Received: 29 January 2021 | Revised: 5 June 2021 | Accepted: 19 July 2021

DOI: 10.1002/dac.4947

RESEARCH ARTICLE

WILEY

Stackelberg game-based task offloading in vehicular edge computing networks

Shuang Liu[1] | **Jie Tian**[1] | **Xiaofang Deng**[2] | **Yuan Zhi**[1] | **Ji Bian**[1]

[1]School of Information Science and Engineering, Shandong Normal University, Jinan, China

[2]School of Information and Communication, Guilin University of Electronic Technology, Guilin, China

Correspondence
Jie Tian, School of Information Science and Engineering, Shandong Normal University, Jinan 250358, China.
Email: tianjie@sdnu.edu.cn

Funding information
National Natural Science Foundation of China, Grant/Award Numbers: 61801278, 61972237, 61901247; Shandong Provincial scientific research programs in colleges and universities, Grant/Award Number: J18KA310; Key Laboratory of Cognitive Radio and Information Processing, Ministry of Education (Guilin University of Electronic Technology), Grant/Award Number: CRKL190205; Shandong Provincial Natural Science Foundation for Young Scholars of China, Grant/Award Number: ZR2020QF001

Summary

With the emergence of intelligent vehicles, how to satisfy the demands of the vehicles with computing-intensive and delay-sensitive tasks has become a challenging issue. Vehicular edge computing (VEC) is proposed as an advanced paradigm to improve the service of vehicles through offloading the task to the VEC servers. Nevertheless, VEC servers always have limited computation resources and do not satisfy the offloading requirements of vehicles. To this end, in this paper, we propose a more flexible offloading scheme by jointly considering the offloading strategies and the price strategy. In the proposed scheme, where the task can be dynamically divided into two parts parallel executed at the vehicles and VEC servers. A multi-leader and multi-follower Stackelberg game -based distributed algorithm is proposed to maximize the utilities of the vehicles and the VEC servers under the delay constraint. Finally, the game equilibrium is analyzed and achieved. Extensive experiments demonstrate that the proposed offloading scheme converges fast and always outperforms the existing schemes in terms of the vehicular utility under different network conditions. For instance, the proposed scheme achieves the utility improvement over 56.62% compared to the fixed selection strategy and achieves the utility improvement up to 161.0% compared to the complete offloading with fixed price strategy when the number of vehicles is 10. Additionally, the effects of key parameters such as the offloading strategies and, the price strategy, and the computation resource on the average utility of vehicles are also discussed and analyzed based on the simulation results.

KEYWORDS

vehicular edge computing (VEC), partial offloading, Stackelberg game, vehicular networks

1 | INTRODUCTION

The advancements in the Internet of things (IoT) and wireless communication technologies make vehicles more intelligent[1–3] and promote the development of Intelligent Transportation Systems (ITS). One of ITS' critical functions is to predict the future condition of traffic[4] and support various vehicle-to-everything (V2X) applications that improve road safety. Gradually, as a key branch of the IoT, the Internet of vehicles (IoV) has become an indispensable part of modern transportation and plays an important role in some emerging scenarios, such as autonomous driving and natural language processing.[5] Along with the diversifications of the vehicles applications such as compute-intensive tasks

and delay-sensitive tasks, how to satisfy the quality of service (QoS) requirements of the vehicles' tasks has become a challenge in the vehicular networks.[6–8]

In the conventional task offloading method, vehicles attempt to cope with the exploding demands of vehicular users' tasks by offloading computing tasks to remote servers via cloud-based vehicular networks.[9] However, the latency fluctuations during task offloading may reduce offloading efficiency, and the long-distance transmission between the vehicle and remote server also decreases the QoS.[10] In order to cope with above problems, mobile edge computing (MEC) is proposed as an advanced new technology[11–13] which can mitigate the heavy computation pressure of the vehicles by offloading their tasks to the MEC servers. Because of the proximity, the MEC server is able to provide a fast interactive response in the computation offloading service.[14] VEC which combines vehicular networks and MEC is considered as a potential paradigm, which has received much attention by deploying mobile edge servers in the IoV to satisfy the demands of intelligent applications of vehicles. VEC can reduce the computation response time and alleviate the traffic congestion in the capacity-limited back-haul link by extending the computation capability to the vehicular network edge. The quality of vehicular service can be greatly enhanced by offloading computation-intensive applications from the resource-constrained vehicles to the VEC servers.

In VEC networks, task offloading is a key problem that has been widely studied.[15] Vehicles always offload their tasks to VEC servers by accessing Road Side Units (RSUs). However, the computation capacity of the VEC servers located at the network edge is limited, especially in networks with intensive tasks. It is essential to propose a more flexible and effective offloading scheme. Generally, there are two kinds of conventional offloading schemes, that is, computing locally and offloading totally to the VEC servers. Computing the task locally does not need task transmission, but it may cause a large computation latency due to the limited computation resource. Compared with local execution, offloading to the VEC servers has lower latency due to the larger computation resources. However, the computation resources of the VEC server are always limited, which prevents the VEC server from fully satisfying the vehicle offloading requirements within the specified latency constraints, especially for dense traffic flows. Recently, the most common offloading paradigms are binary offloading and partial offloading. Binary offloading is an offloading scheme where the vehicle decides on its own whether to compute the task locally or offload it to the VEC server. For binary offloading, the computation tasks at vehicles cannot be partitioned and must be executed as a whole either at the vehicle or at the VEC server. However, binary offloading also faces the problem of limited computation resources, so it is more reasonable to use partial offloading where the vehicle's task can be offloaded partially to the VEC server.

Motivated by the aforementioned observation, in this paper, we investigate how to dynamically offload parts of computation tasks from vehicles to VEC servers in a multi-vehicle and multi-server network. We propose a flexible offloading scheme in order to address the following critical points: (1) the offloading strategies of vehicles: how much the task of the vehicle to be offloaded to the VEC server and which VEC server to be chosen. (2) The price strategy charged by VEC servers. The main contributions of this paper are summarized as follows:

(1) We propose a more flexible partial offloading scheme by considering the offloading strategies and price strategy in a multi-server and multi-vehicle network.
(2) We formulate the utility maximization problems of both vehicles and VEC servers as a multi-leader and multi-follower Stackelberg game by taking offloading strategies of vehicles and price strategy of VEC servers into account to optimize offloading latency and cost.
(3) To solve the above problems, we divide the optimization problem into two sub-problems. A distributed algorithm is proposed to find offloading strategies of vehicles and the price strategy of VEC servers. The existence of Stackelberg game equilibrium is analyzed and proved. Simulation results demonstrate the superiority of the proposed scheme by comparing it with other existing solutions.

The rest of the paper is organized as follows. Related works on offloading are presented in Section 2. The system model is described in Section 3. In Section 4, we model the computation offloading among vehicles and VEC servers as a Stackelberg game and formulate the utilities and optimization problems of both vehicles and VEC servers. In Section 5, we introduce the solution to the optimization problems of the vehicles and VEC servers. The Lagrange dual method is utilized to solve the vehicles' side optimization problem, and the existence of the Stackelberg equilibrium is proved; and we also propose an optimal distributed algorithm. We present numerical results in Section 6 and conclude the paper in Section 7.

2 | RELATED WORK

There have been a number of works concentrating on computation offloading in vehicular or cellular networks. The binary offloading decision, where each user can independently choose whether to compute the task locally or to offload the task to the edge servers, has been studied. For example, Zhang et al[16] adopted a Stackelberg game-theoretic approach by offloading the whole task to the VEC server. Wu et al[17] put forward an efficient binary offloading algorithm based on support vector machine (SVMO) to satisfy the fast offloading demand with a high decision accuracy. Du et al.[18] minimized the cost of vehicles, and Nguyen et al[19] improved vehicular service and reduced energy consumption by offloading their task to the VEC server. However, the offloading efficiency of binary offloading will be affected by the limited computation resources. Thus, some studies proposed partial offloading, where the input data were divided into two parts for local and edge computation.[20,21] You et al[20] applied a partial offloading scheme to optimized resource allocation, and Dai et al[21] formulated the joint load balancing and offloading problem as a mixed-integer nonlinear programming problem. Compared to binary offloading, it is more reasonable and flexible to offload partially to edge servers, since the edge servers' resources are limited. Moreover, considering that if it only has one edge server located at the VEC network, the demands of the vehicles with computing-intensive and delay-sensitive tasks may not be satisfied. To this end, in this paper, we investigate the partial offloading scheme in a vehicular network consisting of multiple VEC servers and multiple vehicles. We dynamically divide the computation task into two parts and the tasks are computed in local and edge in parallel.

Many works have been done on optimizing time delay or energy consumption of task offloading in the vehicular network based on MEC. Wang et al[22] proposed an energy-efficient computation offloading scheme by jointly optimizing offloading and radio resource allocation to obtain the minimum energy consumption under the latency constraints and proposed a game theoretic solution. Liu et al[23] and Zhan et al[24] formulated the problem as a semi-Markov process and proposed two reinforcement learning methods: Q-learning and deep reinforcement learning (DRL). To optimize the performance of total task accomplishment delay, Tang et al[25] proposed a temporal-spatial computation offloading scheme. To minimize the energy consumption of the vehicles and the whole VEC system, Zhang et al[26] proposed a contract-based computation resource allocation scheme to maximize the benefit of the VEC service provider while enhancing the utilities about the cost of the vehicles. Yang et al[27] designed the two-layer Stackelberg game-based offloading (TSGO) strategy to achieve green communications. Dab et al.[28] proposed a game theoretic approach for joint offloading and resource allocation optimization problem in MEC system to minimize the energy consumption and monetary cost from mobile terminals perspective. Xu et al[29] proposed a blockchain-based computation offloading method with taking advantage of the non-dominated sorting genetic algorithm and simple additive weighting and multiple criteria decision making to obtain the optimal offloading strategy. Zhang et al[30] formulated the offloading problem as a Markov decision process and proposed a DRL-based decentralized algorithm. Aiming to minimize the latency of the tasks, Zhang et al[31] proposed a load-balancing task offloading scheme base on software-defined networking (SDN). However, most recent works prefer to only consider optimizing the utility of vehicles neglecting the utility of the VEC servers. In addition, most of the literature considers the uniform pricing scheme, where the VEC servers set and broadcast a uniform price to all vehicles. In this paper, we take the utility of the vehicles and VEC servers into account and we apply the differentiated pricing.

To bridge the gap of aforementioned researches, in this paper, we propose a more flexible partial offloading scheme in a multi-server and multi-vehicle network, where the input data can be arbitrarily divided into two parts for local and edge computation in parallel. In addition, we also take the utility of the VEC servers into account and the VEC servers can adjust their price according to the offloading strategies of the vehicles. A multi-leader and multi-follower Stackelberg game is formulated to maximize the utilities of both the vehicles and the VEC servers.

3 | SYSTEM MODEL

3.1 | Vehicular edge network model

We consider a crossroad scenario, where there are N vehicles, M RSUs, and some urban building facilities located along the road, as shown in Figure 1. RSUs are located along the two sides of the road. Each RSU is equipped with a VEC server with limited computation resources. We denote the set of these VEC servers as $\mathcal{M} = \{m_1, m_2, ..., m_M\}$. The vehicles are distributed randomly at this crossroad. We denote the set of the vehicles as $\mathcal{N} = \{n_1, n_2, ..., n_N\}$. Each vehicle

FIGURE 1 An example of the considered scenarios

has a computation task, which can be described by a tuple $D_i \triangleq (R_i, L_{i,k}, T_i^{max})$, where R_i denotes the size of input data in total for the vehicle i, and $L_{i,k}$ denotes the part of the input data of vehicle i, which will be computed on VEC server k, T_i^{max} denotes the maximum latency allowed to accomplish the task of vehicle i. The vehicles offload computation tasks to the VEC servers for calculation by accessing the RSUs.

3.2 | Network delay model

The vehicles should transmit their computation tasks to the VEC servers through the RSUs. Denote $x_{i,k} \in \{0, 1\}$ as the selection decision variable, that is, $x_{i,k} = 1$ denotes that vehicle i chooses VEC server k as its target offloading server and offloads $L_{i,k}$ bits to this server; otherwise, $x_{i,k} = 0$. During the offloading process, assume that each vehicle is allowed to offload its task to one VEC server at most. Thus, we have the constraint $\sum_{k=1}^{M} x_{i,k} \leq 1, i \in \mathcal{N}, k \in \mathcal{M}$. Each task can be divided into two parts and parallel executed at the vehicles locally and VEC servers. Specifically, let C_i denote the required number of CPU cycles for computing 1-bit of input data at vehicle i. It is assumed that vehicle i offloads $L_{i,k}$ bits to the VEC server k while the rest $(R_i - L_{i,k})$ bits are computed locally.

The local CPU frequency of vehicle i is denoted as $f_{loc,i}$ which is measured by the number of CPU cycles per second. Thus, when the vehicle i computes its total task locally, the computation time is $T_{loc,i} = R_i / f_{loc,i}$. When vehicle i chooses to offload its parts task to VEC server k, the time cost of vehicle i computing $L_{i,k}$ bits locally is

$$t_{loc,i} = \frac{(R_i - L_{i,k})C_i}{f_{loc,i}}. \tag{1}$$

The latency cost by offloading the task of vehicle i to VEC server k can be divided into two parts. The first one is the wireless transmission time cost by transmitting the offloaded task from vehicle i to VEC server k. The second part is the computation time. Then, the total time cost for the part of the task offloading from vehicle i to VEC server k is

$$t_{off,i,k} = \frac{L_{i,k}}{r_{i,k}} + \frac{L_{i,k}C_i}{f_{i,k}}, \tag{2}$$

where $r_{i,k} = \frac{B}{N_k} log(1 + \frac{p_i h_{i,k}}{\frac{BN_0}{N_k}})$ denotes the uplink transmission rate from vehicle i to VEC server k. Since the channels are orthogonal, there is no interference between vehicles. B is the bandwidth, N_k is the number of the vehicles who choose VEC server k as its target VEC server. p_i is the uplink transmission power of vehicle i, $h_{i,k}$ is the channel gain between vehicle i and VEC server k, we consider Rayleigh fading and path-loss, path loss exponent is δ, and N_0 is the noise power spectrum density, respectively. $f_{i,k}$ is the computational speed of the VEC server k assigned to vehicle i. Here, we consider VEC server resources are allocated equally for the associated vehicles, that is, $f_{i,k} = F_k/N_k$, where F_k is the total computation resources of VEC server k.

Since each task can be divided into two parts, parallel computed locally and at the VEC server simultaneously, the task delay is determined by the maximum value of the two parallel executed parts. Thus, the total task delay for the vehicle i offloading the task to VEC server k is given as

$$T_{i,k} = \max\{t_{loc,i}, t_{off,i,k}\}. \tag{3}$$

4 | STACKELBERG GAME-BASED PROBLEM FORMULATION

In this section, the multi-leader and multi-follower Stackelberg game is utilized to model the offloading system with the purpose of getting the equilibrium between the vehicles and VEC servers. In this game, VEC servers are the leaders, and the vehicles act as the followers by optimally reacting to the VEC servers' strategies. Every vehicle tries to maximize its utility in the form of a non-cooperative game. Given the decision strategy and offloading strategy of each vehicle, the VEC servers make the price strategy until achieving equilibrium. We will analyze and formulate the utilities and optimization problems of the vehicles and the VEC servers, respectively. And the convexity of the formulated problem of vehicles is proved.

4.1 | Problem formulation for vehicle

Different from task offloading in the previous works which only consider the utility of vehicles by minimizing time delay, our proposed scheme has formulated the utility for the vehicles and VEC servers respectively. For the utility of vehicles, we aim to maximize the difference between the delay of all tasks calculated locally and the delay of partial offloading and minimize the cost charged by the VEC server.

The utility of each vehicle includes its latency and the payment charged by the VEC server; thus, the utility of vehicle i offloading its part task to VEC server k is expressed as

$$U_i(\mathbf{x}, \boldsymbol{L}) = \sum_{k=1}^{M} x_{i,k}(\theta_i \log(T_{loc,i} - T_{i,k}(L) + 1) - \mu_{k,i} L_{i,k} C_i), \tag{4}$$

where θ_i is a positive constant and $\mu_{k,i}$ is the price charged by VEC server k for the vehicle i.

Since each task can be divided into two parts to execute simultaneously, the task processing delay is determined by the maximum value of the two parallel executed parts according to the $L_{i,k}$. Thus, the utility of vehicle i is equivalent to

$$U_i(\mathbf{x}, \boldsymbol{L}) = \begin{cases} \sum_{k=1}^{M} x_{i,k}(\theta_i \log(T_{loc,i} - t_{loc,i} + 1) - \mu_{k,i} L_{i,k} C_i), \text{ if } 0 \le L_{i,k} < z_i; \\ \sum_{k=1}^{M} x_{i,k}(\theta_i \log(T_{loc,i} - t_{off,i,k}(\boldsymbol{L}) + 1) - \mu_{k,i} L_{i,k} C_i), \text{ if } z_i < L_{i,k} \le R_i; \end{cases} \tag{5}$$

where $z_i = \frac{R_i C_i}{f_{loc,i}(\frac{1}{r_{i,k}} + \frac{C_i}{f_{loc,i}} + \frac{C_i}{f_{i,k}})}$.

The objective of any vehicle i is to maximize its utility. To this end, according to Equation (4), the vehicle i needs to increase the difference between $T_{loc,i}$ and $T_{i,k}$ and reduce the cost paid to the server k. Then the optimization problem for vehicle i can be formulated as follows. Define $\boldsymbol{x} = \{x_{i,k}, \forall i \in \mathcal{N}, \forall k \in \mathcal{M}\}$ as the vector of VEC server selection

decision and $\boldsymbol{L}=\{L_{i,k},\forall i\in\mathcal{N},\forall k\in\mathcal{M}\}$ as the offloading allocation vector, respectively. Then, we formulate the optimization problem for vehicle *i*as follows:

$$\begin{aligned} P1:\ & \max_{\{\mathbf{x},\ \boldsymbol{L}\}} U_i(\mathbf{x},\boldsymbol{L}) \\ & s.t.\quad C_1:\ \sum_{k=1}^{M} x_{i,k}T_{i,k}\le T_i^{max}, \quad \forall i\in\mathcal{N} \\ & \qquad C_2:\ \sum_{i=1}^{N} x_{i,k}L_{i,k}\le S_k^{max}, \quad \forall k\in\mathcal{M} \\ & \qquad C_3:\ x_{i,k}(R_i-L_{i,k})\ge 0, \quad \forall i\in\mathcal{N},k\in\mathcal{M} \\ & \qquad C_4:\ x_{i,k}=\{0,1\}, \quad \forall i\in\mathcal{N},k\in\mathcal{M} \\ & \qquad C_5:\ \sum_{k=1}^{M} x_{i,k}\le 1, \quad \forall i\in\mathcal{N} \end{aligned} \tag{6}$$

where, the first constraint C_1 guarantees that the task processing delay cannot exceed the maximum allowed latency. The second constraint C_2 ensures that the sum of bits that the vehicles choose to offload to the same VEC server cannot exceed the maximum allowed processing capacity. The third constraint C_3 restricts the number of bits offloaded by the vehicle i cannot exceed the total number of bits in the task D_i. Constraints C_4 and C_5 ensure that each vehicle offloads its task to only one VEC server.

To solve above optimization problem, the time-sharing method is utilized by introducing time-sharing factor.[32] We relax the binary variable $x_{i,\,k}$ into a real value in [0, 1]. Meanwhile, in order to solve the optimization problem, a variable is introduced, we define $s_{i,\,k}=x_{i,\,k}L_{i,\,k}$. Then, the strategy set of vehicle i can be denoted as $\boldsymbol{s}_i=\{s_{i,1},s_{i,2},...,s_{i,M}\}$. Thus, the strategy space of the followers can be denoted as $\boldsymbol{s}=\{\boldsymbol{s}_1\times\boldsymbol{s}_2\times...\times\boldsymbol{s}_N\}$.

The original problem P1 in (6) can be transformed as follows:

$$\begin{aligned} P2:\ & \max_{\{\mathbf{x},\ \boldsymbol{s}\}} U_i(\mathbf{x},\boldsymbol{s}) \\ & s.t.\quad C_1:\ \sum_{k=1}^{M} x_{i,k}T_{i,k}\le T_i^{max}, \quad \forall i\in\mathcal{N} \\ & \qquad C_2:\ \sum_{i=1}^{N} s_{i,k}\le S_k^{max}, \quad \forall k\in\mathcal{M} \\ & \qquad C_3:\ x_{i,k}R_i-s_{i,k}\ge 0, \quad \forall i\in\mathcal{N},k\in\mathcal{M} \\ & \qquad C_4:\ x_{i,k}=[0,1], \quad \forall i\in\mathcal{N},k\in\mathcal{M} \\ & \qquad C_5:\ \sum_{k=1}^{M} x_{i,k}\le 1, \quad \forall i\in\mathcal{N} \end{aligned} \tag{7}$$

Definition 1. Suppose that f is twice differentiable. Then f is convex if and only if **dom** f is convex and Hessian of f is PSD (positive semidefinite) for all $\boldsymbol{x}\in$ **dom** f, that is,

$$\bigtriangledown^2 f(x)\succeq 0,\forall\boldsymbol{x}\in\mathbf{dom}\ \ f, \tag{8}$$

otherwise, if

$$\bigtriangledown^2 f(x)\preccurlyeq 0,\forall\boldsymbol{x}\in\mathbf{dom}\ \ f, \tag{9}$$

f is concave.

Lemma 1. Problem P2 in (7) is a concave optimization problem.

Proof. The vehicle i chooses to offload its part of task to the suitable VEC server k. We obtain the utility function of the vehicle i.

The second-order derivative of the U_i with respect to $s_{i,k}$ is given by

i. $T_{i,k} = t_{loc,i}$

$$\frac{\partial^2 U_i}{\partial s_{i,k}^2} = -\frac{\frac{\theta_i}{x_{i,k}}\left(\frac{C_i}{f_{loc,i}}\right)^2}{\left(\frac{s_{i,k}C_i}{x_{i,k}f_{loc,i}}+1\right)^2} \leq 0, \tag{10}$$

ii. $T_{i,k} = t_{off,i,k}$

$$\frac{\partial^2 U_i}{\partial s_{i,k}^2} = -\frac{\frac{\theta_i}{x_{i,k}}\left(\frac{1}{r_{i,k}}+\frac{C_i}{f_{i,k}}\right)^2}{\left(\frac{R_i C_i}{f_{loc,i}}-\frac{s_{i,k}}{x_{i,k}r_{i,k}}-\frac{s_{i,k}C_i}{x_{i,k}f_{i,k}}+1\right)^2} \leq 0. \tag{11}$$

The Hessian matrix constructed from the second-order derivatives in terms of $s_{i,k}$ is defined as

$$\boldsymbol{H}_{U_i}(\boldsymbol{s}) = \begin{bmatrix} \frac{\partial^2 U_i}{\partial s_{1,1}^2} & \frac{\partial^2 U_i}{\partial s_{1,2}^2} & \cdots & \frac{\partial^2 U_i}{\partial s_{1,m}^2} \\ \frac{\partial^2 U_i}{\partial s_{2,1}^2} & \frac{\partial^2 U_i}{\partial s_{2,2}^2} & \cdots & \frac{\partial^2 U_i}{\partial s_{2,m}^2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 U_i}{\partial s_{n,1}^2} & \frac{\partial^2 U_i}{\partial s_{n,2}^2} & \cdots & \frac{\partial^2 U_i}{\partial s_{n,m}^2} \end{bmatrix} \preccurlyeq 0. \tag{12}$$

The second-order derivative of the U_i with respect to $x_{i,k}$ is given by

i. $T_{i,k} = t_{loc,i}$

$$\frac{\partial^2 U_i}{\partial x_{i,k}^2} = 0, \tag{13}$$

ii. $T_{i,k} = t_{off,i,k}$

$$\frac{\partial^2 U_i}{\partial x_{i,k}^2} = 0. \tag{14}$$

The Hessian matrix constructed from the second-order derivatives in terms of $x_{i,k}$ is defined as

$$\boldsymbol{H}_{U_i}(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial^2 U_i}{\partial x_{1,1}^2} & \frac{\partial^2 U_i}{\partial x_{1,2}^2} & \cdots & \frac{\partial^2 U_i}{\partial x_{1,m}^2} \\ \frac{\partial^2 U_i}{\partial x_{2,1}^2} & \frac{\partial^2 U_i}{\partial x_{2,2}^2} & \cdots & \frac{\partial^2 U_i}{\partial x_{2,m}^2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 U_i}{\partial s_{n,1}^2} & \frac{\partial^2 U_i}{\partial x_{n,2}^2} & \cdots & \frac{\partial^2 U_i}{\partial x_{n,m}^2} \end{bmatrix} \preccurlyeq 0. \tag{15}$$

According to Definition 1, the utility function U_i in terms of variable $s_{i,k}$ and variable $x_{i,k}$ is concave. In addition, considering that the constraints are also linear and convex, it is proved that the problem p2 in (7) is concave optimization problem.

4.2 | Problem formulation for VEC server

For the utility of the VEC server, we aim to maximize the revenue of the VEC server. To achieve low latency, the vehicle prefers to offload more data to the VEC server. However, first, because of the limitation of computation resources, offloading can be less efficient if vehicles offload too much data. Second, if vehicles offload too much data, it will cost a lot. Therefore, there is a trade-off between latency and cost. The utility of the VEC server is defined as the revenue achieved by selling the VEC computation resources to vehicles. The revenue of the VEC server k is expressed as

$$U_k(\boldsymbol{\mu}) = \sum_{k=1}^{N} \mu_{k,i} L_{i,k} C_i. \tag{16}$$

The goal of the VEC servers is to maximize their revenue. The more tasks vehicles try to offload to the VEC server, the more revenue the VEC server will obtain. Since each vehicle chooses the most optimal VEC server to offload its task under the specified delay constraint, there will be competition among them. Mathematically, the optimization problem at the VEC server k can be expressed as

$$\begin{aligned} P3: \quad & \max_{\{\mu_{k,i}, \forall i \in \mathcal{N}, \forall k \in \mathcal{M}\}} U_k(\boldsymbol{\mu}) \\ & s.t. \quad \mu_{k,i} \geq 0, \quad \forall i \in \mathcal{N}, \forall k \in \mathcal{M}. \end{aligned} \tag{17}$$

Based on the aforementioned analysis, the mathematical model under the problem formulation in this work is reasonable and practical, and it can balance the load among vehicles and VEC servers.

5 | SOLUTION OF THE OPTIMIZATION PROBLEMS

In this section, the Lagrange dual method is utilized to solve the vehicles' side optimization problem. We get the closed expression of the optimization problem of the VEC server by substituting the expression of offloading strategies and the Lagrangian multipliers. Then, the Stackelberg equilibrium is proved, and a distributed algorithm is proposed to achieve the point of equilibrium among VEC servers (leaders) and vehicles (followers), and to seek the optimal strategies of vehicles and VEC servers.

5.1 | Solution of vehicle side

The Lagrange dual method is utilized to solve the vehicles' side optimization problem. For optimization problem P2, the Lagrange function of vehicle i is given by

$$\begin{aligned} \mathcal{L}(\boldsymbol{x},\boldsymbol{s},\boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda}) \; = & \sum_{k=1}^{M} x_{i,k}(\theta_i \log(T_{loc,i} - T_{i,k} + 1) - \mu_{k,i} L_{i,k} C_i) + \alpha_i (T_i^{max} - \sum_{k=1}^{M} x_{i,k} T_{i,k}) \\ & + \varphi_k (S_k^{max} - \sum_{i=1}^{N} \sum_{k=1}^{M} s_{i,k}) + \lambda_i (\sum_{k=1}^{M} x_{i,k} R_i - s_{i,k}) + \omega_i (1 - \sum_{k=1}^{M} x_{i,k}), \end{aligned} \tag{18}$$

where α, φ, ω, λ are the vectors of the Lagrangian multipliers. Then, the Lagrange dual function is given by

$$\mathcal{D}(\boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda}) = \max\ \mathcal{L}(\boldsymbol{x},\boldsymbol{s},\boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda}), \tag{19}$$

and the dual problem is

$$\begin{aligned} &\min \mathcal{D}(\boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda}) \\ &s.t.\ \boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda} \succ 0. \end{aligned} \tag{20}$$

According to the Karush–Kuhn–Tucker (KKT) condition,[33,34] we can obtain the expression of the variables we need. First, we should determine the selection of the vehicles. We take the first-order derivative of $\mathcal{L}$ with respect to $x_{i,k}$ and let it equal to zero, then, we can obtain the optimal solution expression of the $\hat{x}_{i,k}$, $\hat{s}_{i,k}$.

However, as shown in the formulation of latency $T_{i,k}$, there exists two cases considering the different number of the task, below we will illustrate the two cases, that is, $T_{i,k} = t_{loc,i}$ and $T_{i,k} = t_{off,i,k}$, respectively.

i. $T_{i,k} = t_{loc,i}$

In this case, the Lagrange function is equivalent to

$$\begin{aligned} \mathcal{L}(\boldsymbol{x},\boldsymbol{s},\boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda}) &= \sum_{k=1}^{M} x_{i,k}\theta_i \log\left(\frac{s_{i,k}C_i}{x_{i,k}f_{loc,i}}+1\right) - \mu_{k,i}s_{i,k}C_i + \alpha_i\left(T_i^{max} - \sum_{k=1}^{M} x_{i,k}T_{i,k}\right) \\ &+ \left(\varphi_k\left(S_k^{max} - \sum_{i=1}^{N}\sum_{k=1}^{M} s_{i,k}\right)\right) + \lambda_i \sum_{k=1}^{M} x_{i,k}R_i - s_{i,k} + \omega_i\left(1 - \sum_{k=1}^{M} x_{i,k}\right). \end{aligned} \tag{21}$$

Take the first-order derivative with respect to $s_{i,k}$ and let it result to zero; we get the optimal expression of $\hat{s}_{i,k}$:

$$\hat{s}_{i,k} = \frac{x_{i,k}\theta_i}{\frac{-\alpha_i C_i}{f_{loc,i}} + \mu_{k,i}C_i + \varphi_k + \lambda_i} - \frac{x_{i,k}f_{loc,i}}{C_i}. \tag{22}$$

The optimal $\hat{L}_{i,k}$ is equal to $\hat{s}_{i,k}$ divided by $x_{i,k}$:

$$\hat{L}_{i,k} = \frac{\hat{s}_{i,k}}{x_{i,k}} = \frac{\theta_i}{\frac{-\alpha_i C_i}{f_{loc,i}} + \mu_{k,i}C_i + \varphi_k + \lambda_i} - \frac{f_{loc,i}}{C_i}. \tag{23}$$

Take the first-order derivative with respect to $x_{i,k}$:

$$\frac{\partial \mathcal{L}(\boldsymbol{x},\boldsymbol{s},\boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda})}{\partial x_{i,k}} \begin{cases} < 0, & if \quad \hat{x}_{i,k} = 0, \\ = 0, & if \quad \hat{x}_{i,k} \in (0,1), \\ > 0, & if \quad \hat{x}_{i,k} = 1. \end{cases} \tag{24}$$

By solving the above formulation, we obtain

$$\begin{aligned} &\frac{\partial \mathcal{L}(\boldsymbol{x},\boldsymbol{s},\boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda})}{\partial x_{i,k}} = \theta_i \log\left(\frac{\hat{L}_{i,k}C_i}{f_{loc,i}}+1\right) - \mu_{k,i}\hat{L}_{i,k}C_i - \\ &\alpha_i\frac{(R_i - \hat{L}_{i,k})C_i}{f_{loc,i}} - \varphi_k\hat{L}_{i,k} - \mu_{k,i}\hat{L}_{i,k}C_i - \lambda(R_i - \hat{L}_{i,k}) - \omega_i \\ &= H_{i,k} - \omega_i. \end{aligned} \tag{25}$$

ii. $T_{i,k} = t_{off,i,k}$

In this case, compared with the first case which is the $T_{i,k} = t_{loc,i}$, only the latency is different, we will list all the required variables expression below:

For this case, the Lagrange function is equivalent to

$$\begin{aligned}\mathcal{L}(\boldsymbol{x},\boldsymbol{s},\boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda}) &= \sum_{k=1}^{M} x_{i,k}\left(\theta_i \log\left(\frac{R_i C_i}{f_{loc,i}} - \left(\frac{s_{i,k}}{x_{i,k} r_{i,k}} + \frac{s_{i,k} C_i}{x_{i,k} f_{loc,i}}\right) + 1\right) - \mu_{k,i} s_{i,k} C_i + \left(\alpha_i \left(T_i^{max} - \sum_{k=1}^{M} x_{i,k} T_{i,k}\right)\right)\right. \\ &\left. + \left(\varphi_k \left(S_k^{max} - \sum_{i=1}^{N}\sum_{k=1}^{M} s_{i,k}\right)\right) + \lambda_i \sum_{k=1}^{M} x_{i,k} R_i - s_{i,k} + \omega_i \left(1 - \sum_{k=1}^{M} x_{i,k}\right)\right.\end{aligned} \tag{26}$$

The optimal expression of $\hat{s}_{i,k}$:

$$\hat{s}_{i,k} = \frac{1 + \frac{R_i C_i}{f_{loc,i}} + \frac{\theta_i \left(\frac{1}{r_{i,k}} + \frac{C_i}{f_{i,k}}\right)}{\alpha_i \left(\frac{1}{r_{i,k}} + \frac{C_i}{f_{i,k}}\right) + \varphi_k + \lambda_i + \mu_{k,i} C_i}}{\frac{1}{r_{i,k} x_{i,k}} + \frac{C_i}{x_{i,k} f_{i,k}}}. \tag{27}$$

The optimal expression of $\hat{L}_{i,k}$:

$$\hat{L}_{i,k} = \frac{1 + \frac{R_i C_i}{f_{loc,i}} + \frac{\theta_i \left(\frac{1}{r_{i,k}} + \frac{C_i}{f_{i,k}}\right)}{\alpha_i \left(\frac{1}{r_{i,k}} + \frac{C_i}{f_{i,k}}\right) + \varphi_k + \lambda_i + \mu_{k,i} C_i}}{\frac{1}{r_{i,k}} + \frac{C_i}{f_{i,k}}}. \tag{28}$$

Similarly, we also can obtain the

$$\begin{aligned}&\frac{\partial \mathcal{L}(\boldsymbol{x},\boldsymbol{s},\boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda})}{\partial x_{i,k}} = \theta_i \log\left(\frac{R_i C_i}{f_{loc,i}} - \left(\frac{\hat{L}_{i,k}}{r_{i,k}} + \frac{\hat{L}_{i,k} C_i}{f_{loc,i}}\right) + 1\right) - \\ &\alpha_i \frac{(R_i - \hat{L}_{i,k}) C_i}{f_{loc,i}} - \varphi_k \hat{L}_{i,k} - \mu_{k,i} \hat{L}_{i,k} C_i - \lambda_i (R_i - \hat{L}_{i,k}) - \omega_i \\ &= H_{i,k} - \omega_i.\end{aligned} \tag{29}$$

In summary, the vehicle will choose the VEC server with the largest $H_{i,k}$, which is denoted as

$$\hat{x}_{i,k^*} = 1|_{k^* = \max_k H_{i,k}}. \tag{30}$$

The gradients of the Lagrangian multipliers can be obtained as

$$\begin{aligned}\frac{\partial \mathcal{L}(\boldsymbol{x},\boldsymbol{s},\boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda})}{\alpha_i} &= T_i^{max} - \sum_{k=1}^{M} x_{i,k} T_{i,k}, \\ \frac{\partial \mathcal{L}(\boldsymbol{x},\boldsymbol{s},\boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda})}{\varphi_k} &= S_k^{max} - \sum_{k=1}^{N_k} x_{i,k} L_{i,k}, \\ \frac{\partial \mathcal{L}(\boldsymbol{x},\boldsymbol{s},\boldsymbol{\alpha},\boldsymbol{\varphi},\boldsymbol{\omega},\boldsymbol{\lambda})}{\lambda_i} &= \sum_{k=1}^{M} x_{i,k} R_i - s_{i,k}.\end{aligned} \tag{31}$$

By using the gradient method, the Lagrangian multipliers are updated iteratively as follows:

$$\begin{aligned}
\alpha_i(t_2+1) &= \left[\alpha_i(t_2)+\phi_{\alpha_i}\left(T_i^{max}-\sum_{k=1}^{M}x_{i,k}T_{i,k}\right)\right]^+, \forall i\in\mathcal{N}\\
\varphi_k(t_2+1) &= \left[\varphi_k(t_2)+\phi_{\varphi_k}\left(S_k^{max}-\sum_{k=1}^{N_k}s_{i,k}\right)\right]^+, \forall k\in\mathcal{M}\\
\lambda_i(t_2+1) &= \left[\lambda_i(t_2)+\phi_{\lambda_i}\left(\sum_{k=1}^{M}x_{i,k}R_i-s_{i,k}\right)\right]^+, \forall i\in\mathcal{N}
\end{aligned} \tag{32}$$

where $\phi_{\alpha_i},\phi_{\varphi_k},\phi_{\lambda_i}>0$ are the gradient steps, $m$represents the iteration number. $[\cdot\ ]^+$ represents $\max(0,\cdot)$.

5.2 | Solution of VEC server side

Substituting the expression of $\hat{L}_{i,k}$ into (12) and taking the first-order derivative with respect to $\mu_{k,i}$ and let it result to zero, we can obtain the optimal expression of $\hat{\mu}_{k,i}$:

i. $T_{i,k}=t_{loc,i}$

$$\hat{\mu}_{k,i}=\frac{\hat{L}_{i,k}\left(\frac{-\alpha_i C_i}{f_{loc,i}}+\mu_{k,i}C_i+\varphi_k+\lambda_i\right)^2}{\theta_i C_i}, \tag{33}$$

ii. $T_{i,k}=t_{off,i,k}$

$$\hat{\mu}_{k,i}=\frac{\hat{L}_{i,k}\left(-\alpha_i\left(\frac{1}{r_{i,k}}+\frac{C_i}{f_{c,i,k}}\right)-\mu_{k,i}C_i-\varphi_k-\lambda_i\right)^2}{\theta_i\left(\frac{1}{r_{i,k}}+\frac{C_i}{f_{i,k}}\right)C_i}. \tag{34}$$

The price strategies of VEC servers have an influence on the decision strategy and the offloading data sizes of the vehicles, which also impact the revenue of the VEC servers in turn.

5.3 | Stackelberg equilibrium and distributed algorithm

In this section, we prove the existence of Stackelberg equilibrium of the game firstly, then we propose an distributed algorithm to acquire the optimal strategies of vehicles and VEC servers, and finally analyze the computational complexity of the distributed algorithm.

5.3.1 | Stackelberg equilibrium

In Stackelberg game of this paper, the VEC servers are the leaders while the vehicles are the followers. The Stackelberg equilibrium is a stable and the optimal point in the game, where the leaders and followers have no incentive to increase their profit without changing their strategy unilaterally, given all other players' strategies. There exists a Stackelberg equilibrium in the proposed Stackelberg game, if the following conditions are satisfied:

(1) Given the strategy set of the VEC servers, the VEC servers (leaders) firstly achieve a Nash equilibrium:

$$U_k(\mu_{k,i}^*,\boldsymbol{\mu}_{k,i}^*)\geq U_k(\mu_{k,i},\boldsymbol{\mu}_{k,i}^*), \tag{35}$$

(2) The vehicles (followers) then achieve a Nash equilibrium after:

$$U_i((s_{i,k}^*, x_{i,k}^*), (\boldsymbol{s}_{i,-k}^*, \boldsymbol{x}_{i,-k}^*)) \geq U_i((s_{i,k}, x_{i,k}), (\boldsymbol{s}_{i,-k}^*, \boldsymbol{x}_{i,-k}^*)),\ \forall s_{i,k} \in \boldsymbol{s}. \tag{36}$$

Theorem 1. *According to the Lemma 1, there exists a Nash equilibrium between the vehicles, if the game is concave.*

Remark 1. The Stackelberg equilibrium is a hierarchal equilibrium, if the leaders achieve a unique Nash equilibrium, the followers will also achieve a unique Nash equilibrium, resulting in a unique Stackelberg Equilibrium.[35]

According to the Theorem 1, the game between the vehicles is a strictly concave multi-player game, which has a NE.[36] In conclusion, there exists a Stackelberg equilibrium.

5.3.2 | Distributed algorithm

We propose a distributed algorithm for the offloading scheme. In this scheme, following the announced strategies of the VEC servers, each vehicle determines its offloading target servers and the number of offloading data by solving the optimization problem of the vehicle. Based on the response of vehicles, each VEC server adjusts its price. Then, the vehicles make their response to the strategy changes of VEC servers. The strategies of VEC servers and vehicles are updated iteratively until reaching equilibrium. The details of the proposed distributed algorithm are illustrated in Algorithm 1, and the flowchart of the proposed work is also given below as shown in Figure 2.

Algorithm 1 Distributed algorithm

Network information: the number of the vehicles N; the number of the VEC servers M; the vehicles with computation tasks $D_i, i \in \mathcal{N}$; the computation resource of vehicles $f_{loc,i}$; the computation resource of VEC servers F_k ;
Initialization:
Set selection decision $\boldsymbol{x}^{(0)}$, bit of offloading task $\boldsymbol{L}^{(0)}$, the price of VEC servers $\boldsymbol{\mu}^{(0)}$, the Lagrangian multipliers $\boldsymbol{\alpha}$, $\boldsymbol{\varphi}$, λ;
Set iteration $t_1 = 1, t_2 = 1$;
repeat
  repeat
    (1) Based on $\mu^{(t_1-1)}$, each vehicle obtains the selection decision and bits of offloading task;
    (2) Update Lagrangian multipliers $\alpha_i(t_2+1)$, $\varphi_k(t_2+1)$, $\lambda_i(t_2+1)$ based on (32);
    (3) $t_2 = t_2 + 1$;
  until Convergence;
  (4) Each VEC server calculates $\mu^{(t_1)}$ based on (33) or (34);
  (5) $t_1 = t_1 + 1$;
until Equilibrium;

Specifically, the steps of working of the proposed methodology are presented in the following:

- In the first step, construct the simulation environment, that is, a crossroad with multi-vehicle and multi-server. Initialize the offloading strategies and price strategy and the Lagrange multipliers.
- In the second step, as the followers of the game, vehicles achieve the offloading strategies $\boldsymbol{L}$ and $\boldsymbol{x}$ according to solving the P2, then vehicles broadcast their offloading strategies.
- In the third step, as the leaders of the game, VEC servers update the price strategy $\boldsymbol{\mu}$ based on the updated data of $\boldsymbol{L}$ and $\boldsymbol{x}$ achieved by the second step.
- Repeat the second and the third steps until reaching equilibrium.
- Finally, output offloading strategies and price strategy $\boldsymbol{L}, \boldsymbol{x}, \boldsymbol{\mu}$.

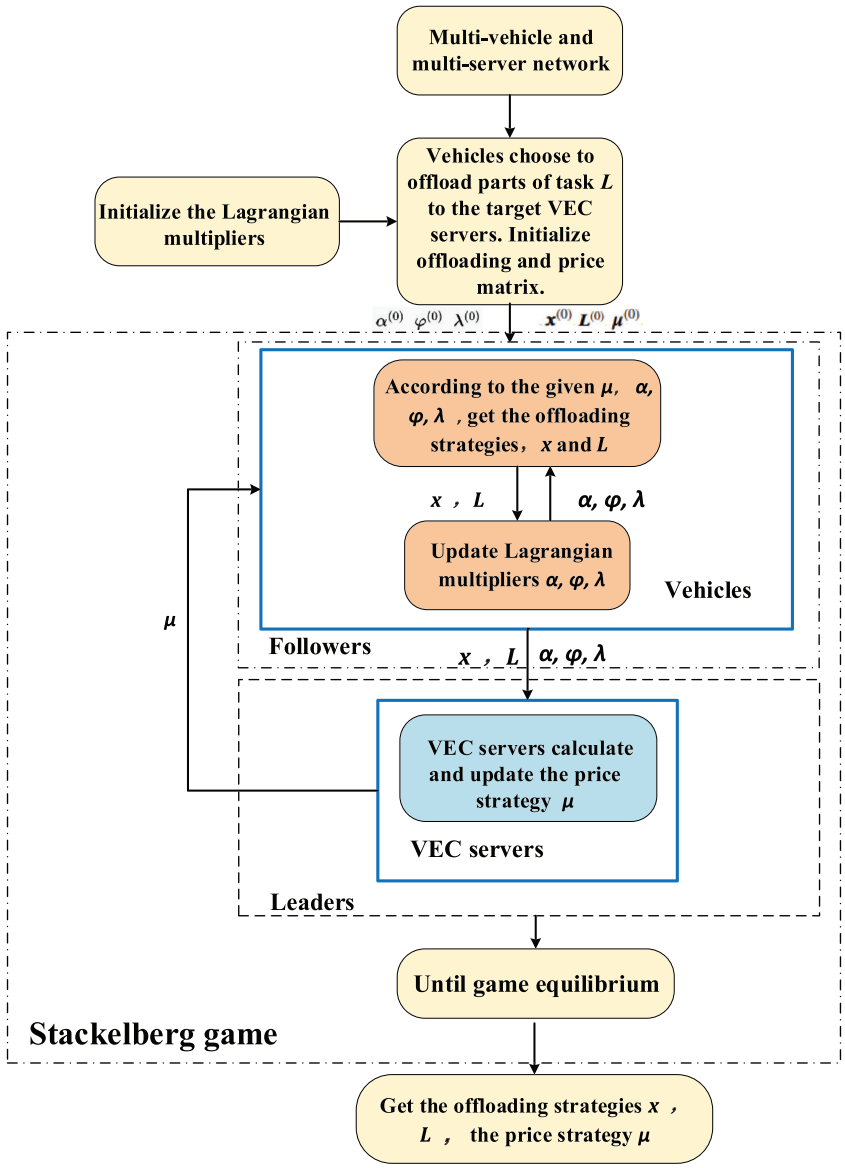

FIGURE 2 The flowchart of the proposed scheme

5.4 | Computational complexity

To facilitate the analysis of the computational complexity of Algorithm 1, we set the number of iterations for the outer loop and inner loop to be N_1 and N_2, respectively. There are N vehicles and M VEC servers in the considered network. The computational complexity of our proposed scheme is with polynomial time. Specifically, in each inner loop iteration, based on the judgment criterion that whether the Lagrangian multipliers converge, the VEC servers decide to perform price iteration or not. When the inner loop converges, the outer loop will start. $\mathcal{O}(N \times M+1)$ denotes the computational complexity related to the number of decision variables about VEC servers in the outer loop. $\mathcal{O}(2NM+N+1)$ denotes the computational complexity related to the number of decision variables about vehicles and linear constraints in the inner loop. Therefore, given the iteration number of both outer loop N_1 and inner loop N_2, the computational complexity of our the distributed algorithm can be expressed as $\mathcal{O}(N_1[(N \times M+1)+(2NM+N+1)N_2])$.

6 | SIMULATION RESULTS

In this section, we evaluate the performance of the proposed scheme. Simulation is implemented in MATLAB. First, we simulate a vehicular communication scenario at an urban crossroad where some RSUs equipped with the VEC servers are deployed following a uniform distribution. The Rayleigh fading channel and the path loss are simulated through MATLAB. The initial locations of vehicles are randomly generated at the crossroad. We take a slot-level simulation. In each slot, the vehicles make the offloading strategies. According to the generated simulation environment, the proposed algorithms are executed.

We assume a crossroad with a range of 200×200 m, where vehicles follow a random distribution. During simulation, we consider a vehicular edge network located at a crossroad. There are five vehicles and three RSUs equipped with the VEC servers distributed uniformly. Each RSU is equipped with a VEC server. Each vehicle occupies one orthogonal channel. The local CPU frequency F_k for each VEC server k is set to be 10 GHz. Each vehicle has a computation task. The computation resource of each vehicle is 1 GHz. The communication parameters used in our simulations are summarized in Table 1. To evaluate the performance of the proposed scheme, the convergence and system performance are presented in the following sections on the basis of the initial values of dual variables $\boldsymbol{\alpha} = 20, \boldsymbol{\varphi} = 50, \boldsymbol{\lambda} = 40$ and the settings of Table 1, respectively. We compare our proposed offloading scheme to the following offloading schemes:

(1) *Local execution*: the vehicles compute their tasks in local totally.
(2) *Fixed selection*: each vehicle chooses the nearest VEC server as its target server.
(3) *Complete offloading with fixed prices*: each vehicle offloads completely with fixed prices and each VEC server broadcasts the same price for each vehicle.

6.1 | Convergence performance

In this section, we evaluate the convergence of the proposed distributed algorithm. The number of vehicles N, the number of VEC servers M, edge computation resource F_k and CPU cycles C_i are set to 5, 3, 1 GHz and 500 cycles/bit, respectively. Firstly, we evaluate the convergence of the Lagrangian multipliers α, φ, λ in the inner loop of the distributed algorithm, and the results are shown in Figure 3. From Figure 3, it can be observed that all the Lagrangian multipliers converge within about four iterations. The convergence of the proposed distributed algorithm is shown in Figure 4 in terms of the average utility of vehicles. From Figure 4, we can see that the average utility of the vehicles converges fast within about 10 iterations. Based on the above results, we can conclude that the proposed distributed algorithm is efficient.

TABLE 1 Simulation parameters

Parameter	Symbol	Value
The data of task	R_i	500 KB
Vehicle transmission power	p_i	30 dBm
Bandwidth	B	10 MHz
Noise spectral density	N_0	−174 dBm/Hz
Path loss exponent	δ	3
The number of CPU cycles for computing 1-bit	C_i	500 cycles/bit
Local vehicle resource	$f_{loc,\,i}$	1 GHz
VEC server resource	F_k	10 GHz
Maximum latency	T_i^{max}	2 s
Maximum number of bits allowed to be processed	S_k^{max}	1500 KB

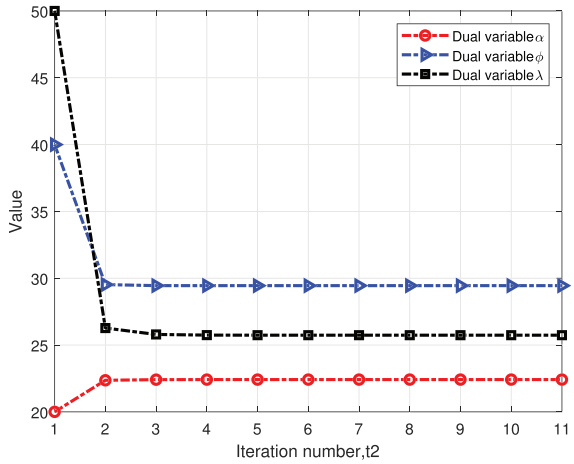

FIGURE 3 Convergence of Lagrangian multipliers

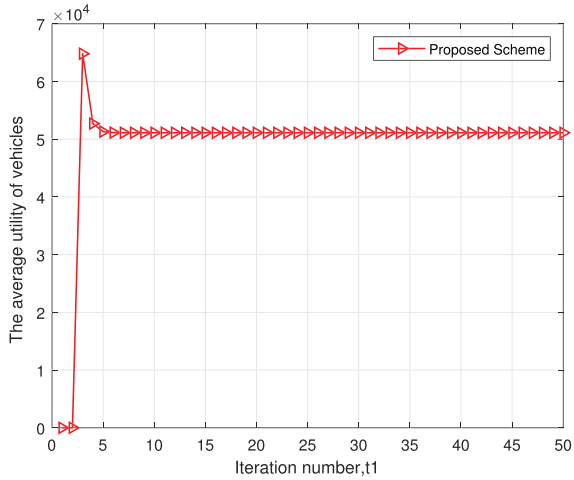

FIGURE 4 Convergence of the distributed algorithm

6.2 | System utility

In this section, we demonstrate the performance of the proposed scheme (denoted as *proposed scheme*) and also compare it with other three schemes: *local execution*, *fixed selection*, and *complete offloading with fixed prices*.

To illustrate the effect of the local computation resources on the average utility of vehicles for different offloading schemes, we compare the proposed scheme with the *local execution* and *fixed selection*, and the results are shown in Figure 5. The number of vehicles N, the number of VEC servers M, local computation resource $f_{loc,\,i}$, edge computation resource F_k, and CPU cycles C_i are set to 5, 3, 1 GHz, 5 GHz, and 500 cycles/bit, respectively. From Figure 5, we can see that if the task is executed in local totally, the utility of the vehicle is set to be zero according to the utility function. By comparing with *local execution* and *fixed selection* schemes, it can be observed that the average utility of vehicles for both the *proposed scheme* and *fixed selection* schemes decrease along with the increase of the local computation resources of vehicle. The reason is that along with the increase of the local computing resources, vehicles prefer to execute tasks locally rather than spending money to buy the computation resource from the VEC server. In addition, the performance in terms of the average utility of vehicles for the proposed scheme is better than the other two schemes, which also shows the superiority of our proposed scheme.

In order to observe the impacts of edge computation resources on the offloading schemes, Figure 6 evaluates the performance of the utility of vehicles with different edge computation resources under different schemes. The *local execution*, *fixed selection*, and *complete offloading with fixed prices* schemes are used as the baselines, where the prices

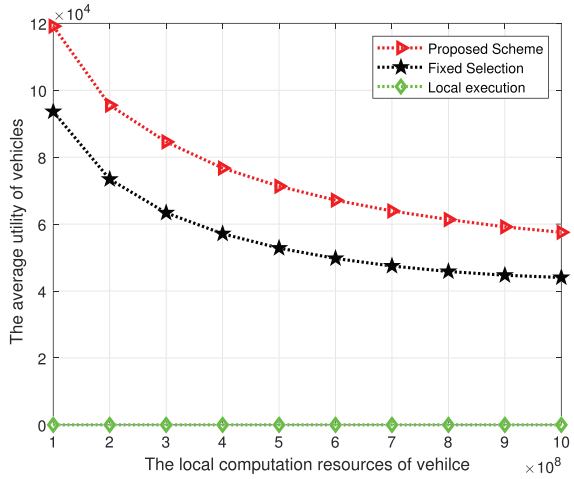

FIGURE 5 Comparison of the average utility of vehicles with different local computation resources under different schemes

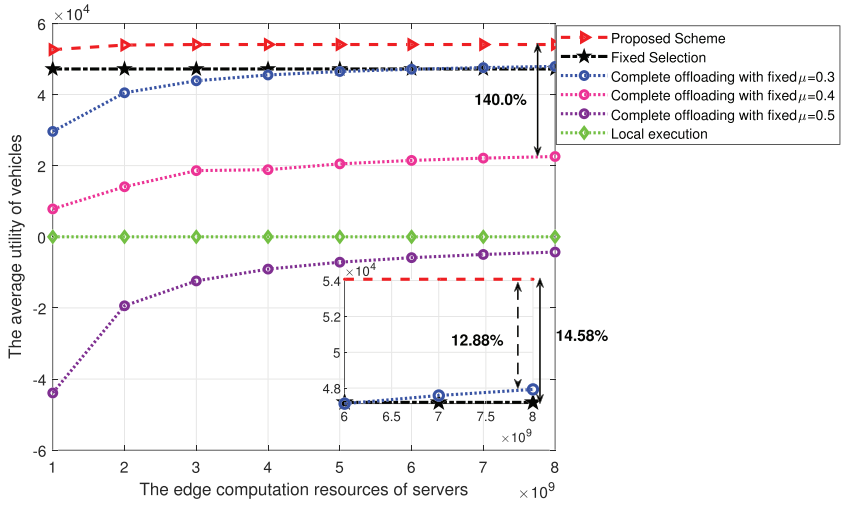

FIGURE 6 Comparison of the average utility of vehicles with different VEC computation resources under different schemes

$\mu_{k,\,i}$ for *complete offloading with fixed prices* scheme is set to be 0.3, 0.4, and 0.5, respectively. Besides, the number of vehicles N, the number of VEC servers M, local computation resource $f_{loc,\,i}$ and CPU cycles C_i are set to 5, 3, 1 GHz, and 500 cycles/bit, respectively. It can be seen that the average utility of vehicles of all the schemes except the *local execution* scheme increases firstly and then tends to stable along with the increase of the edge computation resources of servers. The proposed scheme achieves the best performance by comparing it with the other five schemes. The offloading scheme with the fixed price $\mu_{k,i} = 0.5$ has the lowest utility. The reason is that in the proposed scheme, the price set by the VEC server is adjusted dynamically through considering the supply and demand and the vehicles can make the best selection of the VEC server according to the actual network situation. To be clearer, we enlarge the details of the Figure 6 in a sub-figure and add a table containing the performance data achieved by the proposed scheme and the comparison schemes. Details can be seen in Table 2.

Figure 7 illustrates the comparison of the average utility of the vehicles with respect to the numbers of the vehicles under different offloading schemes, with $M = 3$, $f_{loc,\,i} = 1$ GHz, $F_k = 5$ GHz, $C_i = 500$ cycles/bit. From Figure 7, we can observe that the average utility of the vehicles achieved by our proposed scheme is better than the other five schemes. In addition, it can be seen that the average utility of vehicles for all the schemes except the *Local execution* scheme decreases as the number of vehicles increases. This is because the computation resources of the VEC servers are limited, and the computation resources allocated to each associated vehicle become fewer as the number of vehicles increases.

The impact of the number of vehicles on the average utility of the vehicles under different offloading schemes is simulated, and the results are shown in Figure 7. From Figure 7, it can be seen that along with the increase of the number of vehicles, the computation resources allocated to each vehicle by the VEC servers are reduced, which causes the decrease of the utility of vehicles. Similarly, in a certain range, if the density of vehicles increases, the average number of vehicles increases. There will be multiple vehicles accessing the RSUs at the same time for task offloading and computing. As a result, the computation resources of VEC servers allocated to each vehicle are reduced. The average utility of vehicles decreases as the VEC computation resource decreases. It can be seen that the performance of our proposed scheme will decrease when the vehicle density increases.

We also evaluate the effect of CPU cycles required by computing per bit on the average utility of the vehicles under different offloading schemes with $N = 5$, $M = 3$, $f_{loc,\,i} = 1$ GHz, $F_k = 5$ GHz, and the results are shown in Figure 8. From Figure 8, we can observe that the average utility of vehicles of the *complete offloading with fixed prices* schemes fall down as the CPU cycles per bit increase. This is because the VEC charges more fees from the vehicles as the CPU cycles per bit increase if the price is fixed. Except for the *local execution* and *complete offloading with fixed prices* schemes, the performance of our proposed scheme and the *fixed selection* scheme increase along with the increase of the CPU cycles required by computing per bit. The reason is that the VEC servers can adjust the price to optimize the utility of vehicles. In addition, the results in Figure 8 also illustrate that the proposed scheme gets the optimal performance by comparing with other schemes.

TABLE 2 Comparison of the average utility of vehicles with different VEC computation resources under different schemes

	SCHEME					
The VEC computation resource (Hz)	**PS**	**FS**	**CO 0.3**	**CO 0.4**	**CO 0.5**	**LO**
1×10^9	$\boldsymbol{5.26 \times 10^4}$	4.70×10^4	3.01×10^4	0.78×10^4	-4.39×10^4	0
2×10^9	$\boldsymbol{5.39 \times 10^4}$	4.71×10^4	4.05×10^4	1.41×10^4	-1.94×10^4	0
3×10^9	$\boldsymbol{5.40 \times 10^4}$	4.72×10^4	4.40×10^4	1.87×10^4	-1.24×10^4	0
4×10^9	$\boldsymbol{5.41 \times 10^4}$	4.72×10^4	4.55×10^4	1.89×10^4	-0.91×10^4	0
5×10^9	$\boldsymbol{5.41 \times 10^4}$	4.72×10^4	4.65×10^4	2.05×10^4	-0.71×10^4	0
6×10^9	$\boldsymbol{5.41 \times 10^4}$	4.72×10^4	4.71×10^4	2.15×10^4	-0.59×10^4	0
7×10^9	$\boldsymbol{5.41 \times 10^4}$	4.72×10^4	4.76×10^4	2.21×10^4	-0.50×10^4	0
8×10^9	$\boldsymbol{5.41 \times 10^4}$	4.72×10^4	4.79×10^4	2.26×10^4	-0.43×10^4	0

Note: **PS**, proposed scheme; **FS**, fixed selection; **CO 0.3**, complete offloading with $\mu = 0.3$; **CO 0.4**, complete offloading with $\mu = 0.4$; **CO 0.5**, complete offloading with $\mu = 0.5$; **LO**, local execution.

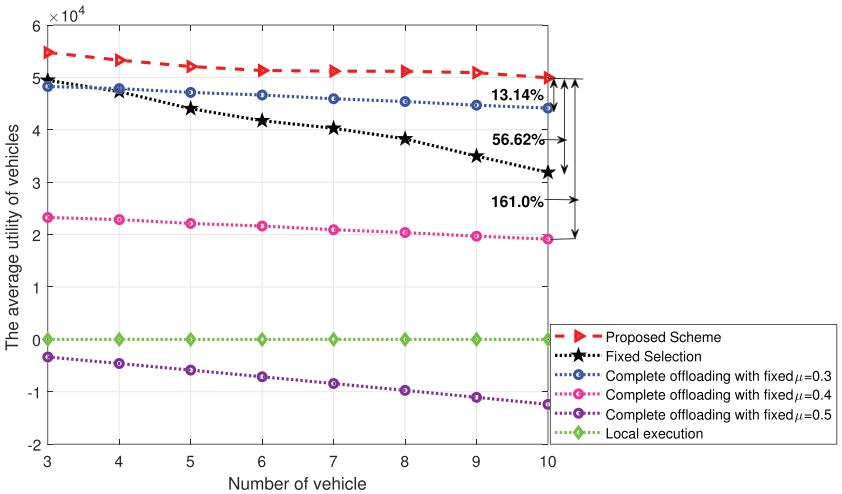

FIGURE 7 Comparison of the average utility of vehicles with different number of vehicles under different schemes

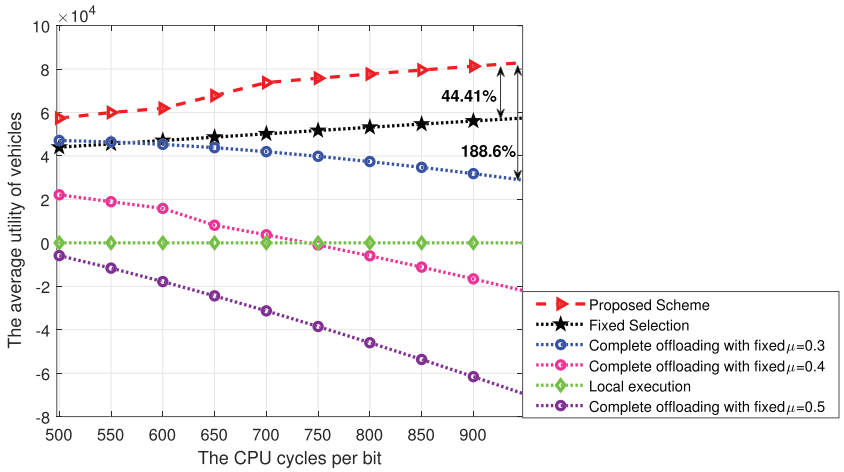

FIGURE 8 Comparison of the average utility of vehicles with different CPU cycles under different schemes

According to the above analysis and discussion about the simulation results, we can see that our proposed scheme can achieve good convergence. Under the different network conditions and different parameter settings of the vehicle number, computation resource, as well as CPU resources, the proposed scheme still maintains superior performance in terms of the average utility of vehicles. The effects of key parameters such as offloading strategies and the price strategy of VEC servers on the utility of vehicles are also evaluated through simulation. When the local computation resources increase, according to the average utility of vehicles, the value of $T_{loc,\,i} - T_{i,\,k}$ will decrease because the vehicle prefers to compute more data locally, and the price of the target VEC server will increase although the data offloaded to the VEC server is reduced. The reason why the average utility of vehicles decreases is that the computation resources allocated equally to each vehicle are reduced when the number of vehicles increases. The average utility of vehicles increases when the VEC servers' computation resource decreases because of the reduction of the task execution latency. The decrease in latency results from the increase of computation resources allocated to each vehicle. The increase of CPU cycles means that the computation resources required to complete the task increase. In order to obtain lower latency, the vehicle offloads more data to the VEC server. And the price of the VEC server will decrease with the increase of offloading data.

In our proposed scheme, the vehicles can select the optimal offloading strategies according to the price set by the VEC servers. Meanwhile, the VEC servers also can adjust their price based on the offloading strategies of all vehicles. Through the interaction between the vehicles and the VEC servers, the vehicles can balance the trade-off between the latency incurred by finishing the computation task and the cost of purchasing the computation resource.

7 | CONCLUSIONS

In this paper, we have investigated the task offloading problem in VEC networks and also have proposed a more flexible offloading scheme based on the Stackelberg game. The task offloading problem has been formulated as a multi-leader and multi-follower Stackelberg game to maximize the utilities of both vehicles and VEC servers under the delay constraints of the computation tasks. To solve the above problems, the Lagrange dual method has been adopted to solve the optimization problem. Meanwhile, we have proposed a distributed algorithm to obtain the optimal strategies including offloading strategies and price strategy. Simulation results validated our theoretical analysis and demonstrated that the proposed scheme can always achieve the best performance in terms of the average utility of vehicles compared to other existing offloading schemes. In the proposed scheme, the vehicles can select optimally more flexible offloading strategies based on the prices set by the VEC servers to balance the trade-off between the latency and the cost. For future work, we will investigate collaborative cache allocation and computation offloading in VEC network to maximize the resource utilization and reduce significantly data exchange, where the more practical mobility model of vehicles will be taken into account to ensure efficient task offloading, for example, constant velocity model and acceleration model.

ACKNOWLEDGMENTS

The work presented in this paper was supported in part by the National Natural Science Foundation of China (No. 61801278, 61972237, and 61901247), Shandong Provincial scientific research programs in colleges and universities (J18KA310) and the Key Laboratory of Cognitive Radio and Information Processing, Ministry of Education (Guilin University of Electronic Technology) (CRKL190205), and Shandong Provincial Natural Science Foundation for Young Scholars of China (grant no. ZR2020QF001).

DATA AVAILABILITY STATEMENT

Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

ORCID

Jie Tian https://orcid.org/0000-0002-1791-2079

REFERENCES

1. Ji B, Zhang X, Mumtaz S, et al. Survey on the Internet of Vehicles: Network Architectures and Applications. *IEEE Commun Standards Mag*. 2020;4(1):34-41.
2. Heidari A, Jabraeil Jamali MA, Jafari Navimipour N, Akbarpour S. Internet of Things offloading: Ongoing issues, opportunities, and future challenges. *Int J Commun Syst*. 2020;33(14):e4474.
3. Ilyas MU, Ahmad M, Saleem S. Internet-of-Things-Infrastructure-as-a-Service: The democratization of access to public Internet-of-Things Infrastructure. *Int J Commun Syst*. 2020;33:e4562.
4. Bhatia J, Dave R, Bhayani H, Tanwar S, Nayyar A. SDN-based real-time urban traffic analysis in VANET environment. *Comput Commun*. 2020;149:162-175.
5. Bian J, Wang CX, Gao X, You X, Zhang M. A general 3D non-stationary wireless channel model for 5G and beyond. *IEEE Trans Wireless Commun*. 2021;1-1.
6. Mao Y, You C, Zhang J, Huang K, Letaief KB. A Survey on Mobile Edge Computing: The Communication Perspective. *IEEE Commun Surv Tutor*. 2017;19(4):2322-2358.
7. Rodrigues TG, Suto K, Nishiyama H, Kato N. Hybrid Method for Minimizing Service Delay in Edge Cloud Computing Through VM Migration and Transmission Power Control. *IEEE Trans Comput*. 2017;66(5):810-819.
8. Tian J, Zhang H, Wu D, Yuan D. QoS-Constrained Medium Access Probability Optimization in Wireless Interference-Limited Networks. *IEEE Trans Commun*. 2018;66(3):1064-1077.
9. Pande SK, Panda SK, Das S, et al. A smart cloud service management algorithm for vehicular clouds. *IEEE Trans Intell Transp Syst*. 2020;1-12.
10. Zhang J, Letaief KB. Mobile Edge Intelligence and Computing for the Internet of Vehicles. *Proc IEEE*. 2020;108(2):246-261.
11. Shi W, Cao J, Zhang Q, Li Y, Xu L. Edge Computing: Vision and Challenges. *IEEE Internet Things J*. 2016;3(5):637-646.
12. Abbas N, Zhang Y, Taherkordi A, Skeie T. Mobile Edge Computing: A Survey. *IEEE Internet Things J*. 2018;5(1):450-465.
13. Ali SSD, Zhao HP, Kim H. Mobile edge computing: A promising paradigm for future communication systems. TENCON 2018-2018 IEEE Region 10 Conference. 2018:1183-1187.
14. Deng M, Tian H, Lyu X. Adaptive sequential offloading game for multi-cell mobile edge computing. 2016 23rd international conference on telecommunications (ICT). 2016; 1–5
15. Zhang K, Mao Y, Leng S, He Y, Zhang Y. Mobile-Edge Computing for Vehicular Networks: A Promising Network Paradigm with Predictive Off-Loading. *IEEE Veh Technol Mag*. 2017;12(2):36-44.
16. Zhang K, Mao Y, Leng S, Maharjan S, Zhang Y. Optimal delay constrained offloading for vehicular edge computing networks. 2017 IEEE International Conference on Communications (ICC). 2017:1-6.
17. Wu S, Xia W, Cui W, Chao Q, Lan Z, Yan F, Shen L. An Efficient Offloading Algorithm Based on Support Vector Machine for Mobile Edge Computing in Vehicular Networks. 2018 10th International Conference on Wireless Communications and Signal Processing (WCSP). 2018:1-6.
18. Du J, Yu FR, Chu X, Feng J, Lu G. Computation Offloading and Resource Allocation in Vehicular Networks Based on Dual-Side Cost Minimization. *IEEE Trans Veh Technol*. 2019;68(2):1079-1092.
19. Nguyen V, Khanh TT, Tran NH, Huh E, Hong CS. Joint offloading and IEEE 802.11 p-based contention control in vehicular edge computing. *IEEE Wireless Commun Lett*. 2020;9:1014-1018.
20. You C, Huang K, Chae H, Kim B. Energy-Efficient Resource Allocation for Mobile-Edge Computation Offloading. *IEEE Trans Wireless Commun*. 2017;16(3):1397-1411.
21. Dai Y, Xu D, Maharjan S, Zhang Y. Joint Load Balancing and Offloading in Vehicular Edge Computing and Networks. *IEEE Internet Things J*. 2019;6(3):4377-4387.
22. Wang Y, Lang P, Tian D, et al. A Game-Based Computation Offloading Method in Vehicular Multiaccess Edge Computing Networks. *IEEE Internet Things J*. 2020;7(6):4987-4996.

23. Liu Y, Yu H, Xie S, Zhang Y. Deep Reinforcement Learning for Offloading and Resource Allocation in Vehicle Edge Computing and Networks. *IEEE Trans Veh Technol*. 2019;68(11):11158-11168.
24. Zhan W, Luo C, Wang J, et al. Deep-Reinforcement-Learning-Based Offloading Scheduling for Vehicular Edge Computing. *IEEE Internet Things J*. 2020;7(6):5449-5465.
25. Tang D, Zhang X & Tao X Delay-Optimal Temporal-Spatial Computation Offloading Schemes for Vehicular Edge Computing Systems. 2019 IEEE Wireless Communications and Networking Conference (WCNC). 2019:1-6.
26. Zhang K, Mao Y, Leng S, Vinel A & Zhang Y Delay constrained offloading for Mobile Edge Computing in cloud-enabled vehicular networks. 2016 8th International Workshop on Resilient Networks Design and Modeling (RNDM). 2016:288-294.
27. Yang B, Wu D, Wang H, Gao Y, Wang R. Two-Layer Stackelberg Game-Based Offloading Strategy for Mobile Edge Computing Enhanced FiWi Access Networks. *IEEE Trans Green Commun Netw*. 2021;5(1):457-470.
28. Dab B, Aitsaadi N & Langar R Joint Optimization of Offloading and Resource Allocation Scheme for Mobile Edge Computing. 2019 IEEE Wireless Communications and Networking Conference (WCNC). 2019:1-7.
29. Xu X, Chen Y, Zhang X, Liu Q, Liu X, Qi L. A blockchain-based computation offloading method for edge computing in 5G networks. *Softw Practice Exp*. 2019.
30. Zhang X, Peng M, Yan S, Sun Y. Deep-Reinforcement-Learning-Based Mode Selection and Resource Allocation for Cellular V2X Communications. *IEEE Internet Things J*. 2019;1-1.
31. Zhang J, Guo H, Liu J, Zhang Y. Task Offloading in Vehicular Edge Computing Networks: A Load-Balancing Solution. *IEEE Trans Veh Technol*. 2020;69(2):2092-2104.
32. Yu W, Lui R. Dual methods for nonconvex spectrum optimization of multicarrier systems. *IEEE Trans Commun*. 2006;54(7):1310-1322.
33. Boyd S, Boyd SP, Vandenberghe L. *Convex optimization*. Cambridge university press; 2004.
34. Zhang J, Zhang J, Zhou Y, Ji H, Sun J, Al-Dhahir N. Energy and Spectral Efficiency Tradeoff via Rate Splitting and Common Beamforming Coordination in Multicell Networks. *IEEE Trans Commun*. 2020;68(12):7719-7731.
35. Sawyer N, Smith DB. Flexible Resource Allocation in Device-to-Device Communications Using Stackelberg Game Theory. *IEEE Trans Commun*. 2019;67(1):653-667.
36. Liu B. Stackelberg-Nash equilibrium for multilevel programming with multiple followers using genetic algorithms. *Comput Math Appl*. 1998;36(7):79-89.

How to cite this article: Liu S, Tian J, Deng X, Zhi Y, Bian J. Stackelberg game-based task offloading in vehicular edge computing networks. *Int J Commun Syst*. 2021;e4947. doi:10.1002/dac.4947