Chinese Journal of Electronics
Vol.32, No.6, Nov. 2023

Mobility-Aware Multi-Task Migration and Offloading Scheme for Internet of Vehicles

LI Xujie[1,2,3], TANG Jing[1], XU Yuan[1], and SUN Ying[1,4]

(1. *College of Computer and Information, Hohai University, Nanjing 210098, China*)
(2. *State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China*)
(3. *Key Lab for Wireless Sensor Network and Communication, Shanghai Institute of Microsystem, Shanghai 200050, China*)
(4. *School of Information Engineering, Jiangsu Open University, Nanjing 210017, China*)

Abstract — In Internet of vehicles, vehicular edge computing (VEC) as a new paradigm can effectively accomplish various tasks. Due to limited computing resources of the roadside units (RSUs), computing ability of vehicles can be a powerful supplement to computing resources. Then the task to be processed in data center can be offloaded to the vehicles by the RSUs. Due to mobility of the vehicles, the tasks will be migrated among the RSUs. How to effectively offload multiple tasks to the vehicles for processing is a challenging problem. A mobility-aware multi-task migration and offloading scheme for Internet of vehicles is presented and analyzed. Considering the coupling between migration and offloading, the joint migration and offloading optimization problem is formulated. The problem is a NP-hard problem and it is very hard to be solved by the conventional methods. To tackle the difficult problem, the idea of alternating optimization and divide and conquer is introduced. The problem can be decoupled into two sub-problems: computing resource allocation problem and vehicle node selection problem. If the vehicle node selection is given, the problem can be solved based on Lagrange function. And if the allocation of computing resource is given, the problem turns into a 0-1 integer programming problem, and the linear relaxation of branch bound algorithm is introduced to solve it. Then the optimization value is obtained through continuous iteration. Simulation results show that the proposed algorithm can effectively improve system performance.

Key words — Vehicular edge computing, Task offloading, Task migration.

I. Introduction

With the emergence of mobile and on-board applications with high computing and low latency requirements, computing offloading based on vehicular edge computing (VEC) architecture has attracted increasing attention [1]. Due to the high mobility of vehicles and limited bandwidth of vehicular networks, the tasks will be migrated among roadside units (RSUs). This aggravates the difficulty of task offloading and becomes a thorny issue.

Recently, there are many researches focusing on the field in the task offloading for Internet of vehicles. Some scholars focus on the architecture of the Internet of vehicles, such as software defined framework, cooperation framework, mobility framework, cloud architecture and so on. A motion-aware task offloading scheme called soft-van is proposed in [2], which aims to minimize the task calculation delay in software-defined vehicular networks. The proposed scheme consists of two stages: fog node selection and task offloading. In the scheme of my paper, these two stages are also included. For the selection of fog nodes, my paper adopts a better 0-1 selection strategy. An effective virtual edge scheme is proposed in [3], which utilizes free computing resources of multiple vehicles as virtual servers to realize collaborative vehicular edge computing. The authors design a virtual edge formation algorithm that takes into account both the stability of virtual edges

Manuscript Received Oct. 8, 2022; Accepted Feb. 14, 2023. This work was supported by the Open Foundation of State key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications) (SKLNST-2022-1-15), the Future Network Scientific Research Fund Project (FNSRFP-2021-YB-7), the Provincial Water Science and Technology Program of Jiangsu (2020028), the Social and People's Livelihood Technology in Nantong City (MS22021042), the Fundamental Research Funds for the Central Universities (B200205007), the Provincial Key Research and Development Program of Jiangsu (BE2019017), and the Open Research Fund Key Laboratory of Wireless Sensor Network and Communication, Chinese Academy of Sciences (20190914).

and the available computing resources of vehicles constituting virtual edges. However, this paper does not consider the vehicle unwilling to contribute its free computing resources, so a reward mechanism is set up in my paper to avoid the occurrence of this situation. Zhang *et al.* [4] propose a cloud-based mobile edge computing offloading framework for vehicular networks. Considering the time consumption of computing tasks and the mobility of vehicles, the authors propose an efficient predictive combination mode grading scheme, which adaptively offloads tasks to mobile edge computing (MEC) servers by direct uploading or predictive relay transmission. A cloud access coverage protocol architecture is proposed in [5] to improve the performance of cloud access in distributed data center cloud architecture. The authors explore how virtual machine mobility and routing can be linked to user mobility to compensate for performance degradation due to increased user-cloud network distance.

The performance is crucial when the Internet of vehicles is deployed and operated. Some important performance parameters are optimized, such as latency, robustness, energy and so on. A parallel task offloading model and a small area-based edge offloading scheme for MEC is presented in [6], and the problem is formulated as an optimization problem to minimize the completion time of all tasks. A new dynamic edge computing model is introduced and robust task offloading is studied in [7], which solves the challenges caused by mobility and power constraints. Zhou *et al.* [8] review the state of the art for task offloading in vehicular fog networks and argue that mobility is not only a barrier to timely computation in vehicular fog networks, but can also improve latency performance. Then the authors identify machine learning and coded computing as key enabling techniques to address and leverage mobility. Yadav *et al.* [9] provide an energy-efficient dynamic computational offloading and resource allocation scheme to reduce energy consumption and service latency. The energy-efficient dynamic computation offloading and resources allocation scheme problem as a joint energy and delay cost minimization problem satisfying vehicular node mobility and end-to-end delay duration constraints is formulated, and a three-stage energy-efficient dynamic computation offloading and resources allocation scheme solution is proposed.

Due to the mobility of the vehicles, task offloading and resource allocation will face enormous difficulties. How to effectively offload tasks to processors based on mobility-aware attract many researchers. A mobility-aware computation offloading design is investigated in [10], which considers random mobility of the vehicle and its potential handover during the offloading process. The computation offloading task in VEC is modeled in [11] to evaluate the ability to simulate the elements that contribute to enhancing its performance for VEC. Hu *et al.* [12] study computational offloading and resource allocation in IoT networks that support both mobility and energy harvesting. The long-term average total service cost of all mobile IoT devices is minimized by optimizing the collected energy, task allocation factor, central processing unit frequency, transmission power, and association vector. A mobile device selection algorithm is proposed in [13], which takes into account the social relationship, location correlation and mobile activity of the mobile device. Then based on the improved Kuhn-Munkres algorithm, a joint social aware and mobile-aware computational offloading algorithm is proposed to obtain a resource allocation strategy that minimizes energy consumption while satisfying the minimum delay condition. Li *et al.* [14] develop a novel computational offloading scheme that utilizes nonorthogonal multiple access and dual connectivity and focuses on jointly optimizing task segmentation and power allocation to minimize total energy consumption. An efficient task offloading scheme in vehicular edge computing network is studied in [15]. The vehicle selection scheme considers the offloading time, communication and computing resources allocation, the mobility of the vehicle, and the delay of the task. An optimization model is proposed in [16] to solve the resource allocation problem of mobile users so as to maximize the total number of completed tasks considering the mobile patterns of users in the network. The authors develop a synthetic correlation mobility model to reproduce human movements during the morning rush hour. On this basis, a practical resource allocation algorithm is proposed to solve the problem of mobile equipment deployment and resource allocation. A mobile access prediction algorithm based on tail matching subsequences is proposed in [17], and the effectiveness and accuracy of the algorithm are verified by experiments on real mobile datasets.

Meanwhile, mobility also brings the task or data migration problems. Some work is beginning to consider these questions. Liang *et al.* [18] consider jointly managing computing and radio resources to optimize migration/handoff strategies between BSs. The goal is to maximize offloading rates, quantify MEC throughput, and minimize migration costs. Then an efficient method based on relaxed rounding is proposed to solve complex combinatorial problems. A general three-layer fog computing network architecture is considered in [19]. The mobility of the fog computing network is characterized by its residence time in each coverage area, which follows an exponential distribution. In order to maximize the terminal revenue, the offloading decision and computing resource allocation are jointly optimized to reduce the migration probability. A virtual machine joint migration model based on ant colony optimization is proposed in [20] for heterogeneous intelligent medical system based on mobile cloud computing in smart city environment. In this model, user migration and provisioning of virtual machine resources in the cloud solve the problem of virtual machine migration.

As mentioned above, most existing studies discuss offloading computing tasks from vehicles to roadside units, cloud servers or vehicles. However, there are still some challenges in performing efficient computing tasks offloading due to the high mobility of vehicles and limited bandwidth of vehicular networks. In the real system, considering the mobility of the vehicle, the task may not be able to complete the task offloading to the computing nodes in the initial connected RSU, so the migration cost of the task needs to be considered. In this paper, we first evaluate the migration process and cost by comparing the offloading delay of the task and the residence time of the vehicle within the coverage of the current RSU. Considering the joint optimization of delay and cost, an effective offloading scheme is designed to minimize the delay and cost. Then, the problem is modeled as a mixed integer nonlinear programming problem. In order to solve the optimization problem, we adopt an alternating optimization method to decompose the original problem into two sub-problems: the resource allocation sub-problem and the node selection sub-problem, and Lagrange function is introduced to solve the resource allocation sub-problem. And an improved branch and bound algorithm based on linear relaxation is proposed to solve the node selection sub-problem.

The main contributions of this paper are as follows:

- The problem is formulated as a mixed integer nonlinear programming problem. And the problem can be decoupled into two sub-problems. An improved branch and bound algorithm based on linear relaxation and the Harris hawk algorithm are presented to solve the two sub-problems, respectively.
- A mobility-aware multi-task migration and offloading scheme for Internet of vehicles is proposed. Considering to mobility of the vehicles, we should consider the task migration when the task is offloading. To tackle the NP-hard problem, the method of alternating optimization and divide and conquer is introduced to minimize the total cost of task offloading and migration.
- Simulation platform is established and the simulation results show that the proposed algorithm can effectively improve system performance.

The remainder of this paper is organized as follows. In Section II, the system model including task offloading model, computing model, migration model and cost model is introduced and described. In Section III, a mobility-aware multi-task migration and offloading scheme is presented. The simulation results are provided and discussed in Section IV and the paper is concluded in Section V.

II. System Model

In our scenario, we consider a straight road, where M RSUs are deployed along one side of the road, as shown in Fig.1. Without loss of generality, we assume that the road is parallel to the horizontal axis. The serial number of the RSUs can be denoted as $m = \{1, 2, \dots, M\}$, and the distance between adjacent RSUs is defined as $2R$. These tasks will be generated by the pedestrians or vehicles on the road. These complex tasks will be transferred to the data center to be processed. Meanwhile, computing ability of vehicles can be a powerful supplement to computing resources. To process these tasks more quickly, the vehicles with avail-

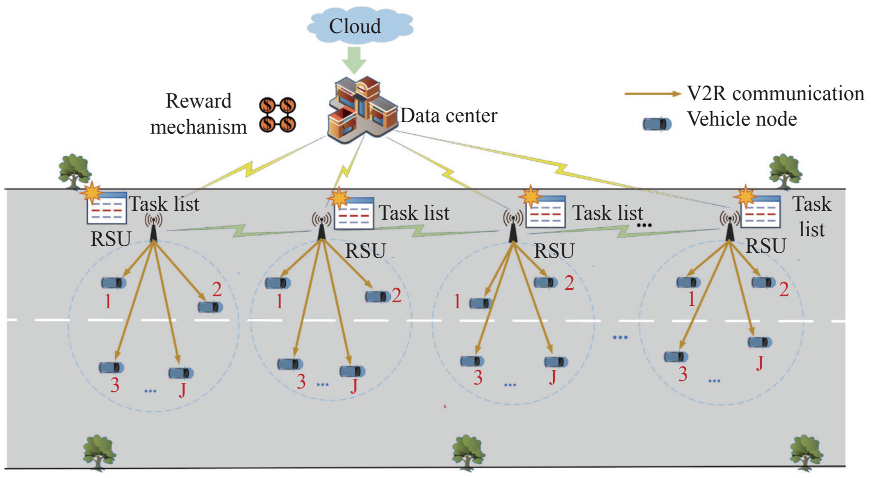

Fig. 1. System model.

able computing resources can cooperatively handle these tasks. We assume that K tasks will be generated in every time slot. Each task can be defined by three items $w_k = \{I_k, C_k, T_k^{\max}\},\ k \in \{1, 2, \dots, K\}$, where I_k is the data size of the task k (Mbits), and C_k represents the total number of CPU cycles required to complete the computing task k, and $T_k^{\max}$ is the delay constraint of the task k. The vehicles can provide the computing ability, and the maximum CPU computing ability of vehicle j can be denoted as $f_j^{\max}$. To describe how the positions of vehicles change over time, the timeline is divided into many time slots, and d represents the time length of each time slot.

In addition, considering the mobility of vehicles, tasks will be migrated between different RSUs. According to the offloading model, the residence time of the vehicle and the amount of data transferred in every RSU can be calculated. Once the task offloading cannot be completed within the coverage of the current RSU, the task will be migrated to next adjacent RSU through optical fiber among RSUs. Sometimes, the task offloading maybe completed across multiple RSUs, then the total number of migrations should be calculated.

1. Task offloading model

The speed of vehicle j can be written as $v_j, j \in \{1, 2, \dots, J\}$, and it follows uniform distribution. So the distance that vehicle j can travel in each time slot is $v_j d$. We denote the position of vehicle j in time slot t as $(p_j[t], q_j[t])$. Here, $(p_j[t], q_j[t])$ are the abscissa and ordinate of the position, respectively. Because the road is parallel to the horizontal axis, the horizontal and vertical coordinates of vehicle j in time slot t can be expressed as

$$\begin{aligned} p_j[t] &= p_j[0] + \sum_{i=1}^{t} v_j d[i] \\ q_j[t] &= q_j[0] \end{aligned} \tag{1}$$

where, $p_j[0]$ and $q_j[0]$ is the initial horizontal and vertical coordinates of vehicle j. We denote the horizontal and vertical coordinates of RSU m as p_m^{RSU} and q_m^{RSU}, respectively. Here, we set $p_1^{\text{RSU}} = R$ and $q_1^{\text{RSU}} = 0$. Then the distance between RSU m and vehicle j in time slot t can be expressed as

$$D_{m,j}[t] = \sqrt{(p_j[t] - p_m^{\text{RSU}})^2 + (q_j[t] - q_m^{\text{RSU}})^2} \tag{2}$$

Usually, we assume that the vehicles will establish communication link with the nearest RSU. Therefore, the communication distance between vehicle j and the RSUs in time slot t can be expressed as

$$D_j^{\text{off}}[t] = \min_{m \in \{1,2,\dots,M\}} \{D_{m,j}[t]\} \tag{3}$$

Therefore, the transmission rate between nearest RSU and vehicle j in time slot t is written as

$$R_j[t] = B \cdot \log_2 \left(1 + \frac{P_{\text{RSU}}/(D_j^{\text{off}}[t])^{\alpha}}{n_0 \cdot B}\right) \tag{4}$$

Here, B is the bandwidth of the communication channel between RSU and vehicle j, P_{RSU} is the transmission power of RSUs, α is the path loss coefficient, n_0 is the power spectral density of Gaussian white noise. Without loss of generality, we assume that vehicle j locates in the coverage of RSU 1 in the beginning. Then the dwell time of the vehicle j in the process of unloading the task is divided into three periods: the dwell time in the first RSU, the dwell time in the intermediate RSUs, and the dwell time in the last RSU. And the dwell time of vehicle j in the first RSU can be calculated as

$$t_{1,j}^{\text{stay}} = \frac{2R - p_j[0]}{v_j} \tag{5}$$

It can be obtained that the dwell time of vehicle j in the intermediate RSUs for the next time slots except for the last time slot is equal and can be written as

$$t_{2,j}^{\text{stay}} = t_{3,j}^{\text{stay}} = \dots = t_{M-1,j}^{\text{stay}} = \frac{2R}{v_j} \tag{6}$$

Next, the dwell time of vehicle j in the last RSU can be written as

$$t_{M,j}^{\text{stay}} = \frac{2R \cdot M - p_j^{\text{last}}}{v_j} \tag{7}$$

Here, p_j^{last} is horizontal axis of the last position where the task is completely offloaded to vehicle j.

Then, the whole dwell time of vehicle j during the process of uploading the task can be calculated as

$$t_j^{\text{stay}} = \sum_{m=1}^{M} t_{m,j}^{\text{stay}} \tag{8}$$

Once the dwell time of vehicle j is represented, the total transmitted data of the vehicle j during the process of uploading the task can be written as

$$D = \int_0^{t_j^{\text{stay}}} R_j^{\text{off}}[t] \mathrm{d}t, \quad m = 1, 2, \dots, M \tag{9}$$

where, t_j^{stay} represents the whole dwell time of vehicle j during the process of uploading the task, and $R_j^{\text{off}}[t]$ represents the transmission rate between the nearest RSU and vehicle j in time slot t.

2. Computing model

When the RSUs offload the multi-task to the vehicle nodes, the vehicle nodes will allocate CPU resources for different tasks. As mentioned above, $f_j^{\max}$ is the maximum computing ability of the CPU of vehicle j. We denote the computing ability of vehicle j allocated to task k as $f_{k,j}$. Then the computing time that vehicle j computes task k can be expressed as

$$t_{k,j}^{\text{com}} = \frac{C_k}{f_{k,j}} \tag{10}$$

where C_k is the computing amount of task k.

3. Migration model

Considering the mobility of vehicles, the task offloading may not be able to be completed within the coverage of the initial RSU, so the task migration should be noted. Meanwhile, the task migration will result in the additional cost associated with the task switching process and computing migration. We denote one time migration cost of task k as $H \cdot I_k$, where I_k is the data size of the task k and H is the migration cost per bit. Then we denote the number of the RSU migration if task k is offloaded to vehicle j as $s(k,j)$. Then we can calculate the overall migration cost if task k is offloaded to vehicle j as follows:

$$h_{k,j} = H \cdot I_k \cdot s(k,j) \tag{11}$$

4. Cost model

In addition, considering the actual circumstance, not every vehicle is willing to contribute their spare computing resources. So we also need to set up a reward mechanism to give certain rewards to the vehicles that provide computing resources for computing tasks. This also can be regarded as the computing cost of the tasks. Let γ represent the unit cost of computing task. Then we can denote the reward φ_k of completing the computing of task k as $\gamma \cdot C_k + \frac{1}{t_{k,j}^{\text{com}}}$, where $t_{k,j}^{\text{com}}$ is the computing delay of vehicle j for task k, and its reciprocal represents an extra reward.

5. Problem formulation

In the process of unloading task, the task will be assigned to one vehicle. Then we denote the indicator that task k is unloaded to vehicle j as $u_{k,j}$, and we have

$$u_{k,j} = \begin{cases} 1, & \text{if task } k \text{ is offloaded to vehicle } j \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

Meanwhile, we assume that one task is only offloaded to one vehicle node. Then based on the above equation, we have the constraints $\sum_{j=1}^{J} u_{k,j} = 1, k \in \{1, 2, \dots, K\}$.

Next, the delay of returning the resuls can be ignored because the size of the returned result is small in general. Then the total delay is the sum of the transmission delay and the computing delay and can be expressed as

$$t_{k,j} = u_{k,j} \cdot (t_{k,j}^{\text{off}} + t_{k,j}^{\text{com}}) \tag{13}$$

In our scenario, multiple tasks are unloaded to multiple vehicles. We denote the overall delay as the sum of the transmission time and the computing time of the last vehicle to return the result. Therefore, the overall delay of multiple tasks can be expressed as

$$t^{\text{sum}} = \max_{k \in K, j \in J} t_{k,j} = \max_{k \in K, j \in J} u_{k,j} \cdot (t_{k,j}^{\text{off}} + t_{k,j}^{\text{com}}) \tag{14}$$

For the optimal uploading scheme, we should consider how to simultaneously minimize the delay, reward, and migration cost. Therefore, the optimization problem can be formulated as

$$\begin{aligned} \text{P1} &= \underset{u_{k,j}, f_{k,j}}{\text{minimize}} \left\{ \eta \cdot t^{\text{sum}} + (1-\eta) \cdot \left(\sum_{k \in K} \sum_{j \in J} u_{k,j} \cdot \varphi_{k,j} + \sum_{k \in K} \sum_{j \in J} u_{k,j} \cdot h_{k,j} \right) \right\} \\ &= \underset{u_{k,j}, f_{k,j}}{\text{minimize}} \left\{ \eta \cdot \max_{k \in K, j \in J} u_{k,j} \cdot \left(t_{k,j}^{\text{off}} + \frac{C_k}{f_{k,j}} \right) + (1-\eta) \cdot \left(\sum_{k \in K} \sum_{j \in J} u_{k,j} \left(\gamma \cdot C_k + \frac{1}{t_{k,j}^{\text{com}}} \right) + \sum_{k \in K} \sum_{j \in J} u_{k,j} \cdot h_{k,j} \right) \right\} \\ \text{s.t.}\ & \text{C1: } f_{k,j} \geq 0, \quad k \in K, j \in J \\ & \text{C2: } \sum_{k \in K} f_{k,j} \leq f_j^{\max}, \quad k \in K, j \in J \\ & \text{C3: } \sum_{j \in J} (u_{k,j} t_{k,j}^{\text{off}} + u_{k,j} t_{k,j}^{\text{com}}) \leq T_k^{\max}, \quad k \in K, j \in J \\ & \text{C4: } u_{k,j} \in \{0,1\}, \quad k \in K, j \in J \\ & \text{C5: } \sum_{j \in J} u_{k,j} = 1, \quad k \in K, j \in J \end{aligned} \tag{15}$$

Here, η represents the balance parameter of the optimization problem, t^{sum} represents the overall delay of multiple tasks, $\varphi_{k,j}$ represents the reward given by vehicle j for completing task k, and $h_{k,j}$ represents the overall migration cost when task k is offloaded to vehicle j. In the constraints, C1 and C2 represent that the computing resource of each vehicle is non-negative and the allocated computing resource for the task cannot exceed its allowed maximum computing resource, respectively. Constraint C3 indicates that the delay of unloading task cannot exceed allowed maximum delay. C4 and C5 describe that one task can only be offloaded to one vehicle node.

III. Proposed Mobility-Aware Multi-Task Migration and Offloading Scheme

In this section, a mobility-aware multi-task migration and offloading scheme is proposed. As shown in the above optimal problem, the variable $u_{k,j}$ and the variable $f_{k,j}$ are coupled. Here, $u_{k,j}$ is a binary variable and $f_{k,j}$ is a continuous variable. Therefore, the problem P1 is a mixed integer nonlinear programming problem, which makes it difficult to be solved. But it is shown that the problem P1 can be decoupled into two sub-problems: vehicle node selection and computing resource allocation. Therefore, we present an alternative optimization techniques to solve the problem.

1. Task migration algorithm

Meanwhile, in the process of uploading the task to the vehicle, we need to determine whether the migration has occurred. If the migration has happened, we should calculate how many times the task has been migrated. Therefore, a task migration algorithm is proposed as shown in Algorithm 1.

Algorithm 1 Task migration algorithm

```
1: Input: $w_k = \{I_k, C_k, T_k^{\max}\}$ $(k \in \{1,2,\ldots,K\})$, $R$, $v_j$
   $(j \in \{1,2,\ldots,J\})$, $\text{Data}_j^{\text{tmp}}$ $(j \in \{1,2,\ldots,J\})$.
2: Output: $h(k,j)$.
3: for $k = 1$ to $K$ do
4:   for $j = 1$ to $J$ do
5:     for $t = 1$ to $T$ do
6:       $\text{Data}_j^{\text{tmp}} = \text{Data}_j^{\text{tmp}} + R_j[t] \cdot d$;
7:       If $\text{Data}_j^{\text{tmp}} > I_k$
8:         $s(k,j) = \arg\min_{m \in \{1,2,\ldots,M\}} \{D_{m,j}[t]\}$;
9:       end if
10:    end for
11:  end for
12: end for
```

2. Computing resource allocation

When the node selection strategy $u_{k,j}$ is given, P1 becomes a function of $f_{k,j}$. Then we can omit the irrelevant terms to $f_{k,j}$, and rewrite the original problem as follows:

$$\begin{aligned} \text{P2} = \min_{f_{k,j}} \eta \cdot & \max_{k\in K, j\in J} \left\{ \frac{C_k \cdot u_{k,j}}{f_{k,j}} + (1-\eta) \cdot \left(\sum_{j=1}^{J}\sum_{k=1}^{K} u_{k,j} \left(\gamma \cdot C_k + \frac{1}{t_{k,j}^{\text{com}}} \right) \right) \right\} \\ \text{s.t.} \quad & \text{C1}: f_{k,j} \geq 0, \quad \forall k \in K, \forall j \in J \\ & \text{C2}: \sum_{k\in K} f_{k,j} \leq f_j^{\max}, \quad \forall j \in J \\ & \text{C3}: \sum_{j\in J} \left(t_{k,j}^{\text{off}} + \frac{C_k}{f_{k,j}} \right) \cdot u_{k,j} \leq T_k^{\max}, \quad \forall k \in K \end{aligned} \tag{16}$$

Obviously, the set of feasible solutions of P2 is convex. Now, we can show the convexity of the objective function P2. Denoting the objective function as $\varphi(f)$, we can obtain the second derivative $\nabla^2\varphi(f)$ as follows:

$$\frac{\partial^2 \varphi(f)}{\partial f_{k,j}^2} = \frac{2C_k \cdot \eta \cdot u_{k,j}}{f_{k,j}^3} + (1-\eta) \cdot \frac{u_{k,j}}{C_k} \tag{17}$$

It is easy to validate $\nabla^2\varphi(f) > 0$, so the Hessian matrix is a positive semi-definite matrix. Therefore, the problem P2 is a convex optimization problem, and its Lagrange function can be expressed as

$$\begin{aligned} L(f_{k,j}, \theta_k, \varepsilon_j) = & \eta \max_{k\in K, j\in J} \frac{C_k \cdot u_{k,j}}{f_{k,j}} + (1-\eta) \cdot \sum_{j=1}^{J}\sum_{k=1}^{K} u_{k,j} \cdot (\gamma \cdot C_k + \frac{1}{t_{k,j}^{\text{com}}}) \\ & + \sum_{k\in K} \theta_k \left(\sum_{j\in J} \left(t_{k,j}^{\text{off}} + \frac{C_k}{f_{k,j}} \right) \cdot u_{k,j} - T_k^{\max} \right) + \sum_{j\in J} \varepsilon_j \left(\sum_{k\in K} f_{k,j} - f_j^{\max} \right) \end{aligned} \tag{18}$$

According to the KKT condition, we have

$$\nabla L(f_{k,j}, \theta_k, \varepsilon_j) = 0 \tag{19}$$

$$\sum_{j\in J} \left(t_{k,j}^{\text{off}} + \frac{C_k}{f_{k,j}} \right) \cdot u_{k,j} - T_k^{\max} = 0 \tag{20}$$

$$\sum_{k\in K} f_{k,j} - f_j^{\max} = 0 \tag{21}$$

Finally, we have

$$\frac{\partial L}{\partial f_{k,j}} = \eta\cdot\frac{-C_k}{f_{k,j}^2} + \theta_k\cdot\frac{-C_k}{f_{k,j}^2} + \varepsilon_j = 0$$

$$\Rightarrow f_{k,j}{}^* = \sqrt{\frac{C_k\cdot(\eta+\theta_k)\cdot u_{k,j}}{\varepsilon_j}} \tag{22}$$

Substituting (22) into (20) and (21), we can get

$$\sum_{j\in J}\left(t_{k,j}^{\text{off}} + \frac{C_k}{\sqrt{\frac{C_k\cdot(\eta+\theta_k)\cdot u_{k,j}}{\varepsilon_j}}}\right) = T_k^{\max} \tag{23}$$

$$\sum_{k\in K}\sqrt{\frac{C_k\cdot(\eta+\theta_k)\cdot u_{k,j}}{\varepsilon_j}} = f_j^{\max} \tag{24}$$

Based on the above analysis, the computing resource allocation algorithm is presented as shown in Algorithm 2 if the vehicle node selection strategy is given.

Algorithm 2 Computing resource allocation algorithm

```
1: Select $\theta_k = \theta_k^0$, $k \in \{1,2,\dots,K\}$, $\varepsilon_j = \varepsilon_j^0$, $j \in \{1,2,\dots,J\}$ as the Lagrange initial value, set step = 0.01;
2: for $m = 1 : L$ ($L$ is the number of iteration)
3:   Update $\theta_k^m$ and $\varepsilon_j^m$ according to the equations below:
     $\theta_k^m = \theta_k^{m-1} + \text{step}\cdot\text{grad}(\theta_k^{m-1})$
     $\varepsilon_j^m = \varepsilon_j^{m-1} + \text{step}\cdot\text{grad}(\varepsilon_j^{m-1})$
4:   Update step = step/sqrt($m$);
5:   Substitute $\theta_k^m$ and $\varepsilon_j^m$ into equations (23) and (24), if
     $\text{abs}\left(\sum_{j\in J}\left(t_{k,j}^{\text{off}} + \frac{C_k}{\sqrt{\frac{C_k\cdot(\eta+\theta_k^m)\cdot u_{k,j}}{\varepsilon_j^m}}}\right) - T_k^{\max}\right) < \xi$
     and $\text{abs}\left(\sum_{k\in K}\sqrt{\frac{C_k\cdot(\eta+\theta_k^m)\cdot u_{k,j}}{\varepsilon_j^m}} - f_j^{\max}\right) < \xi$, then
     output $\theta_k^m$ and $\varepsilon_j^m$, go to step 7; Otherwise, let $m = m+1$ and go to step 2;
6: end for
7: Substitute $\theta_k^m$ and $\varepsilon_j^m$ into equaiton (22) to calculate $f_{k,j}{}^*$, then output $f_{k,j}{}^*$.
```

3. Vehicle node selection problem

Based on the above discussion, we can obtain the optimal computing resource allocation $f_{k,j}{}^*$. So if $f_{k,j}{}^*$ is given, we can rewrite the original problem P1 as

$$\begin{aligned} \text{P3} = \min_{u_{k,j}} &\left\{ \eta \max_{k\in K, j\in J}\left(u_{k,j}t_{k,j}^{\text{off}} + u_{k,j}\frac{C_k}{f_{k,j}{}^*}\right) + (1-\eta)\left(\sum_{j=1}^{J}\sum_{k=1}^{K} u_{k,j}\cdot\left(\gamma\cdot C_k + \frac{f_{k,j}{}^*}{C_k}\right) + \sum_{j=1}^{J}\sum_{k=1}^{K} u_{k,j}\cdot h(k,j)\right)\right\} \\ \text{s.t. } & \text{C4}: u_{k,j}\in\{0,1\},\quad \forall k\in K, \forall j\in J \\ & \text{C5}: \sum_{j\in J} u_{k,j} = 1,\quad \forall k\in K \end{aligned} \tag{25}$$

The problem P3 is a 0-1 integer programming problem for $u_{k,j}$. Now, we introduce a branch and bound algorithm to solve this problem[21]. Considering that problem P3 is a 0-1 integer programming problem, it can be relaxed if we let $0\le u_{k,j}\le 1$, so problem P3 can be rewritten as

$$\begin{aligned} \text{P4} = \min_{u_{k,j}} &\left\{ \eta \max_{k\in K, j\in J}\left(u_{k,j}t_{k,j}^{\text{off}} + u_{k,j}\frac{C_k}{f_{k,j}{}^*}\right) + (1-\eta)\left(\sum_{j=1}^{J}\sum_{k=1}^{K} u_{k,j}\cdot\left(\gamma\cdot C_k + \frac{f_{k,j}{}^*}{C_k}\right) + \sum_{j=1}^{J}\sum_{k=1}^{K} u_{k,j}\cdot h(k,j)\right)\right\} \\ \text{s.t. } & \text{C6}: u_{k,j}\in[0,1],\quad \forall k\in K, \forall j\in J \end{aligned} \tag{26}$$

The optimal solution of P4 is a lower bound of P3. It can be easily obtained by using convex optimization algorithm. Then, an integer solution for $u_{k,j}$ can be obtained by

$$u_{k,j} = \begin{cases} 1, & u_{k,j} > 0.5 \\ 0, & u_{k,j}\le 0.5 \end{cases},\quad \forall k\in K, j\in J \tag{27}$$

Once the value of $u_{k,j}$ is determined, P3 can be rewritten as

$$\text{P5} = \max_{k\in K, j\in J, u_{k,j}=1} \eta\cdot\left(t_{k,j}^{\text{off}} + \frac{C_k}{f_{k,j}{}^*}\right) + (1-\eta)\cdot\left(\sum_{j=1}^{J}\sum_{k=1}^{K}\gamma\cdot C_k + \frac{f_{k,j}{}^*}{C_k} + \sum_{j=1}^{J}\sum_{k=1}^{K} h(k,j)\right) \tag{28}$$

The solution of P5 is an upper bound of P3, and the optimal solution of P5 can be easily found using convex optimization algorithms. By solving problems P4 and P5, the lower and upper bounds of P3 can be obtained. Therefore, the optimal solution of P3 can be obtained by using the branch and bound algorithm, as shown in Algorithm 3.

Algorithm 3 Branch and bound algorithm

```
1: Initialize the optimal solution of P3 as an empty set, that is, $P3^* = \emptyset$ and the upper bound $UB = \{+\propto\}$;
2: Linear relaxation: let $u_{k,j} \in \{0,1\}$ relax into $u_{k,j} \in [0,1]$, $\forall k \in K$, $\forall j \in J$, i.e., from P3 to P4;
3: Solve P4 and P5 to obtain its optimal solution $P4^*$ and $P5^*$, and set them as the lower bound $LB^*$ and the upper bound $UB^*$ of P3, respectively;
4: Iteration:
5: There are $n$ branches in branch queue $Q$ to be solved;
6: for $m = 1 : n$
7:    Obtain the lower bound $LB_m$ of branch $m$ based on the form of P4;
8:    Obtain the upper bound $UB_m$ of branch $m$ based on the form of P5;
9:    If $LB_m < UB^*$, divide branch problem $m$ into 2 branches $m_1$ and $m_2$ and insert $m_1$ and $m_2$ into branch queue $Q$;
10:   Update $LB^*=\max\{LB^*, LB_m\}$;
11:   Update $UB^*=\min\{UB^*, UB_m\}$;
12:   If $LB^* \geq UB^*$, update the optimal solution $P3^*$ and corresponding feasible solution $u^*$;
13:   $m=m+1$;
14: end for
15: If $Q = \emptyset$, stop iteration and output current optimal solution $P3^*$ and feasible solution $u^*$. Otherwise, implement next iteration.
```

4. Mobility-aware multi-task migration and offloading algorithm

Based on the above analysis, a mobility-aware multitask migration and offloading scheme for Internet of vehicles (MAMMOS) is proposed to minimize the total cost of task offloading and migration. As mentioned above, the method of alternating optimization and divide and conquer is introduced. Then the initial problem can be decoupled into two sub-problems: computing resource allocation problem and vehicle node selection problem. Given the solution of vehicle node selection problem, the solution of computing resource allocation problem will be obtained according to Algorithm 2. Similarly, given the updated solution of computing resource allocation problem, the solution of vehicle node selection problem will be calculated according to Algorithm 3. By constantly iterating the Algorithm 2 and Algorithm 3 until the solution converges to a certain range. The detailed MAMMOS algorithm is shown as follows.

Algorithm 4 MAMMOS algorithm

```
1: Initialization parameters (the number of iteration $L$);
2: for $m = 1 : L$
3:    Solve problem P2 according to Algorithm 2, and obtain the optimal computing resource allocation $f_{k,j}^m$;
4:    Solve problem P3 based on Algorithm 3, and obtain optimal vehicle node selection $u_{k,j}^m$;
5:    If $\mathrm{abs}(f_{k,j}^m - f_{k,j}^{m-1}) < \sigma$ and $\mathrm{abs}(u_{k,j}^m - u_{k,j}^{m-1}) < \sigma$, output $f_{k,j}^{m*}$ and $u_{k,j}^{m*}$, and go to step 6; otherwise, let $m = m + 1$ and go to step 2;
6 end for
```

IV. Simulation and Discussion

1. Simulation scenarios and settings

In this section, some simulation results are presented to evaluate the effectiveness of the algorithm. In our simulation, Monte Carlo method are used to implement the simulation in Matlab software, which uses repeated random sampling to generate tasks and attributes such as the speed and direction of the vehicle, then simulate the process of the multiple tasks offloading and computing according to the proposed algorithm. Meanwhile, by changing the size of the task, the speed of vehicle, the number of vehicles and other parameters, the performance of the proposed algorithm is compared with other algorithms. And the Monte Carlo simulation is executed 100 times for every performance comparison. Detailed simulation parameters are shown in Table 1.

Table 1. Simulation parameters

Parameters	Description	Value
B	Channel bandwidth between RSUs and vehicles	1 MHz
∂	Pass-loss exponent	4
N_0	Noise power	10^{-15} W
P_{RSU}	Transmitted power of RSUs	0.1 W
$2R$	Distance between the adjacent RSUs	100 m
d	One time slot	0.1 s
J	Number of vehicles	12
K	Number of tasks	8
M	Number of RSUs	20
η	Trade-off factor	0.99999
v	The speed of vehicle	U(15,65) km/h
S	Population size of HHO algorithm	30
T	The max iterations of HHO algorithm	500

2. Discussion of simulation results

• Random algorithm: Each task randomly selects a vehicle node to offload and be completed.

• Harris hawks optimization (HHO) algorithm : Harris hawks algorithm is a kind of effective intelligent optimization algorithm inspired by the process of Harris Eagle hunting rabbits. In our scenario, the vehicle node selection problem is a typical combination optimization question and Harris hawks algorithm is also introduced to solve it.

Fig.2 demonstrates the objective function values of the proposed algorithm, HHO algorithm and random algorithm under different task data sizes. It can be seen that the proposed algorithm can obtain the best performance, which is about 15.03% higher than that of HHO algorithm and 35.37% higher than that of random algorithm.

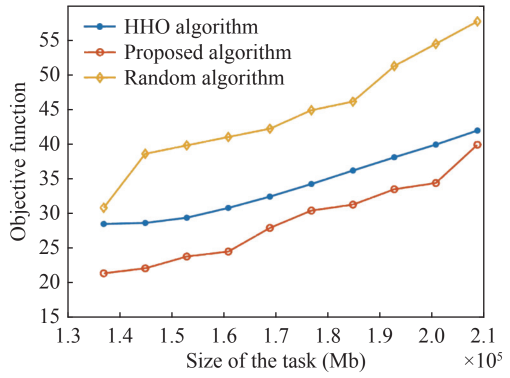

Fig. 2. Objective function with different task sizes.

Fig.3 shows the relationship between delay and different task size. With the increasing of the data size of the task, the delay also increases because bigger data needs more computing ability. And the proposed algorithm can get minimum delay. It is about 9.5% lower than that of HHO algorithm and 19.55% lower than that of random algorithm. Figs.3 and 4 validate that the proposed offloading method can achieve higher performance in terms of cost and latency.

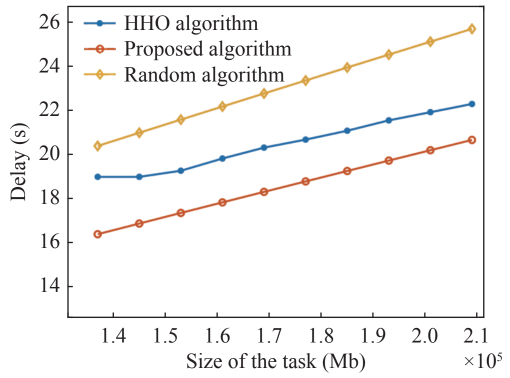

Fig. 3. Delay with different task sizes.

Fig.4 illustrates the relationship between the number of vehicle nodes and the value of the objective function under the proposed algorithm, HHO algorithm and random algorithm. It can be seen that with the increasing of vehicle computing nodes, the value of the objective function shows a downward trend. The reason is that the task can preferentially select the vehicle with stronger computing ability when the number of selectable vehicles increases. Stronger computing ability will result in lower delay and cost. The proposed algorithm can get the best performance in terms of objective function, which is about 21.56% higher than that of HHO algorithm and about 37.92% higher than that of random algorithm.

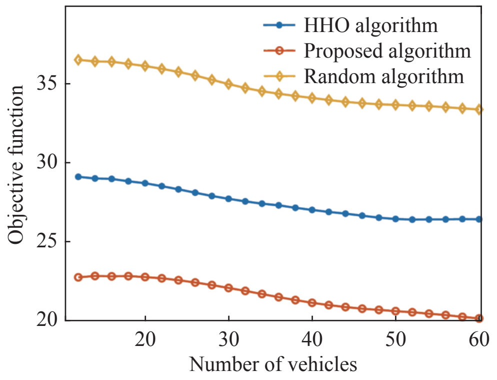

Fig. 4. Objective function with different number of vehicles.

Fig.5 shows the value of objective function of the three algorithms at different vehicle speeds. Obviously, with the increasing of the vehicle speed, the objective function will increase. The main reason is that higher vehicle speed will result in more migrations. This will increase the migration cost of the tasks so as to decrease the performance. In this situation, the performance of the proposed algorithm is about 7.27% higher than that of HHO algorithm and 17.98% higher than that of random algorithm.

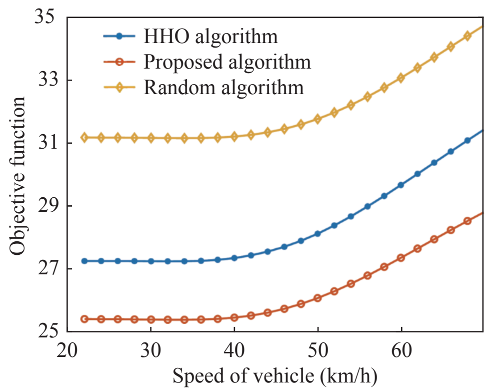

Fig. 5. Objective function with different vehicle speeds.

The trade-off parameter η can be used to adjust the optimization performance index of the system. If η is set to close to 1, the delay dominates the optimization objective. If η is set to close to 0, the reward and migration cost plays a leading role.

Fig.6 shows the delay with different η and number of vehicles. It can be seen that the delay decreases significantly with the increasing of η. Therefore, we can get the optimization results with different emphasis by adjusting η. Meanwhile, more number of vehicles can provide stronger computing ability and more choices to offload the tasks.

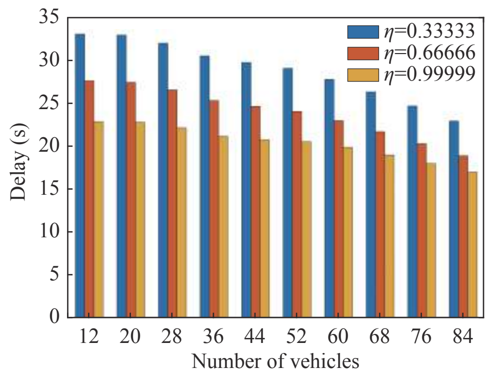

Fig. 6. Delay with different number of vehicles.

V. Conclusions

In this paper, a mobility-aware multi-task migration and offloading scheme for Internet of vehicles is proposed to minimize the total cost of task offloading and migration. Firstly, task offloading model, computing model, migration model and cost model is presented. Then, the problem is formulated as a mixed integer nonlinear programming problem. To solve the thorny problem, it is decoupled into two sub-problems: computing resource allocation problem and vehicle node selection problem. Next, Lagrange function and improved branch and bound algorithm based on linear relaxation are presented to solve the two sub-problems, respectively. Finally, simulation results validate that the proposed algorithm can effectively improve system performance.

References

[1] S. Wang, Y. Lu, J. Zhu, *et al.*, "A novel collision supervision and avoidance algorithm for scalable MAC of vehicular networks," *Chinese Journal of Electronics*, vol.30, no.1, pp.164–170, 2021.

[2] S. Misra and S. Bera, "Soft-VAN: Mobility-aware task offloading in software-defined vehicular network," *IEEE Transactions on Vehicular Technology*, vol.69, no.2, pp.2071–2078, 2020.

[3] N. Cha, C. Wu, T. Yoshinaga, *et al.*, "Virtual edge: Exploring computation offloading in collaborative vehicular edge computing," *IEEE Access*, vol.9, pp.37739–37751, 2021.

[4] K. Zhang, Y. M. Mao, S. P. Leng, *et al.*, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," *IEEE Vehicular Technology Magazine*, vol.12, no.2, pp.36–44, 2017.

[5] S. Secci, P. Raad, and P. Gallard, "Linking virtual machine mobility to user mobility," *IEEE Transactions on Network and Service Management*, vol.13, no.4, pp.927–940, 2016.

[6] H. X. Zhang, Y. J. Yang, X. Z. Huang, *et al.*, "Ultra-low latency multi-task offloading in mobile edge computing," *IEEE Access*, vol.9, pp.32569–32581, 2021.

[7] H. B. Wang, H. L. Xu, H. Huang, *et al.*, "Robust task offloading in dynamic edge computing," *IEEE Transactions on Mobile Computing*, vol.22, no.1, pp.500–514, 2023.

[8] S. Zhou, Y. X. Sun, Z. Y. Jiang, *et al.*, "Exploiting moving intelligence: Delay-optimized computation offloading in vehicular fog networks," *IEEE Communications Magazine*, vol.57, no.5, pp.49–55, 2019.

[9] R. Yadav, W. Z. Zhang, O. Kaiwartya, *et al.*, "Energy-latency tradeoff for dynamic computation offloading in vehicular fog computing," *IEEE Transactions on Vehicular Technology*, vol.69, no.12, pp.14198–14211, 2020.

[10] V. H. Hoang, T. M. Ho, and L. B. Le, "Mobility-aware computation offloading in MEC-based vehicular wireless networks," *IEEE Communications Letters*, vol.24, no.2, pp.466–469, 2020.

[11] A. Boukerche and V. Soto, "An efficient mobility-oriented retrieval protocol for computation offloading in vehicular edge multi-access network," *IEEE Transactions on Intelligent Transportation Systems*, vol.21, no.6, pp.2675–2688, 2020.

[12] H. Hu, Q. Wang, R. Q. Hu, *et al.*, "Mobility-aware offloading and resource allocation in a MEC-enabled IoT network with energy harvesting," *IEEE Internet of Things Journal*, vol.8, no.24, pp.17541–17556, 2021.

[13] C. L. Xu, C. Xu, B. Li, *et al.*, "Joint social-aware and mobility-aware computation offloading in heterogeneous mobile edge computing," *IEEE Access*, vol.10, pp.28600–28613, 2022.

[14] C. X. Li, H. Wang, and R. F. Song, "Mobility-aware offloading and resource allocation in NOMA-MEC systems via DC," *IEEE Communications Letters*, vol.26, no.5, pp.1091–1095, 2022.

[15] C. Yang, Y. Liu, X. Chen, *et al.*, "Efficient mobility-aware task offloading for vehicular edge computing networks," *IEEE Access*, vol.7, pp.26652–26664, 2019.

[16] S. Thananjeyan, C. A. Chan, E. Wong, *et al.*, "Deployment and resource distribution of mobile edge hosts based on correlated user mobility," *IEEE Access*, vol.7, pp.148–159, 2019.

[17] Y. Shi, S. Z. Chen, and X. Xu, "MAGA: A mobility-aware computation offloading decision for distributed mobile cloud computing," *IEEE Internet of Things Journal*, vol.5, no.1, pp.164–174, 2018.

[18] Z. Z. Liang, Y. Liu, T. M. Lok, *et al.*, "Multi-cell mobile edge computing: Joint service migration and resource allocation," *IEEE Transactions on Wireless Communications*, vol.20, no.9, pp.5898–5912, 2021.

[19] D. Y. Wang, Z. L. Liu, X. X. Wang, *et al.*, "Mobility-aware task offloading and migration schemes in fog computing networks," *IEEE Access*, vol.7, pp.43356–43368, 2019.

[20] M. Islam, A. Razzaque, M. M. Hassan, *et al.*, "Mobile cloud-based big healthcare data processing in smart cities," *IEEE Access*, vol.5, pp.11887–11899, 2017.

[21] X. H. Deng, Z. H. Sun, D. Li, *et al.*, "User-centric computation offloading for edge computing," *IEEE Internet of Things Journal*, vol.8, no.16, pp.12559–12568, 2021.

LI Xujie received Ph.D. degree in communication engineering from National Mobile Communications Research Lab., Southeast University in 2012. From 2016 to 2017, He was a Visiting Associate Professor with the University of Warwick, UK. Currently, he is working as a Professor in the College of Computer and Information Engineering in Hohai University now. He is an Associate Editor of the *IEEE Access*. He serves as a reviewer for many journals and conferences, such as *IEEE JSAC*, *TCOM*, *TWC*, *IET Com*, TSP, GLOBECOM, etc. His research interests include fog computing network, device-to-device (D2D) communications, energy efficient wireless sensor networks, and small cell technique.
(Email: lixujie@hhu.edu.cn)

TANG Jing received the B.S. degree from the School of Electronic Information, Huaiyin Normal University, Huai'an, China, in 2020. She is currently working toward the M.S. degree in signal and information processing with the School of Computer and Information, Hohai University, Nanjing, China. Her current research interests include Internet of vehicles, task offloading, mobile edge computing and task migration. (Email: 201307020028@hhu.edu.cn)

XU Yuan received the B.S. degree from the School of Electronic Engineering, Jiangsu Ocean University, Lianyungang, China, in 2020. She is currently working toward the M.S. degree in signal and information processing with the School of Computer and Information, Hohai University, Nanjing, China. Her research interests include Internet of vehicles, task offloading, blockchain and smart contract.
(Email: 201307020031@hhu.edu.cn)

SUN Ying received the B.S. and M.S. degrees in communication engineering from Hohai Universtiy, Nanjing, China, in 2012 and 2019, respectively. Now she is currently pursuing the Ph.D. degree at the College of Computer and Information Engineering in Hohai University. Her current research interests include space-ground integrated networks, resource allocation, D2D communication and system performance analysis. (Email: sunying2016@hhu.edu.cn)