloaded from u_i to v_j and represents the system utility of v_j processing the task s_i^k for u_i. The amount of offloaded data indicates the execution efficiency of UAVs providing computational support for IoT nodes. e_j^f is the propulsion energy consumption of v_j, and e_{ij}^k is the computational energy consumption.

Offloading decision variable x_{ij} needs to satisfy the following constraint

$$x_{ij}(t) \in \{0,1\}, \forall s_i^k(t) \in S(t), v_j \in V. \tag{11}$$

The task data of an IoT node can only be offloaded to one UAV for execution, and we have

$$\sum_{v_j \in V} x_{ij}(t) \leq 1, \forall s_i^k(t) \in S(t). \tag{12}$$

For UAV v_j, the maximum computational resource constraint needs to be satisfied, that is

$$\sum_{u_i \in U} x_{ij}(t)\varpi_{ij}^k(t) \leq f_j^{\max}, \forall v_j \in V. \tag{13}$$

Denote $\tilde{U}_j(t)$ as the user set in the coverage of UAV v_j that is located at the position $p_j(t)$ in time slot t. If u_i does not belong to $\tilde{U}_j(t)$, its tasks cannot be offloaded to v_j, that is, the offloading decision satisfies

$$x_{ij}(t) = 0, \forall s_i^k(t) \in S(t), v_j \in V, u_i(t) \notin \tilde{U}_j(t). \tag{14}$$

In the process of positioning, the positioning unit needs to be constructed by selecting multiple anchor nodes. The anchor nodes in the positioning unit need to satisfy

$$d_m^i \leq \Re, \forall r_m \in R, \tag{15}$$

$$e_m \geq \hat{e}, \forall r_m \in R, \tag{16}$$

where d_m^i is the distance from r_m to u_i, $\Re$ and $\hat{e}$ are the communication range and energy threshold constraints, respectively.

Therefore, our optimization objective is to maximize the total energy efficiency of the system that is expressed as

$$\max_{\{x_{ij}(t)\},\{p_j(t)\}} \sum_{t \in T} \sum_{v_j \in V} z_j(t), \quad \text{s.t. (11) - (16) .} \tag{17}$$

There are two key issues to be solved in the afore-mentioned model. On the one hand, due to the uncertainty of IoT nodes mobility, it is necessary to design a real-time positioning algorithm to obtain the position information of IoT nodes in time to provide accurate and efficient computational support. On the other hand, the joint decisions of computation offloading for distributed large-scale IoT nodes and path planning for multiple UAVs have high computational complexity. The system lacks centralized management, and traditional optimization solutions based on global information are not applicable. In order to solve the above problems, we first design a positioning algorithm TSIP, and then propose a distributed edge computing algorithm DCOP, which will be introduced in detail in Section IV.

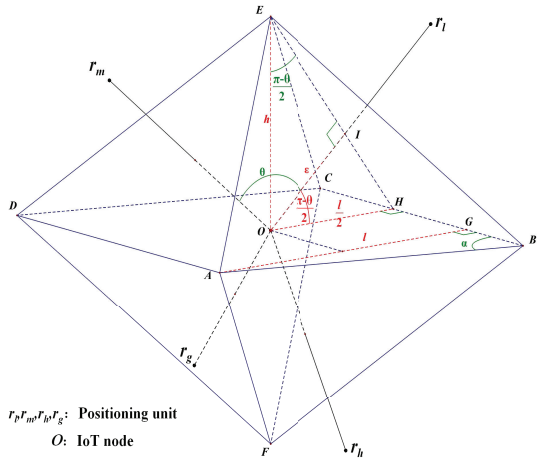


Fig. 2. Illustration of positioning error analysis.

IV. Multi-UAV-Enabled Distributed Edge Computing

Taking into account the mobility of IoT nodes, we first propose a positioning algorithm TSIP, which reasonably constructs the positioning units and corrects the locations of IoT nodes according to the proposed anchor node selection theorem in Section IV-A. Then, to deal with the complexity of the system caused by multiple IoT nodes and UAVs, we propose a distributed algorithm DCOP to construct the utility coverage set for UAVs and provide computational support to multiple IoT nodes to improve the energy efficiency of the system.

A. Three-dimensional Space Intelligent Positioning

In the three-dimensional space, the location coordinates of IoT node u_i can be calculated by introducing 4 non-coplanar anchor nodes and measuring the distance d_m^i from the anchor node r_m to IoT node u_i. However, in actual situations, there may be errors in the distance measurements, which affect the accuracy of positioning.

In order to enhance the positioning accuracy of IoT nodes, we need to choose more reliable anchor nodes in constructing the positioning units. Our proposed Theorem 1 and Corollary 1 provide theoretical guidance for constructing positioning units.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3175050, IEEE Internet of Things Journal

Algorithm 1: Three-dimensional space intelligent positioning

Input: $R=\{r_1,r_2,...,r_m,...\}$, $r_m=\{p_m,e_m,d_m^i\}$, $\Re$, $\hat{\theta}$, $\hat{\varepsilon}$, $\hat{e}$

Get the task request $S(t)$ and the distance information d_m^i in time slot t;
for $s_i^k(t)\in S(t)$ **do**
  if $d_m^i\le\Re$ *and* $e_m\ge\hat{e}$ **then**
    $\Psi_i\leftarrow\Psi_i\cup\{r_m\}$;
  end
  do
    Select candidate positioning units $\vartheta_i=(r_m,r_l,r_h,r_g)$ from Ψ_i;
    Solve the angle θ among any two anchor nodes and IoT node;
    if $|\theta-109.4736^\circ|<\hat{\theta}$ **then**
      $\Theta_i\leftarrow\Theta_i\cup\vartheta_i$;
      $p_i(n)=(u_{i,x},u_{i,y},u_{i,z})$;
      $\bar{p}_i(n)=\frac{\sum_{\vartheta\in\Theta_i}p_i(n)}{|\Theta_i|}$;
    end
  while $\|\bar{p}_i(n)-\bar{p}_i(n-1)\|\le\hat{\varepsilon}$;
  $p_i(t)=\bar{p}_i(n)$;
end
$e_m(t+1)=e_m(t)-\sum_{a=1}^{A}e_m^a(t)$;

Output: $p_i(t)=(u_{i,x},u_{i,y},u_{i,z})$

Theorem 1 ***Anchor Node Selection Theorem***: When the angle between any two anchor nodes in the positioning unit and an IoT node is equal to 109.4736°, the positioning solution achieves the minimum error.

Proof. In the process of positioning, denote the measurement error as ε, then the estimated distance interval of 4 anchor nodes $(d_m^i-\varepsilon,d_m^i+\varepsilon)$ can be enclosed in a space area surrounded by eight spherical surfaces, and the volume of the spatial domain can be as the positioning error interval of the IoT node. The spherical surfaces of the spatial domain are approximated as planes, forming an octahedron, as shown in Fig. 2. Denote $\angle r_lOr_m$ as θ, and $\angle ABC$ as α, then $\angle OEH$ is $\frac{\pi-\theta}{2}$. According to the geometric relationship and the symmetry of the octahedron, the volume can be calculated by

$$V=\frac{2}{3}S\cdot h=\frac{8}{3}\frac{\varepsilon^3}{\cos^2(\frac{\pi-\theta}{2})\sin(\frac{\pi-\theta}{2})\sin\alpha},\qquad(18)$$

where S is the area of the quadrilateral $ABCD$ and equals to $\frac{4\varepsilon^2}{\cos^2(\frac{\pi-\theta}{2})\sin\alpha}$, and h is equal to $\frac{\varepsilon}{\sin(\frac{\pi-\theta}{2})}$. In (18), to get the minimum value of V, $\sin\alpha$ gets the maximum value of 1 when α=$\frac{\pi}{2}$. On the other hand, for θ $(0<\theta<\pi)$, let $f(\theta)=\cos^2(\frac{\pi-\theta}{2})\sin(\frac{\pi-\theta}{2})$. We get its extreme point by derivation, and according to the domain of definition and properties of second derivative, when $\sin(\frac{\pi-\theta}{2})$=$\frac{\sqrt{3}}{3}$, that is, $\theta=\pi-2\arcsin(\frac{\sqrt{3}}{3})\approx109.4736^\circ$, the volume of octahedron reaches the minimum value $V_{\min}=4\sqrt{3}\varepsilon^2$. It means that the corresponding positioning error reaches the minimum value. Therefore, Theorem 1 is proved. □

Algorithm 2: Construction of utility coverage set

Input: $S(t)$, $f_j^{\max}$

Initialize $\Pi_{ij}(t)$, and get the set of covered users $\tilde{U}_{ij}$;
do
  Get the score of decision $x_{ij}^n(t)$ by $\xi_{ij}^{k,n}(t)$;
  Select u_i when $\arg\max_{u_i\in\tilde{U}_{ij}\backslash\Pi_{ij}^n(t)}\xi_{ij}^{k,n}(t)\frac{1}{\varpi_{ij}^n(t)}$;
  $\Pi_{ij}^n(t)\leftarrow\Pi_{ij}^n(t)\bigcup u_i$;
  $(X^*(t),P^*(t))\leftarrow(X^*(t),P^*(t))\bigcup(x_{ij}{}^*(t),p_j{}^*(t))$;
  $S^n(t)\leftarrow S^{n-1}(t)\backslash s_i^k(t)$;
  $f_j(t)=f_j(t)+\varpi_{ij}^n(t)$;
while $f_j(t)\le f_j^{\max}$ *and* $\tilde{U}_{ij}\backslash\Pi_{ij}^n(t)\neq\phi$;
$\xi_j^n(\Pi_{ij}^n(t))=\sum_{u_i\in\Pi_{ij}^n(t)}\xi_{ij}^{k,n}(t)$;
for $u_i\in U\backslash\Pi_{ij}^n(t)$ **do**
  $\xi_{ij}^{k,n}(x_{ij}^n(t))=0$;
end

Output: $\xi_j^n(\Pi_{ij}^n(t))$, $\Pi_{ij}^n(t)$, $(X^*(t),P^*(t))$

Corollary 1: In the process of three-dimensional space positioning, if the number of available positioning anchor nodes is more than 4, we can select any two anchor nodes with the angle between them and the IoT node approaching 109.4736° to construct the positioning unit, which can reduce positioning error.

Then we propose TSIP to obtain the location information of IoT nodes and the pseudo code is shown in Algorithm 1. Denote the set of anchor nodes as $R=\{r_1,r_2,...,r_m,...\}$, where each anchor node contains location, energy, and distance information, denoted as $r_m=\{p_m,e_m,d_m^i\}$. First, we select the available anchor nodes that satisfy the constraints of (15) and (16), and add them to the set of available anchor nodes Ψ_i. Our goal is to use anchor nodes with more residual energy to extend the service life of the system. TSIP randomly selects available anchor nodes from Ψ_i and constructs the positioning unit $\vartheta_i=(r_m,r_l,r_h,r_g)$ through the anchor node selection theorem, then adds them to the set of positioning unit Θ_i. And $\hat{\theta}$ is the threshold of the difference between the actual angle and the optimal angle. Next, TSIP calculates the position coordinate of IoT node $p_i(n)=(u_{i,x},u_{i,y},u_{i,z})$ by the positioning unit. Furthermore, since there may be a large error in the positioning by one positioning unit, the position coordinate $p_i(n)$ is corrected by the average value of multiple positionings. When the average error of the two positionings is less than the threshold $\hat{\varepsilon}$, the algorithm tends to converge and exits the positioning process. In addition, Theorem 1 provides a theoretical guarantee for our positioning algorithm in reducing positioning error.

B. Distributed Computation Offloading and Path Planning

The proposed DCOP mainly solves two key issues: offloading decision x_{ij} of IoT node u_i and path planning p_j of UAV v_j, $i\in\{1,2,3,...,|U|\}$ and $j\in\{1,2,3,...,|V|\}$.

Algorithm 3: Distributed computation offloading and path planning

Input: $p_j(t-1)$, $p_i(t)$, $S(t)$, $f_j^{\max}$

do
  if $s_i^k(t) \in S(t)$ **then**
    if $n \neq 0$ **then**
      $\xi_{ij}^{k,n}(t) = \xi_{ij}^{k,n-1}(t)$;
    end
    do
      $\tilde{\xi}_{ij}^{k}(t) = \xi_{ij}^{k,n}(t)$;
      if $\xi_{ij}^{k,n-1}(t) \neq 0$ *and* $\xi_{ij}^{k,n}(t) = 0$ **then**
        Update offloading decision $x_{ij}(t)$ to maximize energy efficiency, $\xi_{ij}^{k,n}(t) = gain(u_i, \Pi_{ij}^n(t), v_j)$;
      end
      if $x_{ij}^n(t) \neq x_{ij}^{n-1}(t)$ **then**
        $send(u_i, x_{ij}^n(t))$;
      end
      $recieve(u_{i'}, x_{i'j}^n(t))$;
      Construct utility coverage set with Algorithm 2;
    while $\tilde{\xi}_{ij}^{k}(t) \neq \xi_{ij}^{k,n}(t)$;
  end
  else
    $recieve(u_{i'}, x_{i'j}^n(t))$;
    Construct utility coverage set with Algorithm 2;
  end
  if $\exists u_i \in U$, $\exists \xi_j^n(\Pi_{ij}^n(t)) \neq \xi_j^{n-1}(\Pi_{ij}^{n-1}(t))$ **then**
    $send(u_i, \Pi_{ij}^n(t))$;
  end
  $receive(u_{i'}, \Pi_{i'j}^n(t))$, $\forall u_{i'} \in \tilde{U}_i$;
  Consensus agreement(u_i, $\Pi_{ij}^n(t)$), $\forall u_i \in \tilde{U}_i$, $\xi_j^n(\Pi_j^{\max}(t)) \leftarrow \max\{\xi_j^n(\Pi_{ij}^n(t)), \xi_j^n(\Pi_{i'j}^n(t))\}$;
while $\exists \Pi_{ij}^n(t) \neq \Pi_{i'j}^n(t)$;
Output: $(X^*(t), P^*(t))$, $\Pi_j^{\max}(t)$

That is, how to make decisions of p_j and x_{ij} to provide computational support to more IoT nodes, and minimize UAV energy consumption. Denote $\tilde{U}_{ij}(t)$ as the user set in the coverage of UAV v_j centered on IoT node u_i in time slot t. Due to processor sharing of UAV-ES, a UAV can process task requests from multiple IoT nodes simultaneously [40]. Therefore, we give the following definitions.

Definition 1 ***Utility Coverage Set***: The set where user tasks can be offloaded to UAV v_j from $\tilde{U}_{ij}(t)$ is the utility coverage set of v_j centered on u_i, expressed as $\Pi_{ij}(t)$.

The main idea of DCOP is that each IoT node makes offloading decision by constructing a utility coverage set centered on itself based on local information. This solves the problem of the lack of global information in multi-UAV-enabled edge computing. Specifically, each IoT node sequentially adds users to the utility coverage set according to the ratio of energy efficiency to cost to maximize the overall energy efficiency of the system. Then each UAV's trajectory is planned according to the decision of optimal energy efficiency of the system. As we mentioned, utility coverage set $\Pi_{ij}(t)$ means that the tasks of users in this set can be offloaded to v_j under the constraint of UAV resources if v_j flies to the location of u_i. Since there exist time-varying offloading requests from IoT nodes in the UAV-enabled edge computing system, we define the gain function ξ_{ij}^k of the system energy efficiency according to the law of diminishing marginal utility, which is more realistic than the simple linear utility function and used in cloud services such as Software as a Service (SAAS) [46]. Specifically, the gain function of adding u_i to the utility coverage set of v_j is defined as

$$\begin{aligned}\xi_{ij}^k(t) &= gain(u_i, \Pi_{ij}(t), v_j) \\ &= \min_{u \in \Pi_{ij}} \left\{ \frac{\delta_{ij}^k(x_{ij}(t))}{e_j^f(p_j(t)) + e_{ij}^k(t)}, \mathcal{L}_{ij}^u(t) \right\},\end{aligned} \tag{19}$$

where $u \in \Pi_{ij}$ means any user that has been added to the utility coverage set, and the gain function is related to energy efficiency. We introduce an auxiliary variable $\mathcal{L}_{ij}^u$ in (19) and it is defined as

$$\mathcal{L}_{ij}^u(t) = \begin{cases} +\infty, \text{if } u_i \text{ select } v_j \text{ at the first time,} \\ \varpi_{ij}^n(t) \dfrac{\xi_{uj}^k(t)}{\varpi_{uj}^n(t)}, \text{otherwise.} \end{cases} \tag{20}$$

where $\frac{\xi_{uj}^k(t)}{\varpi_{uj}^n(t)}$ represents the ratio of user u's gain to resource cost. In (20), the auxiliary variable ensures that the gain of the IoT node u_i added to the utility coverage set can not exceed the gain of any previously added IoT node, thus ensuring the submodularity of the gain function.

In general, the energy consumption of a UAV moving to the navigation target location of any IoT nodes in the utility coverage set $\Pi_{ij}(t)$ is approximately equal. Therefore, the IoT node's own energy efficiency is unchanged when constructing the utility coverage set. Based on the above gain function, we calculate the total gain function of $\Pi_{ij}(t)$ for UAV v_j by

$$\xi_j(\Pi_{ij}(t)) = \sum_{u_i \in \Pi_{ij}(t)} \xi_{ij}^k(t). \tag{21}$$

Next, we give the submodularity of the gain function, which is shown in Lemma 1.

Lemma 1 ***Submodularity of Gain Function***: The total gain function of UAV v_j, that is $\xi_j(\Pi_{ij}(t))$, is submodular with respect to user set $\Pi_{ij}(t)$.

Proof. Suppose $\Pi' \subset \Pi'' \subset U$, $u_i \notin \Pi''$, and $u_i \in U$, according to the definition of the gain function, we have

$$\min_{u \in \Pi''} \left\{ \varpi_{ij}^n(t) \frac{\xi_{uj}^{k,n}(t)}{\varpi_{uj}^n(t)} \right\} \leq \min_{u \in \Pi'} \left\{ \varpi_{ij}^n(t) \frac{\xi_{uj}^{k,n}(t)}{\varpi_{uj}^n(t)} \right\}. \tag{22}$$

Based on (19), (20) and (22), we have

$$gain(u_i, \Pi'', v_j) \leq gain(u_i, \Pi', v_j). \tag{23}$$

For the total gain of each UAV, according to the law of diminishing marginal utility, we have

$$\begin{aligned}\xi_j(\Pi'' + u_i) - \xi_j(\Pi'') &= gain(u_i, \Pi'', v_j) \\ &\leq gain(u_i, \Pi', v_j) = \xi_j(\Pi' + u_i) - \xi_j(\Pi'),\end{aligned} \tag{24}$$

where $\Pi'' + u_i$ means adding u_i to the set Π''. Therefore, according to the definition of submodular function [9], [47], [48], it can prove that the total utility function of each UAV is the submodular function with respect to user set. □

Definition 2 ***Maximum Utility Coverage Set***: The set corresponding to the maximum total gain of v_j, that is, the total gain is the maximum value in (21) for v_j, is defined as the maximum utility coverage set of v_j, expressed as $\Pi_j^{\max}(t)$.

If u_i is the central user of $\Pi_j^{\max}(t)$, which means v_j flying to its location (horizontal coordinate) can bring the maximum energy efficiency. Note that each user can be self-centered to build a utility coverage set for a UAV, but the selection and maintenance processes of the maximum utility coverage set require communications and consensus agreements among IoT nodes.

The process of constructing the utility coverage set for each UAV is shown in Algorithm 2, where n is the iteration rounds for users to update gain $\xi_{ij}^k(t)$, that is, $\xi_{ij}^{k,n}(t)$ represents the gain in the n-th iteration, and $\varpi_{ij}^n(t)$ represents the corresponding resource cost. Specifically, Algorithm 2 gets the score of decision $x_{ij}^n(t)$ by $\xi_{ij}^{k,n}(t)$, and considers the energy efficiency of data offloading and the cost of computing resource consumption by sequentially adding users to the utility coverage set according to the ratio of energy efficiency to cost under the constraint of computing resource f_j^{max}. Then Algorithm 2 updates the computation offloading and path planning decision set $(X^*(t), P^*(t))$. Furthermore, Algorithm 2 updates the task set and the occupied resources of UAV v_j. $\Pi_{ij}^n(t)$ represents the utility coverage set of v_j centered on u_i in the n-th iteration, and $\xi_j^n(\Pi_{ij}^n(t))$ is the total gain of the utility coverage set of v_j. The gain of users that are not added to the coverage set is set to 0.

Based on the submodularity of the gain function and the constructed utility coverage set in Algorithm 2, DCOP can make computation offloading and path planning decisions to improve the total energy efficiency of the system. The pseudo code of DCOP is shown in Algorithm 3. DCOP mainly consists of three steps: i) Obtaining the offloading decisions according to the gain function; ii) Constructing the utility coverage set based on partial enumeration and greedy idea [49]; and iii) Updating and maintaining the maximum utility coverage set of the system through communications and consensus agreements. First, user u_i with tasks makes the offloading decision $x_{ij}(t)$ to maximize its own energy efficiency and scores decision $x_{ij}(t)$. Specifically, the gain function is used as the scoring function to evaluate the gain brought to the system by adding the user to the utility coverage set. Note that the gain function meets the law of diminishing marginal utility, which is defined in (19). Meanwhile, user u_i sends offloading decision to neighbors, and receives the decision information $x_{i'j}^n(t)$ from neighbors $u_{i'}$ to unify local information. Then, using the known decision information of neighbors, u_i utilizes Algorithm 2 to construct a utility coverage set based on partial enumeration and greedy idea, as shown in lines 2-17 of Algorithm 3. Note that for each user without task in time slot t, decision information is also obtained through communications, and a utility coverage set is constructed, as shown in lines 18-21 of Algorithm 3. Meanwhile, each user updates and maintains its current known maximum utility coverage set by communications and consensus agreements with the neighbor set $\tilde{U}_i$, as shown in lines 22-26 of Algorithm 3. When users and their neighbors maintain the same maximum utility coverage set for each UAV, DCOP stops iterating and returns the corresponding offloading decisions and path planning decisions.

V. Mathematical Analysis

Since multiple IoT nodes improve energy efficiency in a distributed computing, which means that each user does not have the global information in constructing the utility coverage set. Therefore, the global maximum utility coverage set cannot be directly obtained. However, through the communication process and consensus agreement in our algorithm, the local maximum utility coverage set maintained by each IoT node is passed to neighbor nodes for updating. According to the submodularity of the gain function in Section IV-B, it can be proved that DCOP obtains a suboptimal solution with a performance guarantee (Theorem 2). Next, we give the detailed proof of the performance guarantee of DCOP.

Theorem 2 ***Performance Guarantee***: The gain of energy efficiency $\xi(\Pi_G)$ of UAV v_j obtained by DCOP approximates the optimal gain $\xi(\Pi^*)$ with an approximation ratio $1 - \frac{1}{e}$, that is

$$\xi(\Pi_G) \geq \left(1 - \frac{1}{e}\right) \xi(\Pi^*). \tag{25}$$

Proof. In the process of constructing the utility coverage set in Algorithm 3, suppose users are added to a non-empty initial set with d elements based on a greedy idea. According to the properties of a greedy algorithm with knapsack constraint, a performance bound depending on the initial set can be obtained [48], [50]. Furthermore, the algorithm has the performance bound with constant-factor approximation, which can be proved in Theorem 2.

Denote the optimal utility coverage set of UAV v_j as $\Pi^* = \{u_1^*, u_2^*, \ldots, u_n^*\}$, the set of the first i elements in Π^* as $\Pi_i^* = \{u_1^*, u_2^*, \ldots, u_i^*\}$, where the elements of Π^* are ordered according to the following formula

$$u_i^* = \underset{u \in \Pi^* \backslash \Pi_{i-1}^*}{\arg\max}\ \xi_{\Pi_{i-1}^*}(u), \tag{26}$$

where $\xi_{\Pi_{i-1}^*}(u)$ is the marginal utility of adding u to the set Π_{i-1}^*. The initial utility coverage set is composed of the first d elements in Π^*, denoted as Π_d^*. Then, Algorithm 3 can be regarded as a greedy algorithm with knapsack constraint. Denote $U \backslash \Pi_d^*$ as the candidate set, the UAV's remaining resource $f_d \triangleq f_j^{\max} - \varpi(\Pi_d^*)$ is the resource constraint, and the function $\xi_{\Pi_d^*}$ is the marginal utility.

In Algorithm 3, if the resource constraint is violated when u^* is added, that is, $\varpi(\Omega_G + u^*) \geq f_d$, the algorithm exits the iteration process. Now the set selected by line 2-9 in Algorithm 2 is denoted as Ω_G, the final utility coverage set generated by Algorithm 3 is denoted as Π_G, and we have

$$\Pi_G = \Pi_d^* \cup \Omega_G. \tag{27}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3175050, IEEE Internet of Things Journal

Let $\Omega_i \subseteq U \backslash \Pi_d^*$ represents the set consisting of the first i elements selected by line 2-9 in Algorithm 2, $\Omega^* \subseteq U \backslash \Pi_d^*$ represents the optimal utility coverage set that can be selected, and we have

$$\Pi^* = \Pi_d^* \cup \Omega^*. \tag{28}$$

According to the properties of the monotone submodular function [48], we have

$$\begin{aligned}
\xi_{\Pi_d^*}(\Omega^*) &\leq \xi_{\Pi_d^*}(\Omega_{i-1}) + \sum_{u_i \in \Omega^* \backslash \Omega_{i-1}} (\xi_{\Pi_d^*})_{\Omega_{i-1}}(u_i) \\
&= \xi_{\Pi_d^*}(\Omega_{i-1}) + \sum_{u_i \in \Omega^* \backslash \Omega_{i-1}} \varpi(u_i) \frac{(\xi_{\Pi_d^*})_{\Omega_{i-1}}(u_i)}{\varpi(u_i)} \\
&\leq \xi_{\Pi_d^*}(\Omega_{i-1}) + \sum_{u_i \in \Omega^* \backslash \Omega_{i-1}} \varpi(u_i) \frac{(\xi_{\Pi_d^*})_{\Omega_{i-1}}(u_i^*)}{\varpi(u_i^*)}
\end{aligned} \tag{29}$$

where $\xi_{\Pi_d^*}(\Omega^*)$ represents the marginal utility brought by adding the set Ω^* to Π_d^*. The first inequation in (29) is based on the properties of submodular function, the second inequation is because the marginal utility of the i-th user u_i^* in the optimal utility coverage set is not lower than the marginal utility of an arbitrary user u_i. Then we have

$$\begin{aligned}
&\xi_{\Pi_d^*}(\Omega_{i-1}) + \sum_{u_i \in \Omega^* \backslash \Omega_{i-1}} \varpi(u_i) \frac{(\xi_{\Pi_d^*})_{\Omega_{i-1}}(u_i^*)}{\varpi(u_i^*)} \\
&= \xi_{\Pi_d^*}(\Omega_{i-1}) + \sum_{u_i \in \Omega^* \backslash \Omega_{i-1}} \varpi(u_i) \frac{\xi_{\Pi_d^*}(\Omega_i) - \xi_{\Pi_d^*}(\Omega_{i-1})}{\varpi(u_i^*)} \\
&\leq \xi_{\Pi_d^*}(\Omega_{i-1}) + \frac{f_d}{\varpi(u_i^*)} \left(\xi_{\Pi_d^*}(\Omega_i) - \xi_{\Pi_d^*}(\Omega_{i-1}) \right),
\end{aligned} \tag{30}$$

The equation in (30) is because the line 4 of Algorithm 2 selects the optimal cost-utility user u_i^* to the utility coverage set. And the inequation is because the resource requirements of partial users in the optimal solution must be less than the available UAV resources f_d. Subtracting $\frac{f_d}{\varpi(u_i^*)} \xi_{\Pi_d^*}(\Omega^*)$ from both sides, we have

$$\xi_{\Pi_d^*}(\Omega_i) - \xi_{\Pi_d^*}(\Omega^*) \geq \left(1 - \frac{\varpi(u_i^*)}{f_d}\right) \left(\xi_{\Pi_d^*}(\Omega_{i-1}) - \xi_{\Pi_d^*}(\Omega^*) \right). \tag{31}$$

Solving the recursive inequation yields

$$\begin{aligned}
&\xi_{\Pi_d^*}(\Omega_i) - \xi_{\Pi_d^*}(\Omega^*) \geq \left(1 - \frac{\varpi(u_i^*)}{f_d}\right) \left(\xi_{\Pi_d^*}(\Omega_{i-1}) - \xi_{\Pi_d^*}(\Omega^*) \right) \\
&\geq \left(1 - \frac{\varpi(u_i^*)}{f_d}\right) \left(1 - \frac{\varpi(u_{i-1}^*)}{f_d}\right) \left(\xi_{\Pi_d^*}(\Omega_{i-2}) - \xi_{\Pi_d^*}(\Omega^*) \right) \\
&\vdots \\
&\geq \prod_{l=1}^{i} \left(1 - \frac{\varpi(u_l^*)}{f_d}\right) \left(-\xi_{\Pi_d^*}(\Omega^*) \right).
\end{aligned} \tag{32}$$

Reorganizing both sides, we have

$$\xi_{\Pi_d^*}(\Omega_i) \geq \left(1 - \prod_{l=1}^{i} \left(1 - \frac{\varpi(u_l^*)}{f_d}\right)\right) \xi_{\Pi_d^*}(\Omega^*). \tag{33}$$

Using $1 - x \leq e^{-x}$, and we have

$$\begin{aligned}
\xi_{\Pi_d^*}(\Omega_i) &\geq \left(1 - \prod_{l=1}^{i} \exp\left(-\frac{\varpi(u_l^*)}{f_d}\right)\right) \xi_{\Pi_d^*}(\Omega^*) \\
&= \left(1 - \exp\left(-\frac{\varpi(\Omega_i)}{f_d}\right)\right) \xi_{\Pi_d^*}(\Omega^*).
\end{aligned} \tag{34}$$

The above analysis does not consider the assumption that the resource constraint needs to be met when adding u_i to Ω_G. In DCOP, adding users to the offloading set will take resource consumption of UAVs. Therefore, the relation of (34) can be applied to $\Omega_G + u^*$, where adding user u^* just violates the resource constraint, $\varpi(\Omega_G) \leq f_d$, and $\varpi(\Omega_G) + \varpi(u^*) \geq f_d$, that is

$$\begin{aligned}
\xi_{\Pi_d^*}(\Omega_G + u^*) &\geq \left(1 - \exp\left(-\frac{\varpi(\Omega_G + u^*)}{f_d}\right)\right) \xi_{\Pi_d^*}(\Omega^*) \\
&= \left(1 - \exp\left(-\frac{\varpi(\Omega_G) + \varpi(u^*)}{f_d}\right)\right) \xi_{\Pi_d^*}(\Omega^*) \\
&\geq \left(1 - \exp\left(-\frac{f_d}{f_d}\right)\right) \xi_{\Pi_d^*}(\Omega^*) \\
&= \left(1 - \frac{1}{e}\right) \xi_{\Pi_d^*}(\Omega^*)
\end{aligned} \tag{35}$$

where $\Omega_G + u^*$ means adding u^* to the set Ω_G. According to the definition of the marginal utility function, (27) and (28), (35) can be further simplified as

$$\xi(\Pi_G + u^*) - \xi(\Pi_d^*) \geq \left(1 - \frac{1}{e}\right) \left(\xi(\Pi^*) - \xi(\Pi_d^*) \right). \tag{36}$$

Furthermore, according to the submodularity of the gain function in Lemma 1 and the ordering of elements in Π_d^*, we have

$$\xi_{\Pi_G}(u^*) \leq \xi_{\Pi_i^*}(u^*) \leq \xi_{\Pi_{i-1}^*}(u_i^*) = \xi(\Pi_i^*) - \xi(\Pi_{i-1}^*), \tag{37}$$

where $\Pi_i^* \subseteq \Pi_G$, and $1 \leq i \leq d$. Summing both sides for $i = 1, 2, \ldots, d$, using the law of telescoping sum yields

$$d \cdot \xi_{\Pi_G}(u^*) \leq \xi(\Pi_d^*) - \xi(\Pi_0^*) = \xi(\Pi_d^*) - \xi(\emptyset) = \xi(\Pi_d^*). \tag{38}$$

Expanding the gain function on the left side of the inequation and reorganizing the terms, we have

$$\xi(\Pi_G + u^*) \leq \xi(\Pi_G) + \frac{1}{d} \xi(\Pi_d^*). \tag{39}$$

Combining (36) and (39), we have

$$\xi(\Pi_G) \geq \left(1 - \frac{1}{e}\right) \xi(\Pi^*) + \left(\frac{1}{e} - \frac{1}{d}\right) \xi(\Pi_d^*). \tag{40}$$

For (40), when $\frac{1}{e} - \frac{1}{d} \geq 0$, we have $d \geq 3$. By scaling the last term on the right side of the inequation, we get the performance bound (25), which shows that the solution of Algorithm 3 can approximate the optimal solution with an approximation ratio $1 - \frac{1}{e}$. Hence, Theorem 2 is proved. □

VI. Performance Evaluation

In this section, we evaluate the performance of the proposed DCOP through extensive simulations. We simulate a three-dimensional space 1000*1000*50 m^3, where 110 anchor nodes

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3175050, IEEE Internet of Things Journal

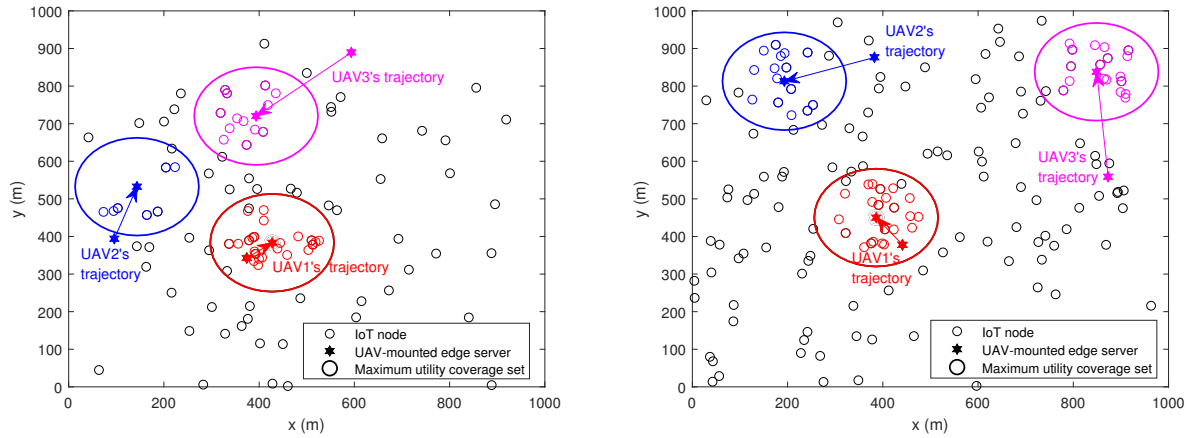


Fig. 3. (a) Trajectories of three UAVs in two adjacent time slots, U=400, P=0.25, (b) Trajectories of three UAVs in two adjacent time slots, U=800, P=0.2.

TABLE II
SIMULATION PARAMETERS

Parameter	Value	Parameter	Value
$\|U\|$	300-1000	$\|V\|$	3
$\|R\|$	110	$f_j^{\max}$	3-17 GHz
$\hat{e}$	5	$\tilde{\theta}$	15°
ρ_0	-50 dB	B	4 MHz
β_i	1 W	σ^2	-110 dBm
η_1	9.26×10^{-4}	η_2	2250

are regularly deployed, 1000 IoT nodes are deployed in random locations and moving continuously, and 3 UAVs provide computational support to IoT nodes. To evaluate the dynamic optimization process of DCOP, we use the quasi-static network scenarios for simulations [34], [38]. Specifically, in one time slot, the locations of UAV and IoT nodes are fixed, but IoT nodes have mobility between different time slots. In order to comprehensively evaluate the performance of the algorithms, IoT nodes follow two movement models in the simulations, including map-based movement model for 40% of IoT nodes and random walk model for 60% of IoT nodes [51]. In each time slot, IoT nodes randomly generate tasks with a probability $P \in [0.2, 0.5]$. Each UAV is equipped with multiple CPU cores with the maximum computing capacity from 3 GHz to 17 GHz. The IoT task size $\sigma_i^k(t)$ is uniformly distributed within [0.5, 2.5] Mb, and the required computing intensity $w_i^k(t)$ is uniformly distributed within [500, 1000] cycles/bit. The parameters of propulsion energy consumption [45] and other important parameters in the simulations are shown in Table II.

Furthermore, we conduct performance comparisons between our algorithm and the following algorithms: 1) Utility-based greedy distributed algorithm (UGDA)-the utility coverage set is constructed based on the maximum marginal utility of users, rather than the ratio of the marginal utility to the occupied resources, that is, $\Pi_{ij}^n(t) \leftarrow \Pi_{ij}^n(t) \bigcup \arg\max_{u_i \in \tilde{U}_{ij} \setminus \Pi_{ij}^n(t)} \xi_{ij}^{k,n}(t)$; 2) Maximum utility first algorithm (MUFA)-borrowing the idea of maximizing the total bits offloaded to UAVs in [37], UAVs sequentially select the user with the largest utility as the center to construct the utility coverage set, and perform computation offloading; 3) Nearest user first algorithm (NUFA)-UAVs construct utility coverage set centered on the nearest user and provide computational support; 4) Random optimization algorithm (ROA)-UAVs randomly select users with tasks to construct utility coverage set and perform computation offloading; 5) Circular fixed cycle algorithm (CFCA)-used in [16], UAVs perform circular cycle movement with a fixed radius to cover the entire area as much as possible and provide computational support. We evaluate the performance of the algorithms in terms of the following metrics: energy efficiency, system utility, energy consumption, number of serving requests, and resource utilization.

Furthermore, we evaluate the positioning algorithm TSIP with different numbers of IoT nodes. Specifically, we utilize the root mean square error (RMSE) [21] to represent the positioning error, which is given by

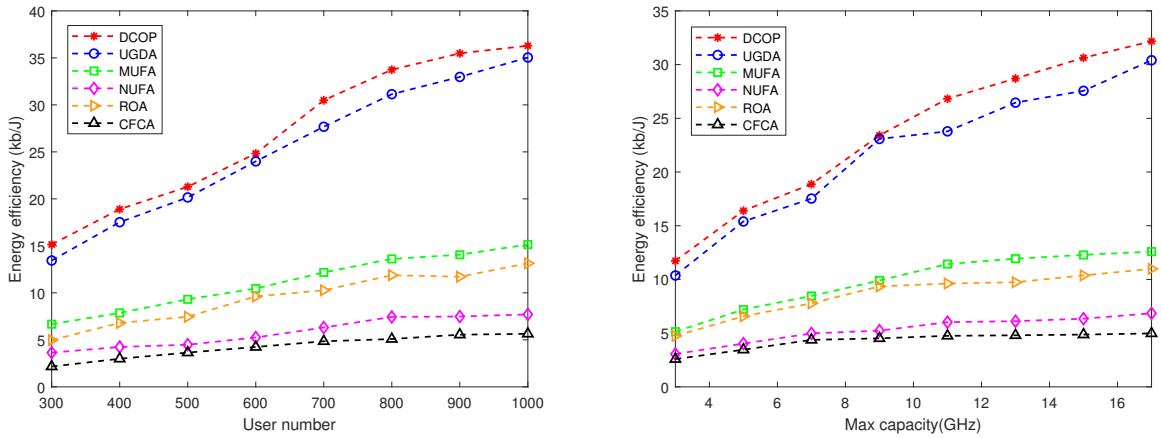
$$\varepsilon = \sqrt{\frac{1}{|U|} \sum_{u_i \in U} \|p_i(t) - p_i^*(t)\|^2}, \tag{41}$$

where $p_i^*(t)$ is the real location of IoT node u_i, and $p_i(t)$ is the estimated location of u_i by the positioning algorithm. We compare TSIP and the latest positioning algorithm iTOA [21] in terms of positioning error.

A. Energy Efficiency

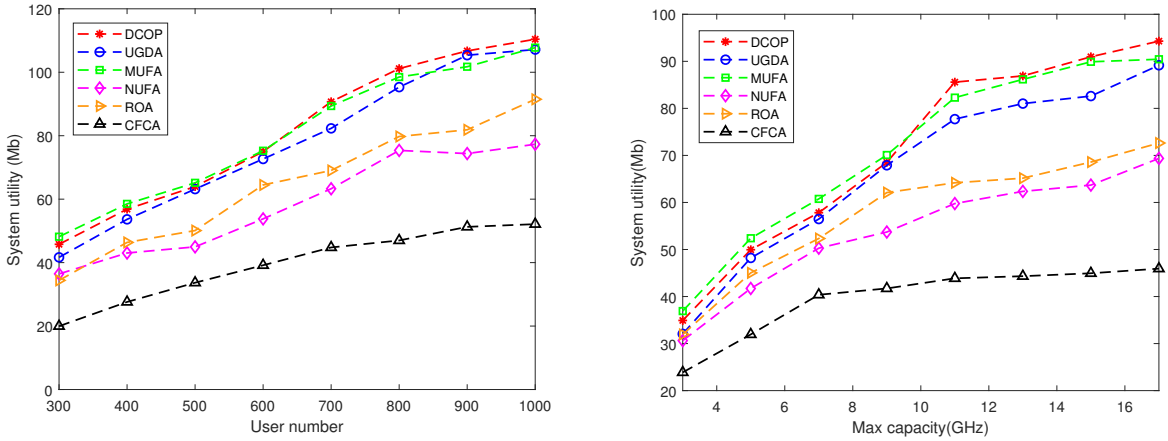
Energy efficiency is an important performance metric, which represents the execution efficiency of the multi-UAV-enabled edge computing system. We evaluate the energy efficiency of our proposed algorithm and compared algorithms under different numbers of IoT nodes and UAV processing capabilities, which is defined as shown in (10). Fig. 3 shows the trajectories of three UAVs in two adjacent time slots and the maximum utility coverage set constructed by DCOP under different numbers of users. Note that Fig. 3 only shows the IoT nodes with tasks in the current time slot, and these tasks are randomly

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3175050, IEEE Internet of Things Journal



(a) (b)

Fig. 4. (a) Energy efficiency under different numbers of users, (b) Energy efficiency under different computation capacities of UAVs.



(a) (b)

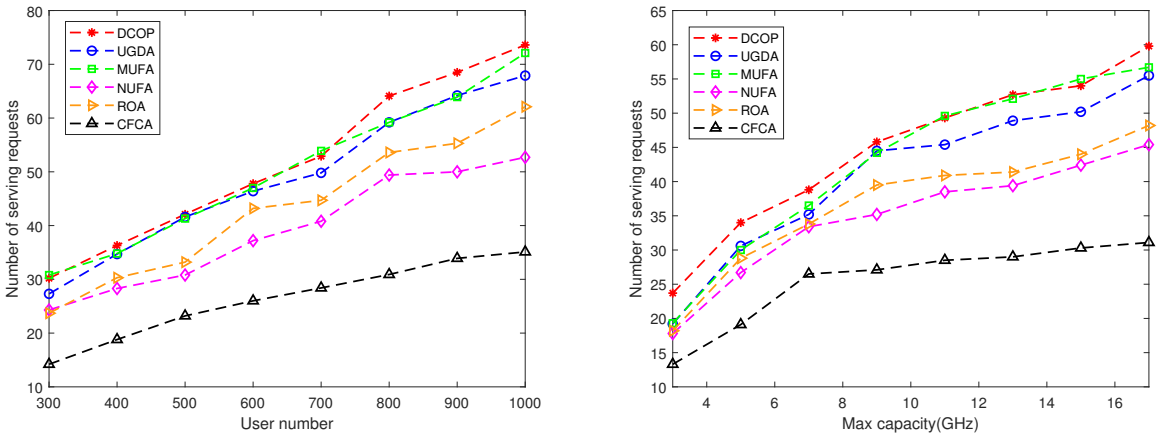
Fig. 5. (a) System utility under different numbers of users, (b) System utility under different computation capacities of UAVs.

generated with probability P. Our proposed DCOP constructs the maximum utility coverage set based on energy efficiency and resource cost. Simulation results of energy efficiency are shown in Fig. 4, where we can observe that DCOP achieves the highest energy efficiency, and the energy efficiency of UGDA approaches DCOP. Specifically, as shown in Fig. 4(a), when the number of IoT nodes is 500, the energy efficiency of DCOP is 21.29, which achieves 128.68% gains than MUFA. When the number of IoT nodes is 800, the energy efficiency of DCOP is 33.74, which achieves 147.72% gains over MUFA and 8.35% gains over UGDA. The energy efficiency with different UAV capacities is shown in Fig. 4(b). When the UAV capacity is 5 GHz, the energy efficiency of DCOP is 16.40, which gains more than 6.50% over the comparison algorithms. This is because DCOP considers both the utility and the energy consumption of UAVs in constructing the utility coverage set, and selects the maximum utility coverage set to plan the UAV's trajectory and perform computation offloading, which improves the energy efficiency of the system. Furthermore, Theorem 2 also provides a theoretical guarantee for the energy efficiency performance of DCOP.

B. System Utility

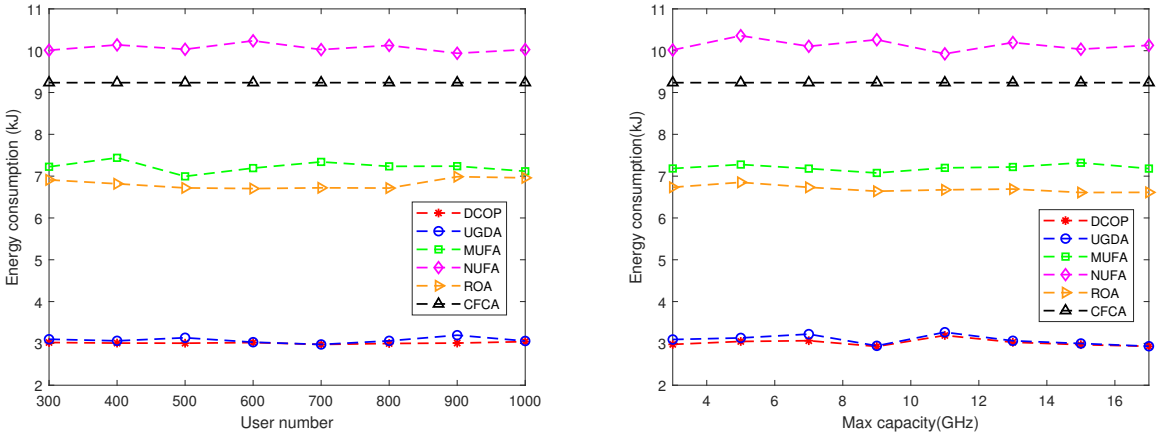
The system utility is the amount of data offloaded to UAVs, which represents the ability of UAVs to provide computational support to IoT nodes. Fig. 5 shows the system utility under different numbers of IoT nodes and UAV capabilities. MUFA achieves a high system utility because it aims to maximize the utility of computation offloading regardless of the energy consumption of UAVs and occupied computing resources. Therefore, when the number of users is insufficient, MUFA can maximize the system utility with sufficient resources. However, MUFA cannot execute decisions according to the optimal cost-utility, and performs worse than DCOP with more users. Specifically, as shown in Fig. 5(a), when the number of IoT nodes is 700, the system utility of DCOP is 90.70 Mb, and when the number of IoT nodes increases to 1000, the system utility is 110.41 Mb, which gains 20.69% than ROA. The system utility versus UAV capability is shown in Fig. 5(b). When the UAV capability is 11 GHz, the system utility of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3175050, IEEE Internet of Things Journal



(a) (b)

Fig. 6. (a) Number of serving requests by UAVs under different numbers of users, (b) Number of serving requests by UAVs under different computation capacities of UAVs.



(a) (b)

Fig. 7. (a) Energy consumption of UAVs under different numbers of users, (b) Energy consumption of UAVs under different computation capacities of UAVs.

DCOP is as high as 85.58 Mb and achieves 10.11% gains over UGDA. The main reason is that DCOP takes into account the amount of data offloaded from IoT nodes and the computing resources occupied by tasks, adding users according to the ratio of utility to cost, and selecting the maximum utility coverage set for execution. DCOP improves system utility and provides powerful computational support to IoT nodes.

C. Number of Serving Requests

The number of serving requests by UAVs is also an important metric, which represents the ability of UAVs to admit IoT node requests. Simulation results of the number of serving requests are shown in Fig. 6. From Fig. 6(a), we can observe that as the number of users increases, DCOP performs best in the number of serving requests. As shown in Fig. 6(b), when the UAV capability is 9 GHz, the average number of user requests processed by DCOP is 45.80. When the UAV capability increases to 17 GHz, the average number of user requests processed by DCOP is 59.80, which achieves 5.47% gains over MUFA and 24.10% gains over ROA. Simulation results demonstrate that DCOP achieves high performance in the number of serving requests. This is because DCOP can reasonably plan the trajectories of UAVs and make offloading decisions by constructing utility coverage sets and consensus agreements, which largely admits IoT node requests.

D. Energy Consumption

The energy consumption of UAVs is an essential metric for evaluating system performance, and it is affected by the randomness of user movement. Fig. 7 shows the average value of the energy consumption of 3 UAVs. The average energy consumption of these algorithms tends to be stable because the propulsion energy consumption of UAVs is much greater than the computational energy consumption. The energy consumption of UGDA approaches the value of DCOP because these two algorithms are implemented by the idea of constructing the maximum utility coverage set and consensus agreement to perform trajectory planning. The average energy consumption

TABLE III
POSITIONING ERROR

Number of IoT nodes	100	300	500	700	1000
TSIP(m)	7.11	7.28	7.56	7.63	7.78
iTOA(m)	9.24	9.27	9.40	9.46	9.52

of DCOP is 3.01 kJ, which is 55.85% lower than ROA. The main reason is that DCOP considers both the amount of data offloaded from IoT nodes and the energy consumption of UAVs, which significantly improves the energy efficiency of UAVs by reasonably planning the trajectories of UAVs.

E. Computation Resource Utilization and Positioning Error

The computation resource utilization represents the utilization efficiency of each UAV, and the positioning error can reflect the positioning accuracy of our proposed TSIP algorithm. We evaluate the computation resource utilization of the algorithms with 3 UAVs as shown in Fig. 8. The average resource utilizations of 3 UAVs of DCOP are 98.37%, 99.61%, and 96.98%, respectively, which are significantly higher than those of other comparison algorithms. This is because DCOP fully considers the amount of data offloaded to UAVs and the computational resources occupied by tasks, and sequentially adds the most cost-utility task to the coverage set. Furthermore, DCOP selects the maximum utility coverage set to plan the UAVs trajectories and perform computation offloading, making full use of UAVs resources and improving resource utilization.

Furthermore, we simulate 1000 time slots and take the average value of positioning error, as shown in Table III. As the number of IoT nodes that need to be positioned increases, the positioning error also increases. This is because the increase in the number of IoT nodes reduces the average number of effective anchor nodes used for each IoT node positioning. These anchor nodes have limited energy and communication range, which increases positioning error. Specifically, when the number of IoT nodes is 100, the average positioning error of TSIP is 7.11 m, and the average positioning error of iTOA is 9.24 m. When the number of IoT nodes increases to 1000, the average positioning error of TSIP is 7.78 m, and the average positioning error of iTOA increases to 9.52 m. Overall, the positioning error of TSIP is lower than that of iTOA. This is because the TSIP constructs the positioning units based on the proposed anchor node selection theorem, which considers the availability of anchor nodes, as well as the angles between anchor nodes and IoT nodes. TSIP greatly reduces the positioning error by constructing the positioning unit reasonably. Moreover, we provide the theoretical analysis in Theorem 1 for reducing the positioning error in TSIP.

VII. CONCLUSION

In this paper, we have studied the joint optimization problem of dynamic computational offloading of multiple IoT nodes and path planning of multiple UAVs. Considering the mobility of IoT nodes, an intelligent positioning algorithm has been designed to obtain real-time location information of IoT nodes in the three-dimensional space. Then, an online distributed algorithm has been proposed for IoT nodes computation offloading and UAVs path planning, which breaks through the limitation of insufficient global information. Furthermore, we have proved that our proposed DCOP algorithm has a performance guarantee related to energy efficiency. Finally, simulation results have demonstrated that DCOP effectively improves the system utility and significantly reduces the energy consumption of UAVs. Since the complex terrain environment affects the performance and stability of the system, we will consider improving the reliability and stability of the UAV-enabled edge computing system under interference and obstacle environments in future works.

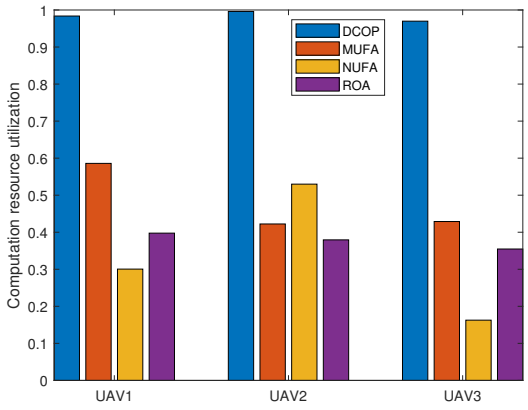


Fig. 8. Resource utilization of UAVs.

REFERENCES

[1] W. Rafique, L. Qi, I. Yaqoob, M. Imran, R. U. Rasool, and W. Dou, "Complementing IoT services through software defined networking and edge computing: A comprehensive survey," *IEEE Commun. Surv. Tut.*, vol. 22, no. 3, pp. 1761–1804, May 2020.

[2] G. M. D. T. Forecast, "Cisco visual networking index: Global mobile data traffic forecast update, 2017–2022," *Cisco white paper*, vol. 2017, pp. 1–36, Feb. 2019.

[3] C. Mateos, K. R. Choo, and A. Zunino, "Sharpening the edge: Towards improved edge computing environment for mobile and IoT applications," *Future Gener. Comput. Syst.*, vol. 107, pp. 1130–1133, Jun. 2020.

[4] W. Zhuang, Q. Ye, F. Lyu, N. Cheng, and J. Ren, "SDN/NFV-empowered future IoV with enhanced communication, computing, and caching," *Proc. IEEE*, vol. 108, no. 2, pp. 274–291, Feb. 2020.

[5] 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. Inform.*, vol. 16, no. 8, pp. 5505–5516, Oct. 2020.

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

[7] T. Nguyen, E. Huh, and M. Jo, "Decentralized and revised content-centric networking-based service deployment and discovery platform in mobile edge computing for IoT devices," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4162–4175, Jun. 2019.

[8] Y. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing-a key technology towards 5G," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, Sept. 2015.

[9] G. Castellano, F. Esposito, and F. Risso, "A distributed orchestration algorithm for edge computing resources with guarantees," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2019, pp. 2548–2556.

[10] B. P. Rimal, D. P. Van, and M. Maier, "Cloudlet enhanced fiber-wireless access networks for mobile-edge computing," *IEEE Trans. Wirel. Commun.*, vol. 16, no. 6, pp. 3601–3618, Mar. 2017.

[11] A. Ksentini and P. A. Frangoudis, "On extending ETSI MEC to support LoRa for efficient IoT application deployment at the edge," *IEEE Commun. Stand. Mag.*, vol. 4, no. 2, pp. 57–63, Jun. 2020.

[12] S. Wan, J. Lu, P. Fan, and K. B. Letaief, "Toward big data processing in IoT: Path planning and resource management of UAV base stations in mobile-edge computing system," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5995–6009, Jun. 2020.

[13] X. Xiong, K. Zheng, L. Lei, and L. Hou, "Resource allocation based on deep reinforcement learning in IoT edge computing," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1133–1146, Apr. 2020.

[14] Z. Chang, L. Liu, X. Guo, and Q. Sheng, "Dynamic resource allocation and computation offloading for IoT fog computing system," *IEEE Trans. Ind. Inform.*, vol. 17, no. 5, pp. 3348–3357, Jun. 2021.

[15] X. Hu, K. Wong, K. Yang, and Z. Zheng, "UAV-assisted relaying and edge computing: Scheduling and trajectory optimization," *IEEE Trans. Wirel. Commun.*, vol. 18, no. 10, pp. 4738–4752, Dec. 2019.

[16] M. Li, N. Cheng, J. Gao, Y. Wang, L. Zhao, and X. Shen, "Energy-efficient UAV-assisted mobile edge computing: Resource allocation and trajectory optimization," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3424–3438, Jan. 2020.

[17] L. D. Nguyen, A. E. Kalør, I. L. Mayorga, and P. Popovski, "Trusted wireless monitoring based on distributed ledgers over NB-IoT connectivity," *IEEE Commun. Mag.*, vol. 58, no. 6, pp. 77–83, Apr. 2020.

[18] M. Pradhan and J. Noll, "Security, privacy, and dependability evaluation in verification and validation life cycles for military IoT systems," *IEEE Commun. Mag.*, vol. 58, no. 8, pp. 14–20, Aug. 2020.

[19] Z. Tan, H. Qu, J. Zhao, S. Zhou, and W. Wang, "UAV-aided edge/fog computing in smart IoT community for social augmented reality," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 4872–4884, Feb. 2020.

[20] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 1927–1941, Jun. 2018.

[21] Y. Kang, Q. Wang, J. Wang, and R. Chen, "A high-accuracy TOA-based localization method without time synchronization in a three-dimensional space," *IEEE Trans. Ind. Inform.*, vol. 15, no. 1, pp. 173–182, Jan. 2019.

[22] Y. Li, Y. Zhuang, X. Hu, Z. Gao, J. Hu, L. Chen, Z. He, L. Pei, K. Chen, M. Wang, X. Niu, R. Chen, J. Thompson, F. M. Ghannouchi, and N. El-Sheimy, "Toward location-enabled IoT (LE-IoT): IoT positioning techniques, error sources, and error mitigation," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4035–4062, Apr. 2021.

[23] X. Chen, X. Wang, B. Yi, Q. He, and M. Huang, "Deep learning-based traffic prediction for energy efficiency optimization in software-defined networking," *IEEE Syst. J.*, vol. 15, no. 4, pp. 5583–5594, Dec. 2021.

[24] L. Li, T. Q. S. Quek, J. Ren, H. H. Yang, Z. Chen, and Y. Zhang, "An incentive-aware job offloading control framework for multi-access edge computing," *IEEE Trans. Mobile Comput.*, vol. 20, no. 1, pp. 63–75, Jan. 2021.

[25] 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, May 2016.

[26] J. Martín-Pérez, L. Cominardi, C. J. Bernardos, A. de la Oliva, and A. Azcorra, "Modeling mobile edge computing deployments for low latency multimedia services," *IEEE Trans. Broadcast.*, vol. 65, no. 2, pp. 464–474, Mar. 2019.

[27] K. Poularakis, J. Llorca, A. M. Tulino, I. J. Taylor, and L. Tassiulas, "Service placement and request routing in MEC networks with storage, computation, and communication constraints," *IEEE/ACM Trans. Netw.*, vol. 28, no. 3, pp. 1047–1060, Apr. 2020.

[28] J. Chen, Z. Chang, X. Guo, R. Li, Z. Han, and T. Hämäläinen, "Resource allocation and computation offloading for multi-access edge computing with fronthaul and backhaul constraints," *IEEE Trans. Veh. Technol.*, vol. 70, no. 8, pp. 8037–8049, Aug. 2021.

[29] T. Taleb, A. Ksentini, and P. A. Frangoudis, "Follow-me cloud: When cloud services follow mobile users," *IEEE Trans. Cloud Comput.*, vol. 7, no. 2, pp. 369–382, Apr. 2019.

[30] A. Aissioui, A. Ksentini, A. M. Guéroui, and T. Taleb, "On enabling 5G automotive systems using follow me edge-cloud concept," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 5302–5316, Feb. 2018.

[31] X. Chen, Y. Bi, X. Chen, H. Zhao, N. Cheng, F. Li, and W. Cheng, "Dynamic service migration and request routing for microservice in multi-cell mobile edge computing," *IEEE Internet Things J.*, vol. Early Access, 2022.

[32] T. Ouyang, Z. Zhou, and X. Chen, "Follow me at the edge: Mobility-aware dynamic service placement for mobile edge computing," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 10, pp. 2333–2345, Sept. 2018.

[33] S. Wang, Y. Guo, N. Zhang, P. Yang, A. Zhou, and X. Shen, "Delay-aware microservice coordination in mobile edge computing: A reinforcement learning approach," *IEEE Trans. Mobile Comput.*, vol. 20, no. 3, pp. 939–951, Dec. 2021.

[34] Y. Liu, Y. Li, Y. Niu, and D. Jin, "Joint optimization of path planning and resource allocation in mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 9, pp. 2129–2144, Sept. 2020.

[35] F. Guo, H. Zhang, H. Ji, X. Li, and V. C. Leung, "Joint trajectory and computation offloading optimization for UAV-assisted MEC with NOMA," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM Workshops)*, Apr. 2019, pp. 1–6.

[36] 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, Sept. 2018.

[37] Y. Qian, F. Wang, J. Li, L. Shi, K. Cai, and F. Shu, "User association and path planning for UAV-aided mobile edge computing with energy restriction," *IEEE Wirel. Commun. Lett.*, vol. 8, no. 5, pp. 1312–1315, Apr. 2019.

[38] Q. Pham, T. LeAnh, N. H. Tran, and C. S. Hong, "Decentralized computation offloading and resource allocation in heterogeneous networks with mobile edge computing," *CoRR*, vol. abs/1803.00683, Mar. 2018.

[39] Q. Ye, W. Shi, K. Qu, H. He, W. Zhuang, and X. S. Shen, "Learning-based computing task offloading for autonomous driving: A load balancing perspective," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2021, pp. 1–6.

[40] Y. Sun, S. Zhou, and J. Xu, "EMM: energy-aware mobility management for mobile edge computing in ultra dense networks," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2637–2646, Sept. 2017.

[41] Y. Zeng, R. Zhang, and T. J. Lim, "Throughput maximization for UAV-enabled mobile relaying systems," *IEEE Trans. Commun.*, vol. 64, no. 12, pp. 4983–4996, Sept. 2016.

[42] 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, Jan. 2020.

[43] Y. Wang, Z. Ru, K. Wang, and P. Huang, "Joint deployment and task scheduling optimization for large-scale mobile users in multi-UAV-enabled mobile edge computing," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3984–3997, Sept. 2020.

[44] H. He, S. Zhang, Y. Zeng, and R. Zhang, "Joint altitude and beamwidth optimization for UAV-enabled multiuser communications," *IEEE Commun. Lett.*, vol. 22, no. 2, pp. 344–347, Nov. 2018.

[45] Y. Zeng and R. Zhang, "Energy-efficient UAV communication with trajectory optimization," *IEEE Trans. Wirel. Commun.*, vol. 16, no. 6, pp. 3747–3760, Aug. 2017.

[46] Y. Jiao, P. Wang, S. Feng, and D. Niyato, "Profit maximization mechanism and data management for data analytics services," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2001–2014, Mar. 2018.

[47] A. Krause and D. Golovin, "Submodular function maximization," in *Tractability: Practical Approaches to Hard Problems*. Cambridge Univ. Press, Feb. 2014, pp. 71–104.

[48] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions - I," *Math. Program.*, vol. 14, no. 1, pp. 265–294, Dec. 1978.

[49] C. Qian, J. Shi, K. Tang, and Z. Zhou, "Constrained monotone k-submodular function maximization using multiobjective evolutionary algorithms with theoretical guarantee," *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 595–608, Aug. 2018.

[50] M. Sviridenko, "A note on maximizing a submodular set function subject to a knapsack constraint," *Oper. Res. Lett.*, vol. 32, no. 1, pp. 41–43, Jan. 2004.

[51] S. Bandyopadhyay, E. J. Coyle, and T. Falck, "Stochastic properties of mobility models in mobile ad hoc networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 11, pp. 1218–1229, Oct. 2007.

Xiangyi Chen received the M.S. degree in computer science from Northeastern University, Shenyang, China, in 2019. She is currently pursuing the Ph.D. degree in computer science at Northeastern University, Shenyang, China. Her research interests include edge computing, software-defined networking and network function virtualization, etc.

Yuanguo Bi (M'11) received the Ph.D. degree in computer science from Northeastern University, Shenyang, China, in 2010. He was a Visiting Ph.D. Student with the BroadBand Communications Research (BBCR) lab, Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada from 2007 to 2009. He is currently a Professor with the School of Computer Science and Engineering, Northeastern University. He has authored/coauthored more than 50 journal/conference papers, including high quality journal papers, such as IEEE JSAC, IEEE TWC, IEEE TITS, IEEE TVT, IEEE IoT Journal, IEEE Communications Magazine, IEEE Wireless Communications, IEEE Network, and mainstream conferences, such as IEEE Global Communications Conference, IEEE International Conference on Communications. His research interests include medium access control, QoS routing, multihop broadcast, and mobility management in vehicular networks, software-defined networking, and mobile edge computing. Dr. Bi has served as an Editor/Guest Editor for IEEE Communications Magazine, IEEE Wireless Communications, IEEE ACCESS. He has also served as the Technical Program Committee member for many IEEE conferences.

Guangjie Han (S'03–M'05–SM'18) is currently a Professor with the Department of Information and Communication System, Hohai University, Changzhou, and State Key Laboratory of Acoustics, Institute of Acoustics, Chinese Academy of Sciences, Beijing, China. He received the Ph.D. degree from Northeastern University, Shenyang, China, in 2004. In February 2008, he finished his work as a Postdoctoral Researcher with the Department of Computer Science, Chonnam National University, Gwangju, Korea. From October 2010 to October 2011, he was a Visiting Research Scholar with Osaka University, Suita, Japan. From January 2017 to February 2017, he was a Visiting Professor at City University of Hong Kong, China. His current research interests include Internet of Things, Industrial Internet, Machine Learning and Artificial Intelligence, Mobile Computing, Security and Privacy. Dr. Han has over 400 peer-reviewed journal and conference papers, in addition to 140 granted and pending patents. Currently, his H-index is 46 and i10-index is 171 in Google Citation (Google Scholar). The total citation of his papers by other people is more than 8600+ times. Dr. Han is a Fellow of the UK Institution of Engineering and Technology (FIET). He has served on the Editorial Boards of up to 10 international journals, including the IEEE Network, IEEE Systems, IEEE/CCA JAS, IEEE ACCESS, Telecommunication Systems, etc. He has guest edited a number of special issues in IEEE Journals and Magazines, including the IEEE JSAC, IEEE Communications, IEEE Wireless Communications, IEEE Transactions on Industrial Informatics, Computer Networks, etc. Dr. Han has also served as chairs of organizing and technical committees of many international conferences. He had been awarded 2020 IEEE Systems Journal Annual Best Paper Award, 2017-2019 IEEE ACCESS Outstanding Associate Editor Award. He is a Senior Member of IEEE.

Dongyu Zhang received the B.S. degree from Liaoning University, Shenyang, China, in 2019. She is currently pursuing the master's degree in computer science at Northeastern University, Shenyang, China. Her research interests include space and terrestrial network and mobile edge computing.

Minghan Liu received the B.S. degree from Hebei University of Technology, Tianjin, China, in 2020. She is currently pursuing the master's degree in computer science in Northeastern University, Shenyang, China. Her research interests include mobile edge computing and artificial intelligence.

Han Shi received the B.S. and M.S. degrees in Information Science and Engineering from the Shenyang University of Technology, Shenyang, China, in 2014 and 2017, respectively. He is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, Northeastern University, Shenyang, China. He current research interests include Wireless Sensor Networks, Body Sensor Networks and information fusion.

Hai Zhao received the B.S. degree in electrical engineering from Dalian Maritime University, Dalian, China, in 1982, and the M.S. and Ph.D. degrees in computer science from Northeastern University, China, in 1987 and 1995, respectively. He is currently a professor with the school of Computer Science and Engineering, Northeastern University, Shenyang, China. He is also the Director of the Liaoning Provincial Key Laboratory of Embedded Technology. His current research interests include embedded Internet technology, wireless sensor networks, vehicular ad hoc networks, body area networks, pervasive computing, operating systems, data and information fusion, computer simulation, and virtual reality.

Fengyun Li received her Ph.D. degree in computer architecture from Northeastern University, Shenyang, China, in 2013. She is currently an associate professor with the School of Computer Science and Engineering, NortheasternUniversity, Shenyang, China. Her current researches interests include edge computing, autonomous driving, network security, privacy preserving.