An UAV-Enabled Intelligent Connected Transportation System With 6G Communications for Internet of Vehicles

Run Liu, Anfeng Liu, Zhenzhe Qu, and Neal N. Xiong, *Senior Member, IEEE*

***Abstract*— 6G networks provide faster communication for connected vehicles. These vehicles are connected to the Internet, forming the Internet of Vehicles (IoV). Due to the development of Intelligent Transportation Systems (ITS), more and more vehicles are deployed with data-intensive applications. These applications interact heavily with IoT devices at the edge of the network, which causes IoT devices to consume a lot of limited and valuable power. Task offloading can help overcome resource constraints of IoT devices by offloading task to edge server which has sufficient computational power in ITS. Unmanned Aerial Vehicles (UAV) is a promising solution by serving as Computing-Communications Edge Server (CCES) for resource-constrained IoT devices that there is no edge server nearby that can offload task. Due to the IoT devices' limited battery capacity and UAV energy budget, it is a challenging issue to reduce the energy for task offloading in UAV-enable edge network. In this paper, an UAV-enabled Computing-Communications Intelligent Offloading (UAV-CCIO) scheme is proposed to offload task energy-efficiently. First, some nodes with a large amount of data are selected as Task Gathering Nodes (TGNs), and TGNs collect all the tasks of the left nodes. In this way, the UAV can only fly the TGNs and so all the IoT devices' tasks can be offloaded. The distance needed for the UAV can be greatly reduced and energy is saved. On the other hand, tasks that route to TGNs have a relatively small amount of data, while nodes with a large amount of data have already been selected as TGNs without routing, thus saving energy. Second, an optimization strategy for collection tasks is proposed to reduce UAV's energy. The extensive experimental simulations indicate that the performance of UAV-CCIO scheme is better than the existing scheme.**

***Index Terms*— UAV, intelligent connected transportation systems, 6G communications, Internet of Vehicles, task offloading, edge server.**

I. Introduction

INTELLIGENT Transportation Systems with 6G networks is expected to provide reliable and efficient large-scale connectivity for vehicles. These vehicles are connected to the network via wireless communication technology to form the Internet of Vehicles system which can make effective use of all vehicles and environment information to provide users with various services, improving the security, intelligence and efficiency of the intelligent transportation system. With the wide application of Artificial Intelligence (AI) in intelligent transportation systems, coupled with the high complexity and high data volume, more and more data-intensive and computing-intensive applications are deployed at the edge of networks. These applications interact heavily with the Internet of Things (IoT) devices in ITS [1]–[3].

At present, ITS are developed rapidly with billions of IoT devices deployed in various applications for data sensing and task processing [4]–[6]. The number of devices connected to IoT has exceeded 20 billion and is growing at an even faster rate [7]–[9]. As a large number of IoT devices are deployed at intelligent transportation systems, the edge network possesses huge computing and storage capacity [10]–[15]. At the same time, these massive IoT devices generate tens of TB of data per day for connected vehicles to provide services [16]–[18]. In addition, as mentioned above, more and more computing-intensive applications such as machine learning and virtual reality are interacting with IoT devices [19], [21], especially on a 6G network. As 6G networks improve communication, IoT devices are getting more and more heavily loaded. Relatively speaking, resource-constrained IoT devices do not have enough computing capability to satisfy the computational requirements of these types of tasks [4]–[8]. In this case, edge computing and fog computing are the better solutions in which tasks are offloaded to the proximity of the edge servers which have sufficient computing power [11]–[13]. Edge computing and fog computing form a new computing model based on the current network [7], [13], [17]. Task offloading is an effective computing method that can be used to use computing resources of the edge devices and provide services in time with reduced accesses to the cloud and save time and devices' energy effectively, extended devices' lifetime [22], [23]. In this computing model, computing and data processing at intelligent transportation systems are carried out on the edge server such as Vehicular Edge Server (VES), avoiding the energy consumption and bandwidth utilization of data uploaded to the cloud as well as the poor QoE situation, thus promoting the development of the intelligent transportation systems [19], [20].

Manuscript received March 10, 2021; revised May 27, 2021 and September 8, 2021; accepted October 20, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 62072475 and Grant 61772554. The Associate Editor for this article was S. Mumtaz. *(Corresponding author: Neal N. Xiong.)*

Run Liu, Anfeng Liu, and Zhenzhe Qu are with the School of Computer Science and Engineeing, Central South University, Changsha 410083, China (e-mail: runliu@csu.edu.cn; afengliu@mail.csu.edu.cn; zhenzhequ@csu.edu.cn).

Neal N. Xiong is with the Department of Computer Science and Mathematics, Sul Ross State University, Alpine, TX 79830 USA (e-mail: xiong-naixue@gmail.com).

Digital Object Identifier 10.1109/TITS.2021.3122567

In ITS, researchers often use vehicles to construct edge server named VES. However, many studies have shown that some VES in intelligent transportation systems cannot be found by IoT devices to offload tasks with a large number of IoT devices in the network. Moreover, there are a large number of IoT devices that cannot even communicate with the Internet, let alone task offloading [24], [25]. Therefore, for the realization of ubiquitous sensing and computing to provide better services to ITS, many researchers have proposed a method to construct Computing-Communications Edge Servers to solve this problem [4].

An effective method for CCES in ITS is to adopt the Unmanned Aerial Vehicles as CCES to offload tasks for IoV and IoT [26]–[28]. In UAV as CCES, one of the important factors that need to be taken into account is energy consumption. On the one hand, IoT devices tend to be powered by batteries and thus have extremely limited power [4]. If the IoT device runs out of power, it will not be able to provide better service to the IoV system in ITS. Communication is the most energy-intensive part of IoT devices. In 6G network, the communication bandwidth and the communication frequency increase, and the amount of data exchanged becomes larger. Generally speaking, tasks are associated with a certain amount of data. Moreover, different tasks require different amounts of data when their offloading is requested. Some data-intensive tasks require a large amount of data to be uploaded when offloading. Tasks belonging to these types of application include virtual reality, which requires uploading a large amount of data for calculation [10]. However, some tasks with a small amount of data only need to upload a small amount of data in offloading [17]. For IoT devices, their energy consumption is mainly determined by the amount of data to be uploaded [29], [30]. Therefore, careful consideration should be given to the amount of data of tasks to maximize energy saving in task offloading, which has not been fully studied in previous studies. In order to overcome the shortcomings mentioned above, a UAV-enabled Computing-Communications Intelligent Offloading scheme is proposed to energy-efficient offload tasks for minimizing the overall energy consumption of IoT devices in the intelligent transportation system interacting with connected vehicles in IoV system under 6G network. The main contributions of this article can be summarized as follows.

(1) An UAV-enabled Computing-Communications Intelligent Offloading scheme is proposed to offload tasks energy-efficiently. It minimizes the overall energy consumption of IoT devices in the intelligent transportation system interacting with connected vehicles in the IoV system under 6G network. In UAV-CCIO scheme, tasks with a large amount of data are selected as TGNs and tasks with a small amount of data are transmitted to TGNs through in-network routing so that the energy consumption is small. The in-network task route minimizes energy consumption of the systems and improves network life. However, due to the clustering of tasks to TGNs, the required flight points of the UAV are greatly reduced, which effectively reduces the flight distance of the UAV and thus reduces the energy consumption of the UAV. At the same time, as tasks are concentrated in TGNs, communication competition and instability in previous strategies are reduced and the efficiency of task offloading is further improved.

(2) An algorithm for selecting TGNs based on the data volume of IoT devices' tasks is proposed to prevent a large amount of data from being transmitted between IoT devices and reduce the energy consumption of ITS. Then, a new load balancing scheme is proposed to balance the energy consumption of task transmission between TGNs which can effectively reduce the energy consumption of the system.

(3) Simulation experiments show that UAV-CCIO scheme can reduce the energy consumption by more than 75.53%, increase the network life by more than 2.04 times, and balancing the load of the entire network. In addition, UAV-CCIO is superior to other schemes in some special cases. At the same time, we point out the potential problems of UAV-CCIO and the work to be carried out in the future.

The rest of this paper is organized as follows. The related works are given in Section II. The system model and problem statements are presented in Section III. In Section IV, a novel UAV-enabled Computing-Communications Intelligent Offloading scheme is established to minimize the energy consumption of task offloading. The experimental results are reported in Section V. We conclude in Section VI.

II. Related Work

Task offloading in edge computing has attracted extensive attention of researchers [4], [21], [28]. In intelligent transportation systems, task offloading is mainly to solve the current IoT devices' computing resource and energy constraints and unload their tasks into edge servers, such as VES, with sufficient computing power and energy [31]–[33]. On the one hand, IoT devices can offload tasks into VESs and thus reduce their load and save energy [31], [32]. On the other hand, the tasks are carried out on the VES with more powerful computing power, so the result can be obtained faster, which reduces the time needed for the calculation result and improves the application QoE [4]. As for VES, on the one hand, it makes full use of its computing and energy resources, enabling tasks to carry out operations at the edge of the network [10], thus avoiding the huge energy consumption and bandwidth consumption in the process of sending the results of task and computation to the cloud and returning them to the user through long routes in traditional cloud computing [17], [18].

According to the differences of VES platform studied, task offloading in edge can be divided into the following two categories. One is the task offloading model between traditional IoT devices and VESs [31]–[33]. In such a network, IoT devices, VESs, and clouds can communicate directly with each other. Thus, there is no need to consider constructing Computing-Communications Edge Servers in such a network. On the other hand, IoT devices themselves cannot directly communicate with VESs, so CCES need to be constructed to perform task offloading. With the widespread deployment of IoT devices and human exploration into unknown areas, a large number of IoT devices cannot establish direct communication links with the Internet [4]. Therefore, the on-demand construction of CCES becomes most important for these IoT devices.

There have been a large number of researches on traditional edge computing, which initially focused on the architecture of network [10], [17], [34]–[36]. Generally speaking, the three-layer network architecture has been approved by researchers. Such a network system is divided into three layers: IoT devices layer, Edge layer, and Cloud layer. Although the three-layer network architecture is the overall structure, in the actual network, there are still corresponding changes between different layers, and the objects of offloading are also different [10], [17].

Subsequently, load balancing of tasks is the most important research content of edge computing. Within the network, there are multiple IoT devices and VESs. Different IoT devices have different tasks, and different edge servers have different workloads with different tasks being performed. When a task is simply abstracted as a load, it makes sense to assign tasks to VESs with light loads. Because, in this case, the task can be implemented smoothly, it needs to be completed in a short time, which is able to reduce the computation delay. Jia *et al.* [30] researched the problem of load balancing between edge servers, they proposed a scheduling algorithm, the computing tasks from the overloaded server are transferred to other less load of servers so that each edge server to good use. However, in practice, tasks often require multi-dimensional resources, so the task offloading problem can be turned into a multi-dimensional resource target optimization problem. Therefore, in many studies, task offloading is often combined with the optimal allocation of resources. At the same time, the communication resources required by task offloading and the communication delay caused by it should also be considered in task offloading. Obviously, tasks should be computed as locally as possible to reduce the communication costs and delays associated with non-local calculations. In this regard, Li *et al.* [31] studied the compromise between computing delay and communication in multi-layer edge cloud architecture, proposed a nonlinear integer programming problem, and derived the selection strategy and computer resource allocation. Binary unload problem [31] and non-binary unload problem [32] can be divided according to whether tasks can be split during offloading. In the binary unload problem, tasks are inseparable and can only be unloaded as a whole. Such a load balancing problem is similar to the backpack problem. Instead of binary unloading, tasks can be further subdivided and a portion of the task can be offloaded to the edge server. Vehicle Edge Computing (VEC) [31]–[33] aims to solve the problem of increased computing delays caused by the growing number of vehicle applications.

A basic premise of these IoT devices is that they are capable of communicating directly with edge servers [37], [38]. However, as mentioned above, this assumption is not always true in the actual network [39]–[42]. Therefore, it is sometimes necessary to construct a computing-communications platform. VEC is a well-researched network. In VEC, Mobile Vehicles (MVs) can be used as mobile edge servers due to their rich computing capacity. Roadside Smart Devices (RSD) has limited computing power and there is no direct communication with the Internet in parts of RSD. But RSD can be their own task offload to MVs when the MVs move to their communication range. This kind of research also is more, for example, Wang *et al.* [32] proposed a cloud-based layered VEC offloading framework and adopting the Stacklberg game theory method to design the optimal multistage offloading program. Li *et al.* [31] proposed a joint best VEC server selection and offloading algorithm.

Computing-Communications Edge Servers in the air using UAVs is a low-cost and effective approach for IoT devices deployed for specific purposes and without Internet communication links [4], [24], [28]. In such methods, the UAVs act as edge server flies over the IoT devices to establish communication with them. Meanwhile, UAVs themselves have a processor with large computing power, so it can be used as UAV-Edge Server for task offloading [4]. At present, a data collection network using UAVs is widely used [45]. In such a network, IoT devices can be deployed as needed in areas requiring monitoring [45]. While UAVs act as data mules fly to IoT devices to collect data as needed [4]. Hu *et al.* proposed a MVs joint UAVs joint number collection strategy [24]. In their proposed strategy, MVs are used for data collection at the beginning of data collection, and then UAVs are used at the later stage of data collection for data that cannot be collected by isolated IoT devices or MVs [24]. Since MVs have a very low cost for data collection, and UAVs have a fast speed for data collection, the combination of the two can be used for data collection at a low cost. Actually, data collection and task offload are essentially similar, data collection can be thought of as a simple task to compute the IoT devices only need to be a task that is data upload to the UAVs. Dai [45], this paper proposed that an UAV is adopted to improve data collection based on the clustering strategy, namely to wireless sensor network is divided into several clusters, clusters in the data sent to the cluster head, in the end, UAVs collects data for each CH. Although the research and the work of this paper have some similarities, the complexity of this study is far more than the data collection network. Guo and Liu *et al.* [4] the study and research of this paper have some similarities, namely USES the UAVs are edge server. Guo and Liu [4] study did not consider the task of the properties of the problem, without considering the task data property, therefore, is simpler than the research of this paper.

III. System Model and Definitions

In this paper, we define the working cycle of UAV as the time it takes for UAV to visit all IoT devices to collect data. In this section, communication model, system energy consumption model and network internal transmission energy consumption model are established in a working cycle of the UAV. For convenience, the key symbols in this article are arranged in Table I.

A. Communication Model

Let the set of IoT devices in the network be $V_k = \{v_1, v_2, \ldots, v_k\}$, where $v_i \mid i \in \{1, 2, \ldots, k\}$ represents the ith node. Communication between wireless sensors and UAV is the main way of data transmission energy consumption, and the related physical quantities include Euclidian distance,

TABLE I
THE EXPLANATION OF SYMBOLS

Symbols	Explanation
V_k	The kth IoT device
H	The altitude at which the UAV flies as it collects data
$d_k[i]$	The distance of the UAV to the kth sensor at position i
$S_k[i]$	Communication channel power gain of the kth sensor at position i of the UAV
h_0	Channel power gain at 1m reference distance
$C_k[i]$	The maximum rate of information transmission at the kth IoT device
$C_{uav}[i]$	The maximum rate at which the UAV transmits information at position i
W	Channel bandwidth
N	White Gaussian noise
P_{uav}	UAV communication power
P_k	Communication power of the kth IoT device
E_{fly}	Flight energy consumption of UAV
c	Coefficient of flight energy consumption
d_{ij}	The distance from node i to node j
$t_k[i]$	Information uploading time of the kth IoT device
U_k	The number of cycles required for the CPU on the kth IoT device to process a bit
M_k	The amount of data sent by the kth IoT device
E_k^{ol}	The amount of energy the UAV to receive data from the kth sensor
E_k^{uav}	The energy consumption of data uploading for the kth sensor
E_k^{tr}	Energy consumption for data transmission from the kth IoT device to the TGN
E_k^{loc}	The local processing energy consumption of the kth IoT device
β_k	The energy consumption coefficient of the local treatment of the kth IoT devices
f_k	The computing power of the kth sensor
E_{wsn}	Total energy consumption of WSN
E_{uav}	Total energy consumption of UAV
P_i	The probability of the new solution being accepted
T_i	The value in the i state in the simulated annealing algorithm
d_0	Threshold of data transmission distance
E_{elec}	The transmission line loss
E_{dev}	The energy consumption of the transmission circuit per bit
E_t	The internal network energy consumption
E_r	The energy consumption of the transmission circuit
h_0	The channel power gain
N	The Number of IoT devices
μ	The mean of the amount of data gathered by each TGN
M_k	The data volume of task k
σ^2	Standard deviation between TGNs

communication channel power gain, transmission bit number, data volume, bandwidth, etc. First of all, we assume that the horizontal height of the geographic location of the wireless sensor network has no influence on the energy consumption of the data collected by the UAV, and assume that the flight altitude when the UAV collects data is H. When the UAV collects data for the kth sensor υ_k, the UAV's spatial coordinate is $R(x_i, y_i, H)$. Use $d_k[i]$ to represent the distance of the UAV to the kth sensor at position i. So the distance is

$$d_k[i] = \sqrt{H^2 + (x_i - x_k)^2 + (y_i - y_k)^2}, \tag{1}$$

where the coordinate of the kth sensor υ_k is $(x_k, y_k, 0)$, and the metric space is Euclidean space. At this point, the power gain of communication channel is

$$S_k[i] = \frac{h_0}{d_k^2[i]}, \tag{2}$$

where h_0 is the channel power gain when the reference distance is 1 m.

The communication rate between UAV and sensor is related to the power of communication equipment. Assuming that the communication power of UAV and sensor υ_k are P_{uav} and P_k respectively, the channel bandwidth is W, and the Gaussian white noise in the channel is N, it can be seen from Shannon's formula that the maximum rate C of information transmission is

$$C_k[i] = Wlog_2\left(1 + \frac{P_k S_k[i]}{N}\right), \tag{3}$$

$$C_{uav}[i] = Wlog_2\left(1 + \frac{P_{uav} S_k[i]}{N}\right). \tag{4}$$

B. System Energy Consumption Model

Wireless Sensor Networks (WSN) and Unmanned Aerial Vehicles constitute a system. The energy consumption of the system is divided into the energy consumption of the UAV flight, the energy consumption of the UAV receiving data, the energy comsumption of IoT devices send data to the UAV, the energy consumed when the IoT devices transmit data to the TGN in a single hop or multiple hop manner.

1) The Energy Consumption of the UAV Flight: After the UAV flies over the node requesting edge computing service, it will transmit the collected data to the edge computing server.

Assuming that the motion state of the UAV can be regarded as uniform motion in this process, the flight energy consumption of the UAV is related to the movement speed and flight distance of the UAV.

Therefore, the flight energy consumption per unit distance of UAV is represented by the flight energy consumption coefficient c, and the flight energy consumption of UAV can be expressed as

$$E_{fly} = c\sum d_{ij}, \tag{5}$$

where d_{ij} represents the distance between node i and node j.

2) Energy Consumption for Receiving and Sending Data: For UAV and sensing devices, receiving data and sending data happen simultaneously. Assuming that sensing device k sends data in the size of M to the UAV at position i, the information upload time is

$$t_k[i] = \frac{U_k M_k}{C_k[i]}, \tag{6}$$

where U_k represents the number of cycles required for the CPU of the kth sensor to process a bit. Therefore, the energy consumption of data uploading and data receiving by UAV are

$$E_k^{ol} = P_k t_k [i]. \tag{7}$$
$$E_k^{uav} = P_{uav} t_k [i]. \tag{8}$$

3) Local Processing Energy Consumption: if a sensor adopts the strategy of calculating data locally, its energy consumption depends on the amount of data and the energy consumed by the processor to process each byte. β is the local processing energy consumption coefficient of the kth sensor, which is related to the structure of the sensor chip. f_k is the computing capacity of the kth sensor, then the local processing energy consumption is

$$E_k^{loc} = \beta_k U_k M_k (f_k)^2. \tag{9}$$

4) Internal Network Energy Consumption: The following is the constraint of energy consumption within the wireless sensor network

$$\begin{aligned} E_t &= \begin{cases} M_k E_{elec} + M_k \varepsilon_{fs} d^2, & if\ d \le d_0, \\ M_k E_{elec} + M_k \varepsilon_{amp} d^4, & if\ d > d_0, \end{cases} \\ E_r &= M_k E_{dev}. \end{aligned} \tag{10}$$

We adopt the classic network energy consumption model, in which the energy consumption of transmitted data and the energy consumption of received data are given in (10). E_{elec} on behalf of the transmission line loss, according to the distance between the transmitter and the receiver, in the model we adopted for space at the same time (d ^2 power loss) multiple failure (multi - path fading) ^4 power loss (d) channel model. ε_{fs} and ε_{amp} are the energies required for power amplification of the two models. M_k stands for data volume. d_0 represents the Threshold distance to distinguish the two models.

5) Total Energy Consumption of Wireless Sensor Network: The total energy consumption of wireless sensor network should be the sum of local processing energy consumption of sensor network, energy consumption of data transmitted to offload node and energy consumption of offload sensor uploaded to UAV. Therefore

$$E_{wsn} = \sum\nolimits_k (E_k^{ol} + E_k^{tr} + E_k^{loc}). \tag{11}$$

6) Total Energy Consumption of UAV: The total energy consumption of aircraft should be the sum of flight energy consumption and data acquisition energy consumption. Therefore

$$E_{uav} = E_{fly} + \sum\nolimits_k E_k^{uav}. \tag{12}$$

IV. Our Proposed UAV-CCIO Scheme

In this section, we first describe the problem, then introduce the research content of this article, and finally explain the details of UAV-CCIO algorithm.

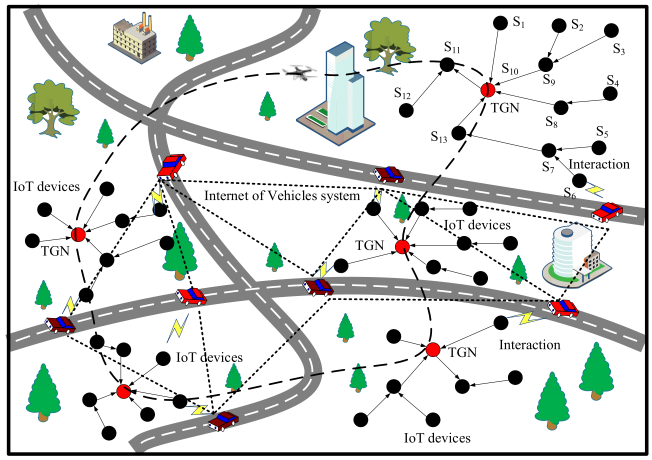


Fig. 1. Network architecture.

A. Problem Description

A large number of IoT devices are deployed in 6G networks, and these devices merge the connected vehicles in the IoV systems to constitute an intelligent transportation system. Connected vehicles in driving form an Internet of Vehicles system that connects to each other to exchange or share data. Within ITS, connected vehicles interact with nearby IoT devices, generating data-intensive or computationally intensive tasks. At the same time, the UAVs and edge servers are deployed in this network to provide task offloading for IoT devices. Suppose there are k IoT devices in the rectangular region with side length l, all of which constitute a wireless sensor network. In this sensor network, there is an edge computing server, which provides edge computing services for IoT devices. The UAV needs to fly over IoT devices to collect data and bring it back to the edge computing server for calculation. In the sensor network, there are various tasks undertaken by IoT devices, including tasks with large amount of data and large amount of computation. The goal of this paper is how to design an offload strategy in such a complex wireless sensor network to save system energy consumption and improve the life of wireless sensor networks. In order to achieve the goal, UAV-CCIO algorithm is proposed in this paper.

In the traditional offload algorithm, IoT devices are divided into clusters according to the deployment situation, and the nodes located in the center of the cluster are set as cluster-head nodes (marked in red in Fig. 1). Other IoT device nodes send data to cluster-head nodes in the way of single hop or multiple hop. Then the UAV goes through the cluster-head node in turn to collect the data of each cluster. After completing the collection mission, the UAV takes the data back to the edge computing server for further processing. For this paper, we focus on the process of tasks being offloaded to the edge server after they are generated. Because the results returned are too small relative to the amount of data the task itself has.

However, the traditional algorithm does not consider the influence of the amount of data and computation of each IoT device on energy transmission. When the IoT device node transmits data to the cluster-head node, it needs to consume

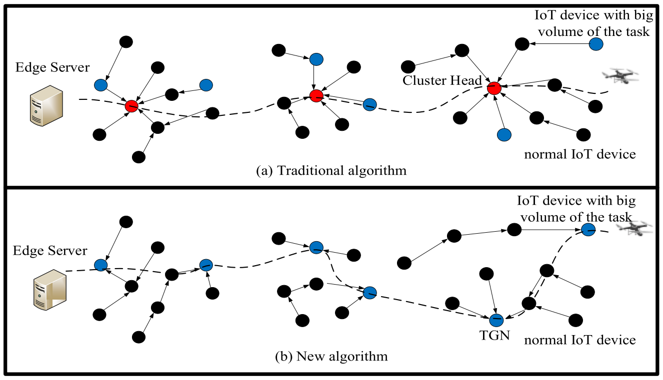


Fig. 2. Algorithm illustration.

energy, and the amount of energy is related to the data volume of the IoT device's task. Since the data transmitted by the IoT device is the most energy-consuming, if a task of an IoT device has a large amount of data and a large amount of data is always transmitted to the cluster-head node, this IoT device will run out of energy too early due to a large amount of energy consumption, which will reduce the life of the whole wireless sensor network. Therefore, if we reasonably plan the data transmission path and select the data collection node according to the data volume and calculation amount of the IoT device, we can improve the use efficiency of energy and improve the life of the sensor network. To solve this problem, the UAV-CCIO algorithm is proposed in this paper. The strategy diagram of this algorithm is shown in Fig. 2.

As can be seen in Fig. 2, there are IoT devices with large data volumes in each small network. These nodes are represented by blue nodes in the diagram. The black nodes in the figure are ordinary nodes with a small amount of data and a large amount of computation or other situations.

In Fig. 2(a), the UAV needs to pass through the cluster-head node in a certain order to collect data, and the IoT devices of each cluster need to transmit data to the cluster-head node according to the direction of the arrow. It can be found that in the cluster with nodes with large data volume, the node may need to transfer a large amount of data to the cluster-head node, which will consume a large amount of energy in the process, so the energy utilization is inefficient and the network life is low.

In Fig. 2(b), the UAV-CCIO algorithm determines the TGNs according to the data volume of IoT devices' tasks and also determines the data transmission direction according to the relative positions of TGNs and ordinary nodes. It determines the data transmission path. The data from ordinary nodes are transmitted to the big data node according to the arrow direction, and the UAV passes through the big data node in sequence to collect the data. In this case, IoT devices with large data volume tasks does not need to transmit large amounts of data to other nodes. Therefore, data transmission energy consumption is reduced, energy utilization is efficient, and network life is increased.

B. Details of UAV-CCIO Algorithm

In order to solve the problems mentioned above, UAV-CCIO algorithm is proposed. UAV-CCIO algorithm includes the algorithms of determining the TGNs, determining the transmission path from the ordinary nodes to TGN, generating the UAV flight path, load balancing.

1) Determining the TGNs: In the network shown in Fig. 1, the size of the amount of data for all nodes is given. Because different IoT-IoV systems have different functions and different nodes have different computing tasks, the amount of data generated by different nodes in different situations is also different. Therefore, it is impossible to use a unified standard to measure the size of node data volume and then judge whether it is a node with a large data volume. According to the size of the data volume of all nodes in the network, this algorithm determines which nodes are big data volume nodes. The specific process of the algorithm is as follows.

Assuming that the data volume of each common node is normally distributed, if there is an IoT device with a large data volume task in the wireless sensor network, the data volume of this task is greatly different from that of other common tasks.

As the data volume of this data volume and other common IoT devices is a whole, but it shows extreme performance in the whole wireless sensor network, so it can be determined whether the data volume value is an outlier through certain judgment criteria. If it is an outlier, then this node is the IoT device of the big data task. If it is not an outlier, it is not the IoT device of a large data volume task. The judgment criterion needs to select an appropriate method according to the actual situation. For the case discussed in this paper, we use the binary clustering method to judge whether it is an outlier. Then judge whether it is an IoT device of a large data volume task.

Clustering algorithms are often used to reduce dimensions [60]. In this paper, the idea and principle of binary clustering are: The data volume of the IoT device of the big data task is obviously different from the data volume of the task of the ordinary node, which leads to the obvious stratification of the data distribution, part of which is the data volume of the big data task, part of which is the data volume of the ordinary task. Therefore, the binary clustering method can be used to divide the data volume of all tasks into two clusters, in which the data volume of ordinary tasks forms a cluster and the data volume of big data tasks forms a cluster. In this way, by clustering tasks according to the data volume, we can determine which nodes are IoT devices of the task with large data volume.

First, the tasks' data of all nodes are taken as objects, and two objects are selected as the initial clustering centers, and the unit is Mb, and then the clustering center is reclassified according to the mean value of the task data of the IoT device and the distance of the clustering center. Then, the distance between each cluster center and the central objects is recalculated until the cluster center no longer changes. This division minimizes the following equation:

$$E = \sum_{j=1}^{k} \sum_{x_i \in w_j} ||x_i - m_j||^2,$$

where x_i represents the position of the data volume of the ith sensor on the coordinate axis, and m_j represents the

Algorithm 1 Algorithm for Determining TGNs

Input: The amount of data for all tasks.
Output: The TGN serial number.
The amount of data of the ith IoT devices is $M[i]$;
Arbitrary selection of two data volumes *M[m], M*$[n]$ as the initial cluster center and record the values *center1= M[m], center2= M[n]*;
Do
  For $i = 1$ to the number of IoT devices N **Do**
    Calculate the distance $d_{ct1}[i]$ from the cluster center1;
    Calculate the distance $d_{ct2}[i]$ from the cluster center2;
    If $d_{ct1}[i] < d_{ct2}[i]$ **Do**
      Assign the i-th data to the cluster center1;
    Else
      Assign the i-th data to the cluster center2;
    End if
  End for
  Calculate the number of objects *num1* and calculate the sum of all values *sum1* at the center of the cluster1;
  Calculate the number of objects *num2* and calculate the sum of all values *sum2* at the center of the cluster2;
  New cluster centers are *newcenter1* and *newcenter2*;
  *newcenter1= sum1/num*1;
  newcenter2= sum2/num2;
Until (*newcenter1,newcenter2*) is equal to (*center1,center2*)

position of the jth cluster center. The above procedure is represented by pseudo code as shown in Algorithm 1.

2) Determining the Transmission Path: We use Algorithm 1 to determine TGNs in the wireless sensor network. According to UAV-CCIO, the IoT devices of these large data volume tasks are used as data collection nodes. The next step is to determine the path for the remaining IoT devices to pass the data to the TGNs. Since the energy consumption of other nodes when transmitting data to the TGNs is related to the transmission distance and the amount of data transmitted, different data transmission paths will lead to different energy consumption of nodes. If the data transmission path can be chosen reasonably, the energy consumption of sensor network can be reduced.

The algorithm process is shown in Fig. 3. In Fig. 3(a), the blue nodes are the big data volume nodes determined by Algorithm 1 and are the TGNs in the network. The tasks of other nodes should be gathered at this node. Black nodes are nodes with a small amount of data. Initially, there was no planned path for data aggregation, so there were no arrows between the nodes. The first step is shown in Fig. 3(b). The black node closest to the TGN is selected. Since this node is the point closest to the blue node, its particularity determines that its data transmission path must be to directly transfer data to the blue node. The second step is shown in Fig. 3(c).

Algorithm 2 Data Transfer Path Algorithm

Input: The position of the nodes, the serial number of TGNs.
Output: The data transfer path.
For each big data sensing devices Bi **Do**
  Calculate the distance between the remaining points and this point and put the results in Bi_dis separately;
  Set processed[Bi]=1;
End for
While not all of the normal sensing devices are processed **Do**
  For i =1 to the number of big data sensing devices
    For each normal sensing devices Ni sorted by Bi_dis in ascending order **Do**
      If processed[Ni]=0 **Do**
        Set processed[Ni]=1;
        Calculate the energy of transmitting the data directly to each Bi E_direct;
        For each normal sensing devices Nj **Do**
          If processed[Nj]=1 **Do**
            Calculate the energy of multihop data E_multihop;
            If E_multihop < min(E_direct) **Do**
              Set Belongto[Ni]= Nj;
            Else
              Set Belongto[Ni]= Bi;
            End if
          End if
        End for
      End if
    End for
  End for
End while

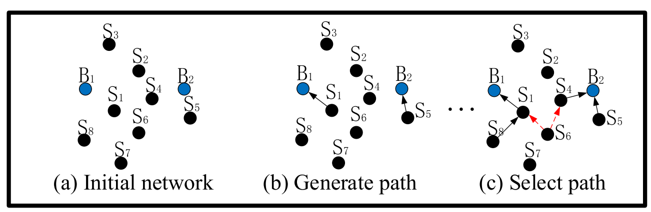


Fig. 3. Algorithm illustration.

The unplanned black node nearest to the TGN has two data transmission paths. The first is to transfer the data to the blue node B1 through S1 in the way of multiple hops, with energy consumption of E1. The second method is to transfer the data to the blue node B2 through S4 in the square of multiple hops, with energy consumption of E2. The magnitude of E1 and E2 is the sum of energy consumption for each jump in the process of multiple jumps. By comparing the energy consumption of the two, the most suitable data transmission path is chosen. The third step is to repeat the second step until the transport path for all nodes is planned.

The above procedure is by pseudo code as shown in Algorithm 2.

Algorithm 3 Algorithm for Generating the UAV Flight Path

Input: The position of TGNs.
Output: The UAV flight path.

Initialize the iteration value T and threshold Tt, initialize the access sequence $A[N]$, Coefficient of iteration value descent a;
Calculate the current access sequence of UAV flight energy consumption $E\_current$;
while $T > Tt$ **Do**
    perturb access sequences by using two - or three-switching methods and record the new access sequences $A\_new[N]$;
    Calculate UAV flight energy consumption of the new access sequence E_new;
    If $E\_new < E\_current$ **Do**
        $E\_current = E\_new, A[N] = A\_new[N]$;
    Else
        Accept the new solution by the Metropolis criterion;
    End if
    $T = T^{*}a$;
End while

3) Generating the UAV Flight Path: The choice of the flight path is important for the energy consumption of the whole system. The different paths make the UAV fly with different energy consumption, and the overall energy consumption of the system will also change. Therefore, how to choose the most appropriate path to minimize the overall energy consumption is the goal of this algorithm.

The problem of choosing the optimal path according to the IoT device of the large data volume task is similar to the TSP problem of the travel agent. The UAV must pass through IoT devices for all big data tasks. The UAV needs to find a flight path after the IoT device of a given big data task so that the sum of energy consumption of UAV and sensor network is minimum after the UAV passes all the big data node. Obviously, this problem is NP-complete, that is, the exact value cannot be solved by a precise algorithm, but the optimal value can only be solved by an approximate algorithm. The simulated annealing algorithm in the heuristic algorithm is used in this paper. Because it can accept the flight path that increases the energy consumption of the system in accordance with the probability, the result breaks out of the local optimal solution, so it is possible to converge to the global optimal solution eventually.

Although sometimes the problem may be non-convex, resulting in the final result may not be the global optimal value, but after the simulated annealing algorithm, through setting the appropriate parameters, the result can also get a satisfactory value.

The main process of the algorithm is as follows: First, the coordinate of the IoT device of the task of input large data volume. These IoT devices act as data collection nodes, and the UAV needs to receive data from them. Then the access queue of TGN is initialized and the energy consumption of the access queue is calculated. Next, the access order in the access queue is disturbed by using the two-switching method or three-switching method, and the energy consumption of the disturbed access queue is calculated. By comparing the energy consumption before and after the disturbance, the new solution is accepted according to the Metropolis criterion. After a certain number of iterations, the access queue gradually converges to the optimal solution. At this time, the UAV accesses TGN in sequence according to the access order, and the path formed is the optimal flight path. The pseudocode of this algorithm is as shown in Algorithm 3.

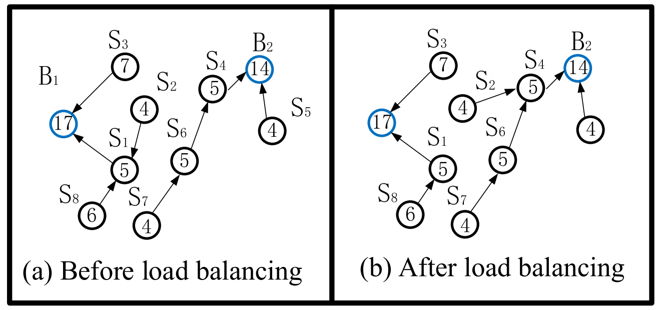


Fig. 4. Illustration of Algorithm 4.

4) Load Balancing: The function of TGNs is to gather the data of ordinary nodes around and then send the data to UAV. Compared with other nodes, which simply pass data to TGN, the TGNs to collect data and passes data to the UAV. Therefore, in comparison, the energy consumption of TGN is greater than that of surrounding nodes. Under the condition that the battery capacity of all nodes is fixed, the energy consumption of TGN often determines the life of the whole wireless sensor network. Therefore, balancing the load of each TGN and appropriately adding TGNs can average and reduce the energy consumption of each TGN, thus improving the life of the wireless sensor network.

Before we propose the load balancing algorithm, we need to describe the load balancing degree. In this paper, we use the standard deviation of the data amount of each TGN to describe the load balancing degree. Standard deviation is often used to indicate the degree of dispersion between data and it is highly explainable, which brings convenience to the analysis of the experiment part. The calculation formula of the standard deviation of the data volume is as follows:

$$\sigma = \sqrt{\frac{\sum (X-\mu)^2}{N}},$$

where μ is the average amount of data borne by each TGN, and N is the number of IoT devices. The greater the standard deviation, the greater the data volume difference between TGNs, the greater the load difference, and the greater the energy consumption difference. So, the load balancing algorithm is to solve such a problem: The task data volume and data transmission path of each IoT device are given. How to fine-tune the data transmission path of nodes to obtain a suboptimal solution with a smaller standard deviation while maintaining such network topology as far as possible.

Fig. 4 is the illustration of the data transmission path optimization algorithm for load balancing. Fig. 4(a) is the

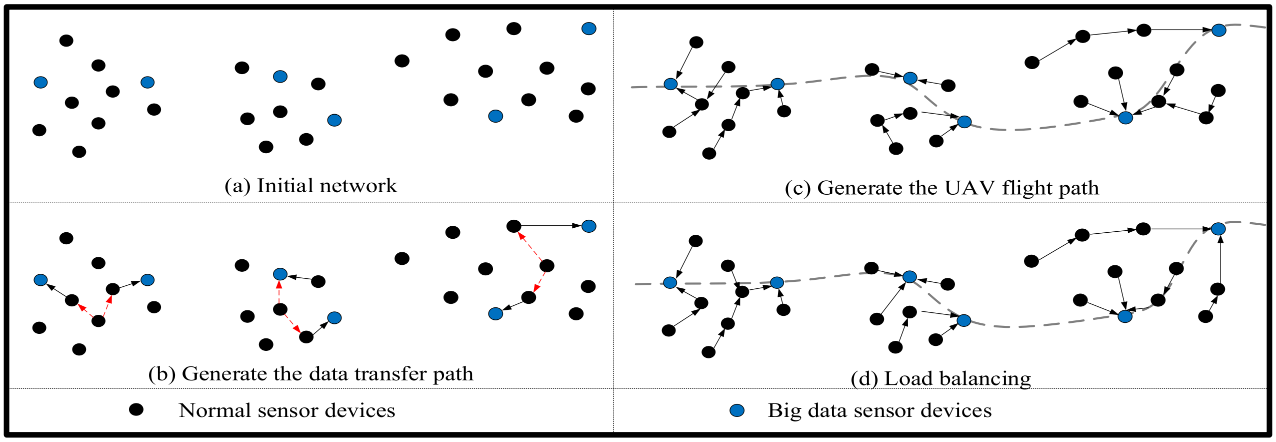


Fig. 5. Illustration of UAV-CCIO.

path of data transmission with minimum energy consumption generated by Algorithm 2. The number in the node represents the amount of data in the node. The amount of data transmitted by B_1andB_2. TGN to UAV is 39 and 32 respectively, and the standard variance is 4.9497.

The optimization results of the data transmission path optimization algorithm after load balancing are shown in Fig. 4(b). Again, data are aggregated to B_1, B_2, but the data transmission path of node S_2 is slightly adjusted, and the path $S_2 \rightarrow S_1 \rightarrow B_1$ is modified to $S_2 \rightarrow B_2$. In this way, the amount of data transmitted by B_1, B_2 TGNs to UAV is changed to 35 and 36, with variance of 0.7071.

Algorithm 4 Load Balancing Algorithm

Input: The amount of data for all tasks, the data transfer path.
Output: The new data transfer path.

```
Calculate the amount of data collected by each TGN;
Calculate the initial standard deviation of the system s0;
Locate the outer sensing device of each TGN;
For each TGN Do
  Find the TGN with the smallest load;
  For each outer sensing device Do
     Calculate the standard deviation of the system s1
     when the load is transferred to the current
     sensor device;
  If s1 < s0 Do
     Modify the data transfer path;
  End if
  If s1> s0 Do
     Maintain the original data transfer path;
  End if
End for
End For
```

After such fine-tuning, the topology of the network is not changed much, and the standard deviation of the load of the network is reduced so that the energy consumption between TGNs is closer.

Algorithm 5 UAV-CCIO Algorithm

Input: The position of the nodes, the amount of data for all tasks.
Output: The data transfer path and the UAV flight path.

```
Determining TGNs by using Algorithm 1;
Determining the path for data transmission by using
Algorithm 2;
Determine the optimal flight path by using Algorithm 3;
Use Algorithm 4 to balance the load;
```

5) UAV-CCIO Algorithm: A schematic diagram of UAV-CCIO algorithm is shown in Fig. 5. In Fig. 5(a), each node represents the IoT device in the network. At the beginning, when the connected vehicle in the intelligent transportation system generates tasks, it interacts with the IoT device through the communication link. The IoT device then sends information about the task's data volume to the entire network via appropriate routes, and then Algorithm 1 determines which nodes in the network are the IoT devices of the large data volume task.

After determining the IoT devices of the big data task, these nodes become TGNs. The surrounding nodes should transfer the data to TGNs, and the UAV only needs to collect the data of TGNs. Next, Algorithm 2 shown in Fig. 5(b) determines the data transmission path of all nodes around TGN. When multiple paths are available, choose the path that consumes the least energy. When the data transmission paths of all nodes are determined, the flight path of the most energy-saving UAV is determined by Algorithm 3 shown in Fig. 5(c), which goes through TGN in turn, and the UAV collects the data of the node. Finally, the Algorithm 4 shown in Fig. 5(c) optimized the load of all TGNs and fine-tuned the data transmission path of surrounding nodes, so as to minimize the variance among the data volumes of all TGNs and thus achieve load balancing.

V. Performance Analysis

A. Experiment Setup

The wireless sensor network discussed in this paper is a network with IoT devices with large data volume. In this network, each IoT device of big data task acts as TGNs.

TABLE II
PARAMETERS SETTING

Symbols	Value	Symbols	Value
h_0	-50dB	U_k	1000cycle
W	40MHz	P_k	5×10^{-3}W
N	10^{-17}W	P_{uav}	1.5×10^{-2}W
c	10J/m	f_k	1GHz~2GHz
H	15m	β_k	10^{-28}
M_k	1Mb~15Mb		

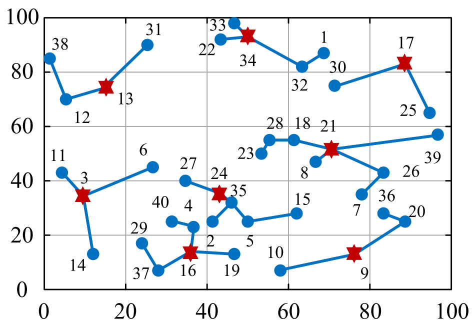


Fig. 6. Illustration of the LEACH algorithm: The red nodes in the figure are cluster nodes selected by traditional algorithms, and these nodes are responsible for collecting tasks from surrounding nodes and uploading them to the UAV. The blue line segment represents the tasks wireless transmission path. These transmission paths connect each node, so that each node is in a connected sub-graph. Since the traditional algorithm does not consider the data volume of each node, there may be IoT devices with large data volume tasks in the connected sub-graph. The presence of these nodes increases the energy consumption of the nodes and thus reduces the life of the wireless sensor network.

The experiment is as follows: N sensors are randomly arranged in a 100×100m two-dimensional topological grid. The value of N is different in different experiments. In each experiment, the position of the sensor is randomly arranged. In order to prevent the occurrence of random outliers in the results from affecting the conclusion, we conducted 30 random tests for each comparison and averaged the results to reduce the influence of outliers on the results caused by the large or small amount of data in the randomly generated tasks. In order to reflect the high efficiency of UAV-CCIO algorithm, we choose the traditional algorithm, the LEACH algorithm for comparison. Since this paper is the first time to propose an offload strategy algorithm combined with IoT device data volume, the selection of LEACH algorithm can well reflect the advantages of UAV-CCIO.

We use Matlab to implement UAV-CCIO algorithm and run simulation experiments on a Dell desktop computer. The CPU of the computer is 2.20GHz Intel Core i5, and it is equipped with 32GB RAM. The experimental parameters are shown in Table II. In order to reflect the role of smart energy-saving offload in reducing system energy consumption, balancing load, and improving wireless sensor network life, we compared this algorithm with LEACH algorithm. Experimental parameters are given in Table II.

B. Energy Consumption in Internal Transmission

First, we generate planar grid topologies randomly and change the number of sensing devices and the density of

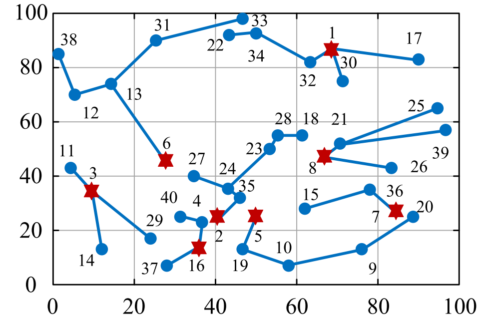


Fig. 7. Illustration of UAV-CCIO algorithm: The red nodes in the figure are the TGNs selected by UAV-CCIO algorithm. Its function is the same as the previous one, which is to collect tasks from surrounding nodes and put them to the UAV. The meaning of the blue line segment is also the same as the previous one. In the figure, all the TGNs are IoT devices of the large data volume task, and they also undertake the task of collecting data from surrounding nodes. In this case, the IoT device of the big data task does not need to transmit data to other nodes, thus reducing a large amount of data transmission energy consumption and thus improving the life of the sensor network.

sensing devices with large data volume tasks to discuss their influence on energy transmission. We respectively take the number of sensing devices as 50, 40, 30, 20, 10, and take the density of sensing devices with large data volume tasks as 30%, 40%, 50%, 60%, and randomly generate 30 planar grid topologies for each situation. We set that the amount of data for the normal sensing devices changes randomly around 1Mb, and the amount of data for the sensing devices for the large-data-volume changes randomly around 15Mb. Fig. 9 shows the difference in energy consumption between LEACH algorithm and UAV-CCIO. As can be seen in Fig. 8, as the number of IoT devices in the systems increases, the network consumes more and more energy for transmission. However, in comparison with the LEACH algorithm, the energy consumption is lower for UAV-CCIO because UAV-CCIO prevents the IoT devices with large data volume tasks from transmitting data to the surrounding IoT devices, thus reducing most of the energy consumption. The comparison shows that for the LEACH algorithm, the higher the density of IoT devices with large data volume tasks in the network, the higher energy consumption. For the UAV-CCIO, the energy consumption is lower, which indicates that the higher the density of IoT devices with large data volume tasks in the wireless sensor network, the more significant the advantage of the UAV-CCIO. According to the results, when the data volume is 15Mb and the TGNs density is 30%, 40%, 50% and 60% respectively, the energy consumption of UAV-CCIO algorithm in the network internal transmission is reduced by 75.53%, 82.70%, 88.36% and 92.20% compared with the traditional algorithm.

We also explored the effect of the data volume difference on the internal energy consumption of the systems by varying the data volume gap between the sensor device with a large data volume task and the normal device with different sensor numbers and sensor densities. To this end, we vary the data volume of the IoT device with a large data volume task from 1Mb to 15Mb, and for each different parameter, we perform 30 experiments and take the average of the data transfer energy consumption, the results are shown in Fig. 8.

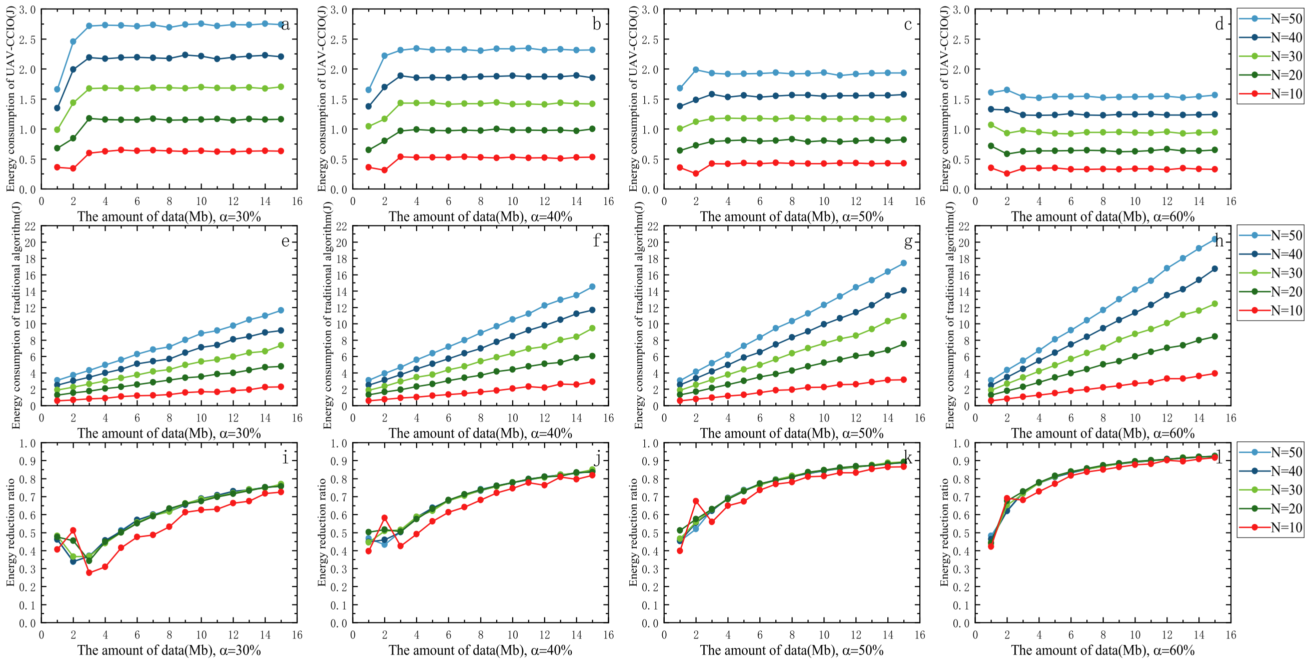


Fig. 8. Energy consumption under different amount of data of IoT devices with large data volume tasks.

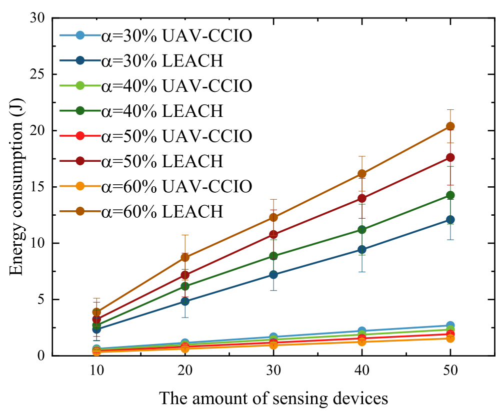


Fig. 9. Energy consumption under different number of sensing devices and different density of sensing devices with large data volume tasks.

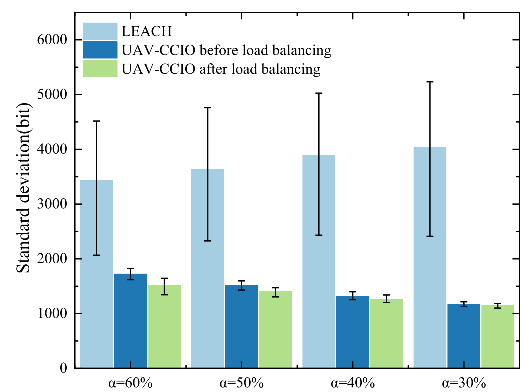


Fig. 10. Standard deviation under different algorithms.

C. Load Balancing

In this part, we also generate 30 random planar grid topologies for each different value, and then obtain the average value of the 30 experiments.

Fig. 10 shows the influence of LEACH algorithm, UAV-CCIO algorithm before load balancing, and UAV-CCIO algorithm after load balancing on the Standard deviation of the data volume of TGNs.

We can see from Fig. 10 that LEACH algorithm has a larger standard deviation of the data volume than UAV-CCIO because LEACH algorithm does not take into account the data volume of the cluster head and the data volume of the nodes around the cluster head node, which leads to the possibility that there are many nodes with large data volume tasks around some cluster head node, making the cluster head node receive the amount of data is large, which in turn increases the deviation of the amount of data received between each cluster head node.

The results show that, when the density of IoT devices of the big data task is 30%, 40%, 50% and 60% respectively, the mean square deviation of the data volume of TGNs decreases by 74.04%, 79.32%, 84.16% and 87.23% compared with the standard variance of the data volume of LEACH algorithm.

UAV-CCIO is based on the size of each node's data to determine the node is as a data aggregation node, so each data aggregation node is a node with a large amount of data tasks, which prevents each node with a large amount of data tasks from transmitting data to other data aggregation nodes, making each data aggregating nodes receive a more even amount of data. Load balancing is a fine-tuning of the data transmission path of the common nodes, which reduces the standard variance of the data volume among the data aggregation nodes in the network, so that the UAV-CCIO (after load balancing) has a lower value than the UAV-CCIO (before

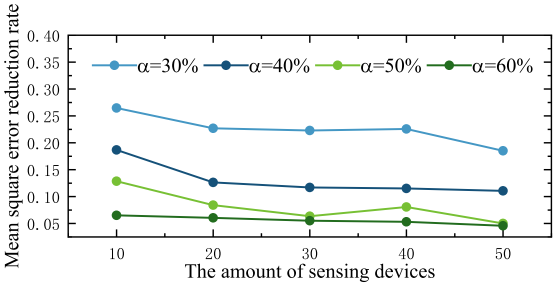


Fig. 11. Mean square error reduction rate.

load balancing). In addition, we can also see from Fig. 10 that increasing the density of nodes for large data volume tasks increase the standard variance of the nodes.

In order to show the effect of UAV-CCIO (ALB), we compare UAV-CCIO (BLB) with UAV-CCIO (ALB) to obtain the standard variance reduction rate for different parameter settings, and the results are shown in Fig. 11. Fig. 11 shows that for different parameter values, the standard variance decreases to different degrees after load balancing. As the node density of large data task increases, the mean square error reduction rate decreases. This is because the node density of the big data task increases, and the average number of nodes to be collected for each TGN decreases. Since the node position arrangement is random, in this case, the probability of a large difference in the amount of data received by each TGN before load balancing is very low, so the value of the standard variance reduction after load balancing is limited. However, when the density of nodes in the big data task is low, the number of nodes around which TGN collects is large, and the probability of collecting data with a large difference is greater. Therefore, after load balancing, the variance decreases significantly. When the IoT device density of the big data task was 30%, 40%, 50% and 60%, the mean square error reduction rate was 22.51%, 13.12%, 8.14% and 5.59% respectively.

D. Network Life

The lifetime of a wireless sensor network refers to the time that the network exists when the first node in the wireless sensor network runs out of energy and dies.

In the experiment, we took different values for both the number of sensors and the number of sensors for the big data task. In addition, we also take different values for the data volume of the big data task. The lifetime of different wireless sensor networks is obtained by using different values. Then, the influence of parameters on the life of wireless sensor network is obtained by changing the life of wireless sensor network. In the experiment, the sensor density of the big data task was set as 30%, 40%, 50% and 60%, the data volume of the big data task was set as 5MB-15MB, and the number of sensors for the wireless sensor network was set as 50, 40, 30 and 20, respectively. In order to reduce the influence of outliers on the experimental results, we conducted 30 simulation experiments for each different value and finally took the average value. See Fig. 12 for the results.

As can be seen from the comparison Fig. 12abcd, the wireless sensor network life of the traditional algorithm will gradually decrease with the increase of sensor density of the big data task, but UAV-CCIO will gradually increase. This is because, when the sensor density of the big data task increases, the probability of receiving a large amount of data by each cluster-head node is higher. Therefore, the cluster-head node will run out of energy earlier because of sending more data, which will lead to premature death of the wireless sensor network. For UAV-CCIO, due to the increase in sensor density of the big data task, the number of TGNs increases accordingly, while the number of nodes around each TGN decreases, which reduces the amount of data sent by each TGN, thus avoiding premature death of the wireless sensor network.

For both LEACH (TA) and UAV-CCIO, the increase in the data volume of the big data task will lead to the increase in the data volume sent by the node, which will increase the energy consumption of the data sent by the sensor, thus reducing the life of the wireless sensor network. It can be found that the wireless sensor network life is not sensitive to the number of sensors in the wireless sensor network. No matter how the number of sensors in the wireless sensor network changes, the life of the wireless sensor network corresponding to the two algorithms changes very little. This is because after load balancing, the data volume of each cluster head or TGN is similar, so the energy consumption is similar, and the wireless sensor network life is similar. The results show that when the IoT device density of the big data task is 30%, 40%, 50% and 60% respectively, the average life of the wireless sensor network is increased by 2.04 times, 2.69 times, 3.78 times and 4.32 times compared with the traditional algorithm.

UAV-CCIO determines TGNs by K-means clustering algorithm according to the data volume of the task. In special cases, it may exist that: 1. The data volume of tasks in the network has a large outlier. 2. The data volume of tasks in the network has a small outlier. All of them will influence the choice of TGNs by UAV-CCIO as shown in Fig. 13

1. When there are large outliers, some nodes with large data volume will be divided into ordinary nodes. In extreme cases, only the node with the largest task data volume will be divided into ordinary node, thus reducing the number of TGNs and increasing the energy consumption of the system, thus reducing the algorithm performance. However, it can be seen from the experiment that even if the extreme situation occurs, compared with the traditional algorithm, it can still improve the life of the system.

2. When there are small outliers, some nodes with small data volume will be divided into TGNs. In extreme cases, only the smallest mission data volume node will be divided into ordinary node, while the rest nodes are all TGNs, which will increase the flight energy consumption of UAV.

In order to reflect the performance of UAV-CCIO in these extreme cases, we specially conducted a simulation experiment, and the results are shown in Fig. 14.

As we can see from Fig. 14, when large or small outliers appear in the network, UAV-CCIO has better performance than LEACH in increasing network lifetime and reducing energy consumption. Even in extreme cases, although the system's

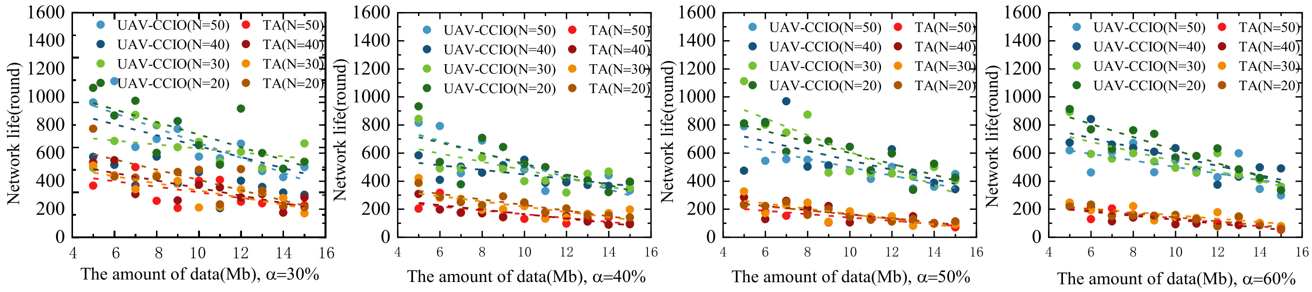


Fig. 12. Network life under different algorithms.

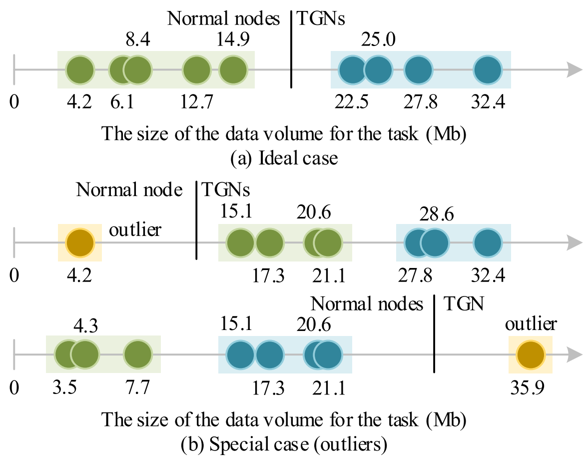


Fig. 13. Outliers affect the selection of TGNs by UAV-CCIO.

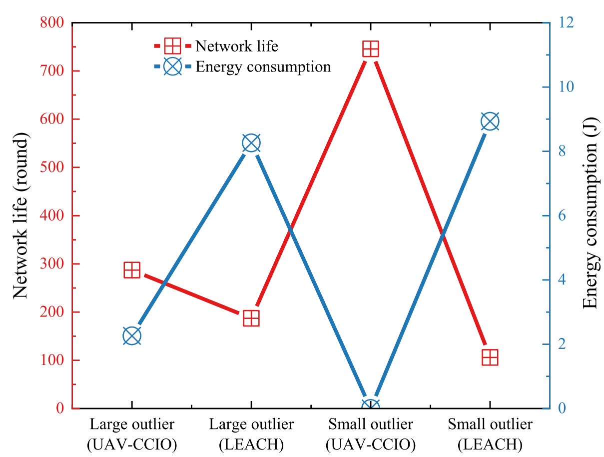


Fig. 14. Impact of outliers on the performance of UAV-CCIO and LEACH.

TGNs are affected, the performance is still better than that of the traditional algorithm.

However, compared with the ideal cases, the performance of UAV-CCIO is still reduced. When there is a large outlier, the network life of UAV-CCIO is only increased by about 50%. When there is a small outlier, although the network life of UAV-CCIO is very high. But at the same time, it can be seen that the UAV-CCIO basically reduces data transfer between nodes, so that the UAV has to fly to every node, thus increasing the energy consumption of the UAV. Therefore, in special cases, UAV-CCIO can perform better than LEACH algorithm, but there are still some shortcomings that need to be improved in the follow-up research.

VI. Conclusion and Future Work

ITS with 6G networks provide high-speed communication for connected vehicles, enabling them to interact extensively with IoT devices at the edge of the network. CCES is one of an effective approach to provide ubiquitous computation-communications platform for widely deployed IoT devices. In this paper, we propose UAVs act as CCES to construct ubiquitous task offloading platform for connected vehicles which form the IoV system and interact with the IoT devices in ITS. In the proposed task offloading platform, for IoT devices, Nodes with a large amount of data are selected as TGNs, and the TGNs collect all the task of left nodes. Then, UAVs fly to these TGNs to provide task offloading platform for TGNs to offload task. IoT task routing and UAV flight path are designed to optimize IoT devices and UAV energy consumption. The extensive experimental simulations indicate that the performance of the proposed solution is better than the existing scheme. In our proposed solution, the number of TGNs optimized has not been further optimized. In fact, the number and location of TGNs selected can have a significant impact on system performance. First, the number and location of TGNs selected can cause the IoT devices' task to route at different distances, affecting its energy consumption. Second, it also affects the UAV's flight path and distance, and thus its energy consumption. However, in some special cases, the data volume of a task is too large or too small, and these outliers affect UAV-CCIO to select TGNs, which will affect system performance. Therefore, in future work, we will study a more robust TGNs selection algorithm and a more efficient UAV path algorithms, which selects the optimal number and the location of TGNs to improve the performance of the system.

References

[1] X. Wang, L. T. Yang, Y. Wang, X. Liu, Q. Zhang, and M. J. Deen, "A distributed tensor-train decomposition method for cyber-physical-social services," *ACM Trans. Cyber-Phys. Syst.*, vol. 3, no. 4, pp. 1–15, Oct. 2019.

[2] A. Villalonga, G. Beruvides, F. Castano, and R. E. Haber, "Cloud-based industrial cyber–physical system for data-driven reasoning: A review and use case on an Industry 4.0 pilot line," *IEEE Trans. Ind. Informat.*, vol. 16, no. 9, pp. 5975–5984, Sep. 2020.

[3] X. Wang *et al.*, "A multi-order distributed HOSVD with its incremental computing for big services in cyber-physical-social systems," *IEEE Trans. Big Data*, vol. 6, no. 4, pp. 666–678, Dec. 2020.

[4] H. Guo and J. Liu, "UAV-enhanced intelligent offloading for Internet of Things at the edge," *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2737–2746, Apr. 2020.

[5] L. Ren, Y. Liu, X. Wang, J. Lu, and M. J. Deen, "Cloud–edge-based lightweight temporal convolutional networks for remaining useful life prediction in IIoT," *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12578–12587, Aug. 2021.

[6] C. Zhuo, S. Luo, H. Gan, J. Hu, and Z. Shi, "Noise-aware DVFS for efficient transitions on battery-powered IoT devices," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 7, pp. 1498–1510, Jul. 2020.

[7] S. Shekhar *et al.*, "URMILA: Dynamically trading-off fog and edge resources for performance and mobility-aware IoT services," *J. Syst. Archit.*, vol. 107, Aug. 2020, Art. no. 101710.

[8] Y. Liu, Z. Zeng, X. Liu, X. Zhu, and M. Z. A. Bhuiyan, "A novel load balancing and low response delay framework for edge-cloud network based on SDN," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5922–5933, Jul. 2020.

[9] L. Calderoni, A. Magnani, and D. Maio, "IoT manager: An open-source IoT framework for smart cities," *J. Syst. Archit.*, vol. 98, pp. 413–423, Sep. 2019.

[10] L. Ren, Y. Laili, X. Li, and X. Wang, "Coding-based large-scale task assignment for industrial edge intelligence," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2286–2297, Oct. 2020.

[11] B. Jiang, G. Huang, T. Wang, J. Gui, and X. Zhu, "Trust based energy efficient data collection with unmanned aerial vehicle in edge network," *Trans. Emerg. Telecommun. Technol.*, Mar. 2020, Art. no. e3942.

[12] J. Tan, W. Liu, T. Wang, M. Zhao, A. Liu, and S. Zhang, "A high-accurate content popularity prediction computational modelling for mobile edge computing by using matrix completion technology," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 6, 2019, doi: 10.1002/ett.3871.

[13] M. Chen, T. Wang, S. Zhang, and A. Liu, "Deep reinforcement learning for computation offloading in mobile edge computing environment," *Comput. Commun.*, vol. 175, pp. 1–12, Jul. 2021.

[14] Y. Ouyang, W. Liu, Q. Yang, X. Mao, and F. Li, "Trust based task offloading scheme in UAV-enhanced edge computing network," *Peer Peer Netw. Appl.*, vol. 14, no. 5, pp. 3268–3290, Sep. 2021, doi: 10.1007/s12083-021-01137-y.

[15] J. Luo, X. Deng, H. Zhang, and H. Qi, "QoE-driven computation offloading for edge computing," *J. Syst. Archit.*, vol. 97, pp. 34–39, Aug. 2019.

[16] Q. Xu, Z. Su, and Q. Yang, "Blockchain-based trustworthy edge caching scheme for mobile cyber-physical system," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 1098–1110, Feb. 2019.

[17] X. Deng, J. Li, E. Liu, and H. Zhang, "Task allocation algorithm and optimization model on edge collaboration," *J. Syst. Archit.*, vol. 110, Nov. 2020, Art. no. 101778.

[18] T. Li, W. Liu, Z. Zeng, and N. N. Xiong, "DRLR: A deep reinforcement learning based recruitment scheme for massive data collections in 6G-based IoT networks," *IEEE Internet Things J.*, early access, Mar. 22, 2021, doi: 10.1109/JIOT.2021.3067904.

[19] C. Huang, G. Huang, W. Liu, R. Wang, and M. Xie, "A parallel joint optimized relay selection protocol for wake-up radio enabled WSNs," *Phys. Commun.*, vol. 47, Aug. 2021, Art. no. 101320.

[20] Y. Ouyang, Z. Zeng, X. Li, T. Wang, and X. Liu, "A verifiable trust evaluation mechanism for ultra-reliable applications in 5G and beyond networks," *Comput. Standards Interfaces*, vol. 77, Aug. 2021, Art. no. 103519.

[21] N. Zhang, N. Cheng, A. T. Gamage, K. Zhang, J. W. Mark, and X. Shen, "Cloud assisted HetNets toward 5G wireless networks," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 59–65, Jun. 2015.

[22] M. Yu, A. Liu, N. N. Xiong, and T. Wang, "An intelligent game based offloading scheme for maximizing benefits of IoT-edge-cloud ecosystems," *IEEE Internet Things J.*, early access, Nov. 23, 2020, doi: 10.1109/JIOT.2020.3039828.

[23] Y. Liu, T. Wang, S. Zhang, X. Liu, and X. Liu, "Artificial intelligence aware and security-enhanced traceback technique in mobile edge computing," *Comput. Commun.*, vol. 161, pp. 375–386, Sep. 2020.

[24] L. Hu, A. Liu, and M. Xie, "UAVs joint vehicles as data mules for fast codes dissemination for edge networking in smart city," *Peer Peer Netw. Appl.*, vol. 12, no. 6, pp. 1550–1574, Nov. 2019.

[25] S. Huang, Z. Zeng, K. Ota, M. Dong, T. Wang, and N. N. Xiong, "An intelligent collaboration trust interconnections system for mobile information control in ubiquitous 5G networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 1, pp. 347–365, Jan. 2021.

[26] X. Wang, L. T. Yang, L. Song, H. Wang, L. Ren, and M. J. Deen, "A tensor-based multiattributes visual feature recognition method for industrial intelligence," *IEEE Trans. Ind. Informat.*, vol. 17, no. 3, pp. 2231–2241, Mar. 2020.

[27] T. Li, W. Liu, T. Wang, Z. Ming, X. Li, and M. Ma, "Trust data collections via vehicles joint with unmanned aerial vehicles in the smart Internet of Things," *Trans. Emerg. Telecommun. Technol.*, Apr. 2020, Art. no. e3942.

[28] J. Wang *et al.*, "HyTasker: Hybrid task allocation in mobile crowd sensing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 3, pp. 598–611, Mar. 2020.

[29] J. Ge, B. Liu, T. Wang, Q. Yang, A. Liu, and A. Li, "Q-learning based flexible task scheduling in a global view for the Internet of Things," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 8, Sep. 2020, Art. no. e4111.

[30] M. Jia, W. Liang, Z. Xu, and M. Huang, "Cloudlet load balancing in wireless metropolitan area networks," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun. (IEEE INFOCOM)*, Apr. 2016, pp. 1–9.

[31] T. Li, A. Liu, N. N. Xiong, S. Zhang, and T. Wang, "A trustworthiness-based vehicular recruitment scheme for information collections in distributed networked systems," *Inf. Sci.*, vol. 545, pp. 65–81, Feb. 2021.

[32] T. Wang, H. Luo, X. Zeng, Z. Yu, A. Liu, and A. K. Sangaiah, "Mobility based trust evaluation for heterogeneous electric vehicles network in smart cities," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1797–1806, Mar. 2021.

[33] Q. Liu, P. Hou, G. Wang, T. Peng, and S. Zhang, "Intelligent route planning on large road networks with efficiency and privacy," *J. Parallel Distrib. Comput.*, vol. 133, pp. 93–106, Nov. 2019.

[34] Y. Qu and N. Xiong, "RFH: A resilient, fault-tolerant and high-efficient replication algorithm for distributed cloud storage," in *Proc. 41st Int. Conf. Parallel Process.*, Sep. 2012, pp. 520–529.

[35] B. Lin *et al.*, "A time-driven data placement strategy for a scientific workflow combining edge computing and cloud computing," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 4254–4265, Jul. 2019.

[36] W. Fang, X. Yao, X. Zhao, J. Yin, and N. Xiong, "A stochastic control approach to maximize profit on service provisioning for mobile cloudlet platforms," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 4, pp. 522–534, Apr. 2016.

[37] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994–6009, Sep. 2017.

[38] Y. Yu, J. Zhang, and K. B. Letaief, "Joint subcarrier and CPU time allocation for mobile edge computing," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2016, pp. 1–6.

[39] L. Chen, S. Zhou, and J. Xu, "Computation peer offloading for energy-constrained mobile edge computing in small-cell networks," *IEEE/ACM Trans. Netw.*, vol. 26, no. 4, pp. 1619–1632, Aug. 2017.

[40] N. Zhang, H. Liang, N. Cheng, Y. Tang, J. W. Mark, and X. Shen, "Dynamic spectrum access in multi-channel cognitive radio networks," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 11, pp. 2053–2064, Nov. 2014.

[41] J. Gui, X. Dai, and X. Deng, "Stabilizing transmission capacity in millimeter wave links by Q-learning-based scheme," *Mobile Inf. Syst.*, vol. 2020, pp. 1–17, Feb. 2020.

[42] Q. Liu, Y. Tian, J. Wu, T. Peng, and G. Wang, "Enabling verifiable and dynamic ranked search over outsourced data," *IEEE Trans. Services Comput.*, early access, Jun. 11, 2019, doi: 10.1109/TSC.2019.2922177.

[43] G. Li, F. Li, T. Wang, J. Gui, and S. Zhang, "Bi-adjusting duty cycle for green communications in wireless sensor networks," *EURASIP J. Wireless Commun. Netw.*, vol. 2020, no. 1, pp. 1–55, Aug. 2020.

[44] X. Liu, H. Song, and A. Liu, "Intelligent UAVs trajectory optimization from space-time for data collection in social networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 853–864, Apr. 2021.

[45] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint load balancing and offloading in vehicular edge computing and networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4377–4387, Jun. 2018.

[46] K. Zhang, Y. Mao, S. Leng, A. Vinel, and Y. Zhang, "Delay constrained offloading for mobile edge computing in cloud-enabled vehicular networks," in *Proc. 8th Int. Workshop Resilient Netw. Design Modeling (RNDM)*, Sep. 2016, pp. 288–294.

[47] Y. Zhao, T. Wang, S. Zhang, and Y. Wang, "Towards minimum code dissemination delay through UAV joint vehicles for smart city," *IET Commun.*, vol. 14, no. 15, pp. 2442–2452, Sep. 2020.

[48] S. Huang, J. Gui, T. Wang, and X. Li, "Mobile vehicles joint UAVs for secure data collection in smart city," *Ann. Telecommun.*, Aug. 2020.

[49] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal delay constrained offloading for vehicular edge computing networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[50] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun. (IEEE INFOCOM)*, Apr. 2016, pp. 1–9.

[51] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2015.

[52] K. Xie *et al.*, "Recover corrupted data in sensor networks: A matrix completion solution," *IEEE Trans. Mobile Comput.*, vol. 16, no. 5, pp. 1434–1448, May 2017.

[53] W. Huang, K. Ota, M. Dong, T. Wang, S. Zhang, and J. Zhang, "Result return aware offloading scheme in vehicular edge networks for IoT," *Comput. Commun.*, vol. 164, pp. 201–214, Dec. 2020.

[54] M. Bonola, L. Bracciale, P. Loreti, P. Amici, A. Rabuffi, and G. Bianchi, "Opportunistic communication in smart city: Experimental insight with small-scale taxi fleets as data carriers," *Ad Hoc Netw.*, vol. 43, pp. 43–55, Jun. 2016.

[55] X. Liu, P. Lin, T. Liu, T. Wang, A. Liu, and W. Xu, "Objective-variable tour planning for mobile data collection in partitioned sensor networks," *IEEE Trans. Mobile Comput.*, early access, Jun. 17, 2020, doi: 10.1109/TMC.2020.3003004.

[56] D. Ebrahimi, S. Sharafeddine, P.-H. Ho, and C. Assi, "UAV-aided projection-based compressive data gathering in wireless sensor networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1893–1905, Apr. 2018.

[57] Y. Ren, T. Wang, S. Zhang, and J. Zhang, "An intelligent big data collection technology based on micro mobile data centers for crowdsensing vehicular sensor network," *Pers. Ubiquitous Comput.*, pp. 1–17, Aug. 2020.

[58] J. Sun, X. Wang, N. Xiong, and J. Shao, "Learning sparse representation with variational auto-encoder for anomaly detection," *IEEE Access*, vol. 6, pp. 33353–33361, 2018.

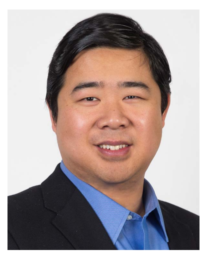
[59] B. Yi *et al.*, "Deep matrix factorization with implicit feedback embedding for recommendation system," *IEEE Trans. Ind. Informat.*, vol. 15, no. 8, pp. 4591–4601, Aug. 2019.

[60] H. Li, J. Liu, R. Liu, N. Xiong, K. Wu, and T.-H. Kim, "A dimensionality reduction-based multi-step clustering method for robust vessel trajectory analysis," *Sensors*, vol. 17, no. 8, p. 1792, Aug. 2017.

Run Liu is currently a Student with the School of Computer Science and Engineering, Central South University, China. His research interest includes wireless sensor networks.

Anfeng Liu received the M.Sc. and Ph.D. degrees in computer science from Central South University, China, in 2002 and 2005, respectively. He is currently a Professor with the School of Information Science and Engineering, Central South University. His major research interests include crowd sensing networks and wireless sensor networks.

Zhenzhe Qu received the master's degree with the School of Software, Central South University, China, in 2019, where he is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering. His research interest includes edge computing.



Neal N. Xiong (Senior Member, IEEE) received the Ph.D. degree in sensor system engineering from Wuhan University in 2007 and the Ph.D. degree in dependable communication networks from the Japan Advanced Institute of Science and Technology, in 2008.

Before he attended Northeastern State University, he worked at Georgia State University, Wentworth Technology Institution, and Colorado Technical University (a Full Professor for about five years) for about ten years. He is currently an Associate Professor (fifth year) with the Department of Computer Science and Mathematics, Sul Ross State University, Alpine, TX, USA. His research interests include cloud computing, security and dependability, parallel and distributed computing, networks, and optimization theory.