EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Energy-efficient offloading decision-making for mobile edge computing in vehicular networks

Xiaoge Huang[1*], Ke Xu[1], Chenbin Lai[1], Qianbin Chen[1] and Jie Zhang[1,2]

Abstract

Driven by the explosion transmission and computation requirement in 5G vehicular networks, mobile edge computing (MEC) attracts more attention than centralized cloud computing. The advantage of MEC is to provide a large amount of computation and storage resources to the edge of networks so as to offload computation-intensive and delay-sensitive applications from vehicle terminals. However, according to the mobility of vehicle terminals and the time varying traffic load, the optimal task offloading decisions is crucial. In this paper, we consider the uplink transmission from vehicles to road side units in the vehicular network. A dynamic task offloading decision for flexible subtasks is proposed to minimize the utility, which includes energy consumption and packet drop rate. Furthermore, a computation resource allocation scheme is introduced to allocate the computation resources of MEC server due to the differences in the computation intensity and the transmission queue of each vehicle. Consequently, a Lyapunov-based dynamic offloading decision algorithm is proposed, which combines the dynamic task offloading decision and computation resource allocation, to minimize the utility function while ensuring the stability of the queue. Finally, simulation results demonstrate that the proposed algorithm could achieve a significant improvement in the utility of vehicular networks compared with comparison algorithms.

Keywords: Mobile edge computing, Offloading, Energy-efficient, Lyapunov optimization

1 Introduction

With the rapid development of internet of vehicles (IoV), vehicular communications have led to the emergence of intelligent transportation systems. Vehicles become as intelligent as smart mobile devices which can support various applications and services, such as autonomous driving, augmented reality (AR), virtual reality (VR), and online gaming [1, 2]. Particularly, most of these applications are delay-sensitive and computation-intensive, which result in significant challenges of the computational capability and the battery capability of vehicles [3, 4]. Moreover, offloading computation tasks and data traffic to a remote centralized cloud is considered as an effective approach for resolving the above challenges by employing abundant computation resources, which could relieve the pressure of the computational capability of vehicles [5]. However, the long transmission distance from vehicles to the centralized cloud results in high latency execution time and challenges for the backhaul bandwidth. In future vehicular networks, there will be an ever-increasing number of high-traffic applications, and multiple tasks from smart vehicles need to be processed simultaneously. The shortage of local computation resource will bring more severe challenges.

Currently, mobile edge computing (MEC)-enabled vehicular network is considered as another potential approach, which could provide powerful computation resources for the computation-intensive applications. Compared with other edge computing, e.g., mobile cloud computing (MCC), fog computing (FC), and cloudlet [6], MEC technology is favored by academia and industry. With the development of IoV, benefit from the MEC, effective transmission for the network with heavy traffic load, high bandwidth and low latency transmission could be guaranteed. Generally, MEC servers are cooperated

*Correspondence: Huangxg@cqupt.edu.cn
[1] School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongwen Road, Chongqing, China
Full list of author information is available at the end of the article

with cellular base stations to offer services to vehicles in the edge of the radio access network.

Additionally, in the vehicular ad hoc network (VANET), vehicles use DSRC (dedicated short-range communications), Wi-Fi, or cellular networks to access the infrastructure (e.g., road side unit (RSU)) at roadside. The vehicle can communicate to the near RSU through vehicle-to-infrastructure (V2I) communications [7]. The MEC server are deployed close to the RSU, which can connect to RSU to provide abundant computation resources for tasks from vehicles, and significantly shorten the execution time and reduce the local energy consumption of vehicles.

Although vehicles may benefit from MEC, it is not easy to make an appropriate offloading decision due to the mobility of vehicles. Notably, not all tasks from vehicles can be offloaded to the MEC server. Based on their individual attributes, tasks are classified into the local subtasks, which should access local components and can only be processed in vehicles, e.g., sensors, cameras, and user interfaces [8]. Besides, there are flexible subtasks, which can be processed either in the vehicle or in the MEC server. For these flexible subtasks, vehicles could make a decision whether offloading them to the MEC server or executing locally according to the network utility.

The large-scale mobile applications are typically served with the assistance of on-board units (OBUs), result in vast energy consumption of CPU, which is the top concern for users [9, 10]. To guarantee low-energy consumption and latency transmission, vehicles with different tasks need to find an appropriate offloading decision criteria to achieve a better network performance. Furthermore, dynamic topology changes caused by the mobility of vehicles and packet drop make offloading decisions more complex. Therefore, in this paper, we propose a dynamic task offloading scheme to jointly minimize the packet drop rate and energy consumption based on various tasks in the vehicular network by Lyapunov optimization.

1.1 Related works

MEC-enabled offloading have been proposed as a promising approach to solve the task offloading problems in recent research literatures [11–15]. In [11], authors proposed a joint task offloading and resource allocation algorithm in vehicular networks to minimize the cost of dual-sides, which includes smart vehicle and the MEC server. In [12], a distributed computation offloading scheme was proposed to optimize offloading decisions, while guaranteeing the quality of experience (QoE) of vehicles and maximizing the utility of the MEC server, where the utility consists of the energy consumption, delay, and computation resources. In [13], authors proposed a support vector machine-based offloading algorithm to reduce computation complexity, which ensures low latency in the offloading process for high-speed vehicles. A cooperative scheme for parallel computing and transmission was proposed in [14] to reduce the latency of VR applications, where the parallel computing in the MEC server was applied for subtasks. In [15], the authors focused on the resource allocation scheme for multi-user MEC offloading systems by orthogonal frequency-division multiple access technologies to minimize energy consumption while reducing the computation complexity.

Vehicular fog computing and other edge computing have also been widely investigated for task offloading in literatures [16–19]. The authors proposed a task scheduling scheme based on the computing capabilities of different vehicles [16], which could improve the utilization of computing resources, while ensuring the low-latency transmission and system stability. A hierarchical cloud-based vehicular edge computing (VEC) offloading framework was proposed in [17], where the backup computing server was deployed close to the VEC server to provide computing resources. Furthermore, an optimal multilevel offloading scheme was designed by employing the Stackelberg game, which introduced an iterative distribution algorithm to maximize the system utility of vehicles. Resource allocation problem of multi-user and multi-server VEC system was investigated in [18], a offloading scheme was proposed to reasonably allocate resources for on-board applications to balance load and offloading. Guo et al. proposed a constrained randomized offloading scheme and a centralized heuristic greedy offloading scheme to improve resource utilization [19]. In addition, collaborative task offloading among the remote cloud, the edge servers and vehicles can be achieved via vehicle-to-vehicle (V2V) and V2I communication by making offloading decision.

Moreover, V2V communication is also considered as an alternative way for task offloading. In [20], a bus-based content offloading algorithm was proposed to maximize the overall amount of offloading tasks while ensuring fairness between vehicles. The number of buses with offloading requirement could be predicted by their positions and the corresponding transmission rates. In [21], a software-defined network inside the mobile edge computing (SDNi-MEC) architecture was proposed, and each vehicle could offload tasks by either the SDN-MEC server or the V2V link according to the transmission cost.

In [22], vehicles were considered as a cloudlet, which could execute tasks for mobile devices. In order to ensure the reliability of the communication link, the task were divided into multiple parts, each vehicle acted as a relay to execute a part of the task. In [23], the author proposed a task offloading scheme based on machine learning, and vehicles could get feedback from their neighboring service vehicles, to effectively share the computing and storage resources from service vehicles. Similar to

[23, 24] proposed an offloading decision scheme based on knowledge driven, which optimized offloading decisions by the deep reinforcement learning to minimize the transmission delay.

In addition, in vehicular heterogeneous networks, which was composed of VANET and the cellular network, authors focused on the optimal offloading strategy for the online video traffic [25]. In [26], authors proposed a contact duration aware optimal offloading scheme to optimize resource management and data offloading of vehicles through the cellular network and vehicular opportunistic communications. Furthermore, game theory was used to opportunistically offload the vehicle traffic through the Wi-Fi network in [27]. Most of the previous works focused on how to make optimal offloading decisions to increase the utilization of the computation resource of service nodes, while reducing the network delay or energy consumption of the task execution. However, the cost of energy consumption of the on-board unit and the packet drop are rarely considered.

1.2 Contributions

Parts of research works considered task offloading schemes for a specific time, meanwhile, the low task granularity assumption, such as bit, is also unrealistic. In this paper, we consider the energy-efficient offloading decision-making for mobile edge computing in vehicular networks to minimize network utility, which includes packet drop rate and energy consumption. A Lyapunov-based dynamic task offloading algorithm is proposed to minimize the total network utility under the optimal offloading decisions by jointly considering energy consumption and packet drop rate. The main contributions are listed as follows:

- Firstly, we consider the uplink transmission from vehicles to road side units in the vehicular network. According to the properties of subtasks, the subtasks are classified into local subtasks and flexible subtasks. The utility of the vehicular network is composed by the weighted sum of energy consumption and packet drop rate. To minimize the network utility, we propose a dynamic task offloading model for flexible subtasks.
- Secondly, to simplify the optimization problem, we firstly optimize the computation resource allocation of MEC server due to the computation intensity and the transmission queue of each vehicle.
- Finally, a Lyapunov-based dynamic offloading decision algorithm is proposed, which combines the dynamic task offloading decision and computation resource allocation, to minimize the utility function while ensuring the stability of the queue.

The rest of this paper is organized as follows. Section 2 introduces the system model and presents the optimization problem formulation. In section 3, a computation resource allocation scheme and a Lyapunov-based dynamic offloading decision algorithm are introduced to solve the optimization problem. The simulation results are presented and discussed in section 4. Finally, section 5 draws the conclusion.

2 Theoretical method

2.1 System model

2.1.1 Scenario description

Consider N RSUs deployed on the roadside and a MEC server is connected to a RSU by the wired line. Let $\mathcal{N} = \{1, 2, ..., n, ..., N\}$ denote the set of RSUs. There are K_n vehicles within the coverage of RSU n, $\mathcal{K}_n = \{1, 2, ..., k_n, ..., K_n\}$ denote the set of vehicles within the coverage of RSU n, the average speed of vehicles is v. The network scenario is shown in Fig. 1. Assume that each vehicle has computation-intensive and delay-sensitive tasks, which could either be offloaded to the MEC server through the associated RSU, or executed locally. For the sake of simplicity, a task offloading period is divided to several time period t. Therefore, the network scenario is quasi-static, the positions of vehicles and wireless channels conditions are unchanged during each optimization iteration, and changing between optimization iterations. Notice that, the handover process between RSUs is not considered in this scenario.

Moreover, the task of vehicle k_n can be divided into several subtasks, each subtask includes $L_{k_n}(t)$ packets with the computation intensity Z_{k_n} (cycle/bit). According to the particular properties, the subtasks can be classified into the following two classes [8].

1) Local subtask: The subtask should be processed locally in the vehicle. It takes more time and energy to transmit relevant information to the MEC server than to process it locally, or the subtask must access local components (e.g., sensors, cameras, and user interfaces). Additionally, there is no transmission delay, and the energy consumption is from the computational energy of the vehicle.
2) Flexible subtask: The subtask can be processed either in the vehicle or in the MEC server. The offloading decision depends on the difference in transmission delay and energy consumption between the MEC offloading and the local execution.

Based on the above discussion, finding the optimal offloading decision of the task is equivalent to optimizing the offloading decisions of the flexible subtask base on energy consumption and packet drop rate. Particularly, if all vehicles decide to offload flexible subtasks to the MEC

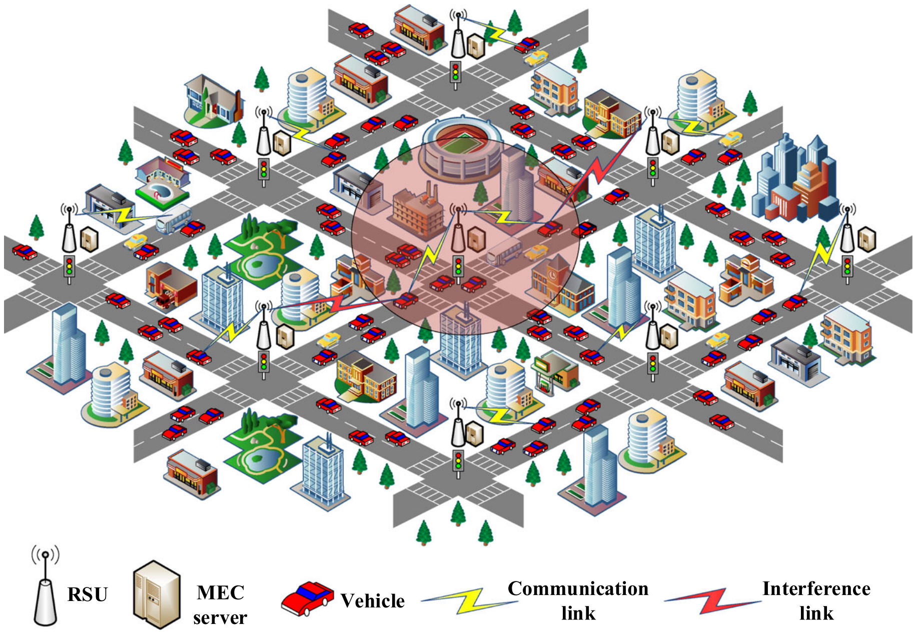

Fig. 1 Network scenario of V2I communications. Vehicles offload the task to the MEC server connected to the RSU via the V2I

server, the transmission delay and the packet drop rate will be increased simultaneously, resulting in low transmission quality of the vehicular network. The notations used in the paper are summarized in Table 1.

2.1.2 Communication and offloading decision model

Consider the uplink transmission from a vehicle to the RSU in the vehicular network, the vehicle could offload computation task to the MEC server via the associated RSU. Orthogonal frequency division multiplexing (OFDM)-based orthogonal channels are assigned to vehicles by RSUs [28]. The transmission power of vehicle k_n is denoted as $p^m_{k_n}$, and the received signal-to-interference-plus-noise ratio (SINR) of RSU n from vehicle k_n at t is given by

$$\mathrm{SINR}^n_{k_n}(t) = \frac{p^m_{k_n} H^n_{k_n}(t)}{I_{k_n}(t) + N_0 B} \tag{1}$$

where $H^n_{k_n}(t)$ is the channel gain between vehicle k_n and RSU n at t, $I_{k_n}(t)$ is the received interference power from other vehicles within the coverage of other RSUs to RSU n at t. N_0 is the noise power spectral density, and B is the channel bandwidth [29]. Therefore, for vehicle k_n, the uplink transmission rate can be expressed as

$$R^m_{k_n}(t) = B \log_2\left(1 + \mathrm{SINR}^n_{k_n}(t)\right)/S \tag{2}$$

where S is the size of a data packet.

Assume the task generated by vehicle k_n at t consists of two different categories subtasks, the number of the local subtasks is $N^l_{k_n}(t)$ and the flexible subtasks is $N^f_{k_n}(t)$.

The local subtasks should be executed on the local vehicle, whereas the flexible subtasks could be executed locally or offloaded to the MEC server depending on the offloading decision. Let $\alpha_{k_n}(t), \beta_{k_n}(t)$ denote the offloading decisions of vehicle k_n at t, $\alpha_{k_n}(t) \in [0, 1]$, $\beta_{k_n}(t) = 1 - \alpha_{k_n}(t)$. For instance, the number of the flexible subtasks is $N^f_{k_n}(t)$, $\alpha_{k_n}(t) = 0.6$, $\beta_{k_n}(t) = 0.4$ indicates $0.6N^f_{k_n}(t)$ flexible subtasks will be offloaded to the MEC server, the rest of $0.4N^f_{k_n}(t)$ flexible subtasks will be executed locally. The offloading decision and execution process of the proposed scheme are shown in Fig. 2.

In the scheme, the total number of the data packets be offloaded to the MEC server at t, $A^m_{k_n}(t)$, and the total number of the data packets be executed locally at t, $A^l_{k_n}(t)$, are given respectively by

$$A^m_{k_n}(t) = L_{k_n}(t)\alpha_{k_n}(t)N^f_{k_n}(t) \tag{3}$$

$$A^l_{k_n}(t) = L_{k_n}(t)\beta_{k_n}(t)N^f_{k_n}(t) + L_{k_n}(t)N^l_{k_n}(t) \tag{4}$$

Based on the above analysis, due to the constraint on $A^m_{k_n}(t)$, the total number of the packets to be offloaded to the MEC server cannot exceed the transmission capacity at t, namely, $A^m_{k_n}(t) \leqslant R^m_{k_n}(t)$.

2.1.3 Queue model

Generally, to avoid drastically increasing latency and energy consumption, the task is divided into several subtasks, vehicles will decide to offload parts of flexible subtasks to the MEC server to ensure the transmission performance according to the data packet queues. Each

Table 1 Notation

Notation	Definition
N	Number of RSUs
$\mathcal{N}$	Set of RSUs
K_n	Number of vehicles connects to RSU n
$\mathcal{K}_n$	Set of vehicles connects to RSU n
$H^n_{k_n}$	Channel gain between k_n and RSU n
B	Channel bandwidth
N_0	Noise power spectral density
S	Size of a data packet
I_{k_n}	Interference power from other RSU to vehicle k_n
$p^m_{k_n}$	Transmission power of k_n
$p^l_{k_n}$	Local execution power of k_n
$R^m_{k_n}$	Transmission rate of k_n
$C^m_{k_n}$	Number of packets executed by the MEC server
$C^l_{k_n}$	Number of packets executed by k_n
$N^f_{k_n}$	Number of flexible subtasks of k_n
$N^l_{k_n}$	Number of local subtasks of k_n
$\alpha_{k_n}, \beta_{k_n}$	Offloading decision
F^m	Total computation resource of the MEC server
$F^m_{k_n}$	Allocated computation resource to k_n by the MEC server
$F^l_{k_n}$	Local computation resource
L_{k_n}	Packet number of each subtask of k_n
Z_{k_n}	Computation intensity of each subtask of k_n
$E^m_{k_n}$	Transmission energy consumption of flexible subtask of k_n
$E^{l1}_{k_n}$	Local energy consumption of flexible subtasks of k_n
$E^{l2}_{k_n}$	Local energy consumption of local subtasks of k_n
$A^m_{k_n}$	Number of arrival packets in the transmission queue of k_n
$A^l_{k_n}$	Number of arrival packets in the local queue of k_n
$Q^m_{k_n}$	Transmission queue of k_n
$Q^l_{k_n}$	Local queue of k_n
$D^m_{k_n}$	Number of drop packets in the transmission queue of k_n
$D^l_{k_n}$	Number of drop packets in the local queue of k_n
$q^m_{k_n}$	Packet drop penalty in the transmission queue of k_n
$q^l_{k_n}$	Packet drop penalty in the local queue of vehicle k_n

flexible subtask consists of several data packets, therefore, the offloading decisions for flexible subtasks are equivalent to finding the optimal decisions for the associated data packets so as to minimize the network utility. Specifically, $Q^m_{k_n}(t)$ is denoted as the transmission queue of vehicle k_n at time t, which includes the data packets to be offloaded to the MEC server. $Q^l_{k_n}(t)$ is the local queue which includes the data packets to be executed locally. Consequently, the transmission queue and the local queue of vehicle k_n at $t+1$ are given respectively by

$$Q^m_{k_n}(t+1) = \max\{Q^m_{k_n}(t) - C^m_{k_n}(t) - D^m_{k_n}(t), 0\} + A^m_{k_n}(t) \tag{5}$$

$$Q^l_{k_n}(t+1) = \max\{Q^l_{k_n}(t) - C^l_{k_n}(t) - D^l_{k_n}(t), 0\} + A^l_{k_n}(t) \tag{6}$$

where $Q^m_{k_n}(0) = 0, Q^l_{k_n}(0) = 0$. In the transmission queue and the local queue of vehicle k_n at t, $D^m_{k_n}(t)$ and $D^l_{k_n}(t)$ are the number of drop packets due to the delay constraint, respectively. $A^m_{k_n}(t)$ and $A^l_{k_n}(t)$ are the number of packets to be offloaded to the MEC server and executed locally, respectively, which are calculated as the arrival data packets at t in the queue updating process. $C^m_{k_n}(t)$ and $C^l_{k_n}(t)$ denote the number of packets executed by the MEC server and the local vehicle at t, respectively, which are related to the allocated computational resource by the MEC server $F^m_{k_n}(t)$, the local computation capability $F^l_{k_n}$, as well as the computation intensity of the subtask. Different vehicles have different computation capabilities according to the local CPU frequency.

Additionally, to guarantee the transmission requirement, data packets will be dropped when the delay constraint is violated, namely, the maximum queue length is exceeded, which is given by

$$Q^m_{k_n}(t) > Q^{m,\max}_{k_n}(t) \tag{7}$$

$$Q^l_{k_n}(t) > Q^{l,\max}_{k_n}(t) \tag{8}$$

where $Q^{m,\max}_{k_n}(t)$ and $Q^{l,\max}_{k_n}(t)$ are the maximum transmission queue length and local queue length at t, respectively.

2.1.4 Energy consumption

Consider the task of vehicle k_n at t is composed of $N^l_{k_n}(t)$ local subtasks and $N^f_{k_n}(t)$ flexible subtasks, $E^m_{k_n}(t)$ denotes the energy consumption to offload the data packets $A^m_{k_n}(t)$ of $\alpha_{k_n}(t)N^f_{k_n}(t)$ flexible subtasks to the MEC server, which is given by

$$E^m_{k_n}(t) = \frac{p^m_{k_n}A^m_{k_n}(t)}{R^m_{k_n}(t)} \tag{9}$$

where $p^m_{k_n}$ is the transmission power. Moreover, $E^l_{k_n}(t)$ denotes the local energy consumption of k_n at t, which is given by

$$E^l_{k_n}(t) = E^{l1}_{k_n}(t) + E^{l2}_n(t) = \frac{p^l_{k_n}A^l_{k_n}(t)}{C^l_{k_n}(t)} \tag{10}$$

where $p^l_{k_n}$ is the local computation power. $E^l_{k_n}(t)$ consists of energy consumption when the data packets $A^l_{k_n}(t)$ of $\beta_{k_n}(t)N^l_{k_n}(t)$ flexible subtasks and $N^l_{k_n}(t)$ local subtasks are executed locally, which are denoted as $E^{l1}_{k_n}(t)$

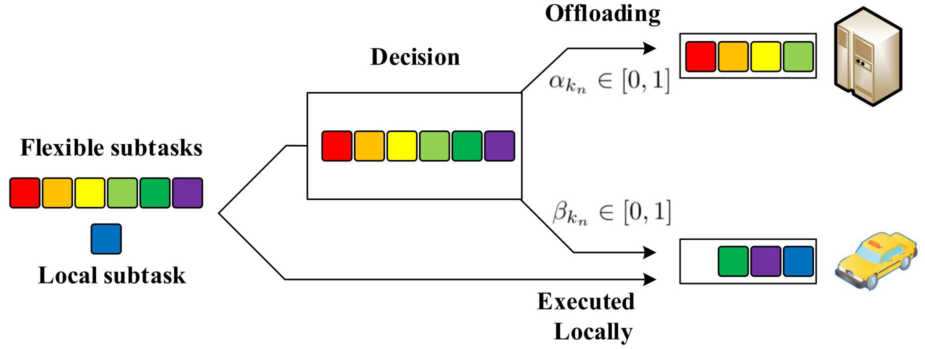

Fig. 2 Process of offloading and execution. The task generated by the vehicle are classified into two subtasks at *t*, the local subtasks should be executed locally, whereas the flexible subtasks could be executed locally or offloaded to the MEC server depending on the offloading decision

and $E^{l2}_{k_n}(t)$, respectively. Notice that, parts of data packets will be dropped when the delay constraint can not be satisfied, and the associated energy consumption is not included in the total energy consumption model. Moreover, the MEC server is powered on, the computational energy consumption can be neglected. Besides, energy consumption and the backward transmission time of results to vehicle k_n from the MEC server also could be neglected [30].

2.1.5 Problem formulation

In this paper, we aim at minimizing the utility of vehicular networks, including energy consumption and packet drop rate, which is given by

$$U(t) = \sum_{k_n \in \mathcal{K}_n} \left(q^m_{k_n} D^m_{k_n}(t) + q^l_{k_n} D^l_n(t) \right) + E_{k_n}(t) \tag{11}$$

where $q^m_{k_n}$ and $q^l_{k_n}$ are the packet drop penalty factor for the transmission queue and the local queue of vehicle k_n, respectively. $E_{k_n}(t)$ denotes the total energy consumption of vehicle k_n at t, which includes energy consumption of offloading data packets to the MEC server and locally executing data packets. Therefore, the total energy consumption can be expressed as

$$\begin{aligned} E_{k_n}(t) &= E^m_{k_n}(t) + E^{l1}_{k_n}(t) + E^{l2}_{k_n}(t) \\ &= \frac{p^m_{k_n} L_{k_n}(t) \alpha_{k_n}(t) N^f_{k_n}(t)}{R^m_{k_n}(t)} + \frac{p^l_{k_n} L_{k_n}(t) \beta_{k_n}(t) N^f_{k_n}(t)}{C^l_{k_n}(t)} + \frac{p^l_{k_n} L_{k_n}(t) N^l_{k_n}(t)}{C^l_{k_n}(t)} \end{aligned} \tag{12}$$

Consequently, the optimization problem $\mathcal{P}_1(t)$ can be formulated as:

$$\begin{aligned} &\min_{\alpha_{k_n}(t), \beta_{k_n}(t)} \quad U(t) \\ &\text{s.t.}\ (C1) \quad C^m_{k_n}(t) \leqslant \frac{F^m}{Z_{k_n} Q^m_{k_n}(t)} \\ &\qquad (C2) \quad A^m_{k_n}(t) \leqslant R^m_{k_n}(t) \\ &\qquad (C3) \quad \alpha_{k_n}(t) + \beta_{k_n}(t) = 1 \\ &\qquad (C4) \quad N^f_{k_n}(t) \geqslant 0, \quad N^l_{k_n}(t) \geqslant 0, \\ &\qquad (C5) \quad p^m_{k_n}(t) > 0, \quad p^l_{k_n}(t) > 0 \end{aligned} \tag{13}$$

where $(C1)$ indicates the maximum number of offloaded packets $C^m_{k_n}(t)$ to the MEC server at t, F^m is the total computation resource of the MEC server. $(C2)$ is the constraint on the number of flexible packets offloaded to the MEC server, which cannot exceed the packet transmission rate. $(C3)$ is the offloading decision vector constraint, which indicates the offloading decision for the flexible subtask of k_n, $0 \leqslant \alpha_{k_n}(t) \leqslant 1, 0 \leqslant \beta_{k_n}(t) \leqslant 1$. In $(C4)$, when $N^f_{k_n}(t) = 0$, there is no flexible subtask. $(C5)$ is the traditional transmission power and local execution power constraint.

3 Computation resource allocation and offloading decision-making

Based on the above analysis, the optimization problem could be solved by a computation resource allocation algorithm and a Lyapunov-based dynamic offloading decision (LDOD) algorithm, which can be used to obtain $C^m_{k_n}(t)$ and $\mathcal{P}_1(t)$, respectively.

3.1 Computation resource allocation

Moreover, we dynamically adjust the computation resource of the MEC server allocated to vehicle k_n with respect to the computation intensity of subtasks. $F^m_{k_n}(t)$ and $F^l_{k_n}$ are the computation resource allocated to vehicle k_n at t by the MEC server and local vehicle respectively, and $F^m_{k_n}(t) \leqslant F^m$. Specifically, to minimize the maximum task execution time for all offloading vehicles, the optimization problem of computation resource allocation is given by:

$$\begin{aligned}
&\min_{F_{k_n}^m(t)} \max_{k_n} \quad \sum\nolimits_{k_n\in\mathcal{K}_n} \frac{Z_{k_n} Q_{k_n}^m(t)}{F_{k_n}^m(t)}\\
&\text{s.t. } (C6) \quad \sum\nolimits_{k_n\in\mathcal{K}_n} F_{k_n}^m(t) = F^m\\
&\qquad (C7) \quad F_{k_n}^m(t) > 0\\
&\qquad (C8) \quad Q_{k_n}^m(t) > 0
\end{aligned} \tag{14}$$

Constraints $(C6)$ and $(C7)$ represent the total computation resources allocated to vehicles at t, $(C8)$ is the constraint on transmission queue of vehicle k_n at t.

3.2 Lyapunov optimization

The queue delay, the packet drop rate and energy consumption are jointly considered to make the optimal offloading decisions and minimize the utility of vehicular networks. The Lyapunov optimization function can be represented as

$$L(t) = \frac{1}{2} \sum_{n\in\mathcal{K}_n} Q_{k_n}^m(t)^2 + Q_{k_n}^l(t)^2 \tag{15}$$

The Lyapunov drift is given by

$$\Delta(t) = L(t+1) - L(t) \tag{16}$$

The Lyapunov penalty item includes the packet drop cost $\sum\limits_{k_n\in\mathcal{K}_n} \left(q_{k_n}^m D_{k_n}^m(t) + q_{k_n}^l D_{k_n}^l(t)\right)$ and total energy consumption $\sum\limits_{k_n\in\mathcal{K}_n} E_{k_n}(t)$, which is given by

$$VU(t) = V \sum_{k_n\in\mathcal{K}_n} \left(q_{k_n}^m D_{k_n}^m(t) + q_{k_n}^l D_{k_n}^l(t)\right) + V \sum_{k_n\in\mathcal{K}_n} E_{k_n}(t) \tag{17}$$

The control parameter V indicates the importance of energy consumption and the number of drop packets. The larger V indicates the higher priority of the packet drop number and energy consumption are considered in the utility function than the stability of the queue. In other words, the smaller the V, the higher the priority of the queue stability in the packet offloading decision. Therefore, to ensure the stability of the data packet queue while minimizing Lyapunov penalty, a Lyapunov drift-plus-penalty item is included in the queue, which is formulated as

$$\Delta(t) + V \sum_{k_n\in\mathcal{K}_n} \left(q_{k_n}^m D_{k_n}^m(t) + q_{k_n}^l D_{k_n}^l(t)\right) + V \sum_{k_n\in\mathcal{K}_n} E_{k_n}(t) \tag{18}$$

Accordingly, the original optimization problem $\mathcal{P}_1(t)$ can be transferred to the equivalent Lyapunov drift-plus-penalty minimization problem $\mathcal{P}_2(t)$, which can be represented as

$$\begin{aligned}
&\min_{\alpha_{k_n}(t),\beta_{k_n}(t)} \quad \Delta(t) + V \sum_{k_n\in\mathcal{K}_n} \left(q_{k_n}^m D_{k_n}^m(t) + q_{k_n}^l D_{k_n}^l(t)\right) + V \sum_{k_n\in\mathcal{K}_n} E_{k_n}(t)\\
&\text{s.t.} (C2) - (C5)\\
&\quad (C9) \quad C_{k_n}^m(t) + D_{k_n}^m(t) \leqslant Q_{k_n}^m(t)\\
&\quad (C10) \quad C_{k_n}^l(t) + D_{k_n}^l(t) \leqslant Q_{k_n}^l(t)
\end{aligned} \tag{19}$$

In (C9)–(C10), the total number of data packets, which are offloaded to the MEC server or executed locally, could not exceed the current data packet queues to ensure the stability of the queue.

Based on [31], with the given constant K, the Lyapunov drift-plus-penalty should satisfy

$$\begin{aligned}
&\Delta(t) + V \sum_{k_n\in\mathcal{K}_n} \left(q_{k_n}^m D_{k_n}^m(t) + q_{k_n}^l D_{k_n}^l(t)\right) + V \sum_{k_n\in\mathcal{K}_n} E_{k_n}(t)\\
&= L(t+1) - L(t) + V \sum_{k_n\in\mathcal{K}_n} \left(q_{k_n}^m D_{k_n}^m(t) + q_{k_n}^l D_{k_n}^l(t)\right) + V \sum_{k_n\in\mathcal{K}_n} E_{k_n}(t)\\
&= \frac{1}{2} \sum_{k_n\in\mathcal{K}_n} \{(C_{k_n}^m(t) + D_{k_n}^m(t))^2 + A_{k_n}^m(t)^2 + (C_{k_n}^l(t) + D_{k_n}^l(t))^2 + A_{k_n}^l(t)^2\}\\
&\quad - \sum_{k_n\in\mathcal{K}_n} \{Q_{k_n}^m(t) C_{k_n}^m(t)\} - \sum_{k_n\in\mathcal{K}_n} \{Q_{k_n}^l(t) C_{k_n}^l(t)\} + \mathcal{P}_3(t) - \mathcal{P}_4(t)\\
&\leqslant K - \sum_{k_n\in\mathcal{K}_n} \{Q_{k_n}^m(t) C_{k_n}^m(t)\} - \sum_{k_n\in\mathcal{K}_n} \{Q_{k_n}^l(t) C_{k_n}^l(t)\} + \mathcal{P}_3(t) - \mathcal{P}_4(t)
\end{aligned} \tag{20}$$

where K at t is given by

$$\begin{aligned}
K = &\frac{1}{2} \sum_{k_n\in\mathcal{K}_n} \left(C_{k_n}^{m,\max}(t) + D_{k_n}^{m,\max}(t)\right)^2 + A_{k_n}^{m,\max}(t)^2\\
&+ \left(C_{k_n}^{l,\max}(t) + D_{k_n}^{l,\max}(t)\right)^2 + A_{k_n}^{l,\max}(t)^2
\end{aligned} \tag{21}$$

According to Eq. (20), the optimal Lyapunov drift-plus-penalty, offloading decisions, energy consumption, and the packet drop strategy could be obtained. Consequently, the optimization problem $\mathcal{P}_2(t)$ is converted to minimize optimization sub-problem $\mathcal{P}_3(t)$ and maximize optimization sub-problem $\mathcal{P}_4(t)$ which are respectively given by

$$\begin{aligned}
\mathcal{P}_3(t) = &\sum_{k_n\in\mathcal{K}_n} \alpha_{k_n}(t) \left(L_{k_n}(t) Q_{k_n}^m(t) N_{k_n}^f(t) + VE_{k_n}^m(t)\right)\\
&+ \sum_{k_n\in\mathcal{K}_n} \beta_{k_n}(t) \left(L_{k_n}(t) Q_{k_n}^l(t) N_{k_n}^f(t) + VE_{k_n}^{l1}(t)\right)\\
&+ \sum_{k_n\in\mathcal{K}_n} L_{k_n}(t) Q_{k_n}^l(t) N_{k_n}^l(t) + VE_{k_n}^{l2}(t)
\end{aligned} \tag{22}$$

$$\mathcal{P}_4(t) = \sum_{k_n \in \mathcal{K}_n} D^m_{k_n}(t)\left(Q^m_{k_n}(t) - Vq^m_{k_n}\right) + D^l_{k_n}(t)\left(Q^l_{k_n}(t) - Vq^l_{k_n}\right) \tag{23}$$

$\mathcal{P}_3(t)$ is related to energy consumption and the stability of queues, which will generate optimal offloading decisions. Whereas $\mathcal{P}_4(t)$ is related to the number of drop packets, which will decide the packets drop strategy.

3.3 Offloading decisions and energy consumption

Accordingly, optimal offloading decisions, energy consumption, and the state of queues at t can be obtained by minimizing $\mathcal{P}_3(t)$, which is composed of K_n polynomials due to K_n vehicles. The optimal offloading decisions of vehicle k_n can be obtained by minimizing polynomial k_n. Therefore, $\mathcal{P}_3(t)$ is given by

$$\min_{\alpha_{k_n}(t),\beta_{k_n}(t)} \alpha_{k_n}(t)\mathcal{W}_1(t) + \beta_{k_n}(t)\mathcal{W}_2(t) + VE^{l2}_{k_n}(t) \tag{24}$$

where $\mathcal{W}_1(t) = L_{k_n}(t)Q^m_{k_n}(t)N^f_{k_n}(t) + VE^m_{k_n}(t)$, $\mathcal{W}_2(t) = L_{k_n}(t)Q^l_{k_n}(t)N^f_{k_n}(t) + VE^{l1}_{k_n}(t)$.

For vehicle k_n, energy consumption and the stability of queues are different with respect to offloading decisions. The optimal offloading decision vectors $\alpha^*_{k_n}(t), \beta^*_{k_n}(t)$ could be obtained by

$$\begin{aligned} \arg\min_{\alpha^*_{k_n}(t),\beta^*_{k_n}(t)} \quad & \alpha_{k_n}(t)\mathcal{W}_1(t) + \beta_{k_n}(t)\mathcal{W}_2(t) + VE^{l2}_{k_n}(t) \\ \text{s.t.} \quad & \text{(C2)–(C5), (C9)–(C10)} \end{aligned} \tag{25}$$

Based on the optimal offloading decisions, the total energy consumption can be obtained by

$$\begin{aligned} E_{k_n}(t) =& E^m_{k_n}(t) + E^{l1}_{k_n}(t) + E^{l2}_{k_n}(t) \\ =& \frac{p^m_{k_n}L_{k_n}(t)\alpha^*_{k_n}(t)N^f_{k_n}(t)}{R^m_{k_n}(t)} + \frac{p^l_{k_n}L_{k_n}(t)\beta^*_{k_n}(t)N^f_{k_n}(t)}{C^l_{k_n}(t)} \\ &+ \frac{p^l_{k_n}L_{k_n}(t)N^l_{k_n}(t)}{C^l_{k_n}(t)} \end{aligned} \tag{26}$$

Thus, the optimal offloading decisions $\alpha^*_{k_n}(t), \beta^*_{k_n}(t)$ can be obtained by minimizing $\mathcal{W}(t)$, as shown in Eq. (25). Furthermore, energy consumption of vehicle k_n can also be obtained with the associated optimal offloading decisions, as shown in Eq. (26).

3.4 Packet drop strategy

To ensure the transmission latency of the vehicular network, the data packets which can not guarantee the delay constraint will be dropped. The number of drop packets in the transmission queue and the local queue could be obtained by maximizing $\mathcal{P}_4(t)$. $\mathcal{P}_4(t)$ is also composed of K_n polynomials, which indicates the associated vehicles, and the number of drop packets in the transmission queue $D^m_{k_n}(t)$ and the local queue $D^l_{k_n}(t)$ at t can be expressed respectively by

$$D^m_{k_n}(t) = \begin{cases} D^{m,\max}_{k_n}, & Q^m_{k_n}(t) > Vq^m_{k_n} \\ 0, & \text{others} \end{cases} \tag{27}$$

$$D^l_{k_n}(t) = \begin{cases} D^{l,\max}_{k_n}, & Q^l_{k_n}(t) > Vq^l_{k_n} \\ 0, & \text{others} \end{cases} \tag{28}$$

where $D^{m,\max}_{k_n}$, $D^{l,\max}_{k_n}$ are the maximum number of drop packets in the transmission queue and the local queue, respectively. In consequence, the packet drop strategy of the vehicular network can be obtained.

To optimize the offloading decisions and energy consumption, while ensuring the transmission delay and the packet drop rate, we propose a Lyapunov-based dynamic offloading decision algorithm, which is given by the following steps:

Step 1, the task of vehicle k_n is divided into flexible subtasks and local subtasks. All the local subtasks are executed locally, and parts of flexible subtasks will be offloaded to the MEC server according to the utility function. The utility of vehicle k_n includes energy consumption and penalty of packet drop, which is related to the transmission queue $Q^m_{k_n}(t)$ and the local queue $Q^l_{k_n}(t)$.

Step 2, calculate $C^l_{k_n}(t)$ based on the computation intensity of the subtask and the computation capacity of vehicle k_n at t. Moreover, $C^m_{k_n}(t)$ is related to the computation intensity of the subtask and $F^m_{k_n}(t)$, which could be obtained by computation resource allocation.

Step 3, vehicle k_n makes optimal offloading decisions for all flexible subtasks to minimize the utility of vehicular networks by the LDOD algorithm.

Assume the average number of vehicles is K, and the average number of flexible subtasks per vehicle is N_k. In the algorithm, the complexity of computation resource allocation is $O(K)$, and the complexity of making offloading decision is $O(KN_k)$. Therefore, the complexity of the algorithm is $O(K) + O(KN_k)$. The Lyapunov-based dynamic offloading decision algorithm is shown in Algorithm 1.

4 Simulation results and discussions

In this section, we set the main parameters, present the simulation results of the optimization algorithm, and evaluate the performance of the proposed LDOD algorithm in comparison with different algorithms in various aspects. The following algorithms are introduced for comparison: random offloading (RO) algorithm, full-offloading (FO) algorithm, and the mobility aware task offloading (MATO) algorithm, which was proposed in [32] to offload parts of the task so that the offloading delay of the subtask is the same as the local execution delay, thereby, to

Algorithm 1 Lyapunov-based Dynamic Offloading Decision Algorithm

```
Detect the data packet queue $Q^m_{k_n}(t)$, $Q^l_{k_n}(t)$, calculate the number of arrival packets $A^m_{k_n}(t)$ and $A^l_{k_n}(t)$.
Allocate computation resources of MEC server to vehicles, which is $F^m_{k_n}(t)$.
Calculate $C^m_{k_n}(t)$ and $C^l_{k_n}(t)$ based on the $Z_{k_n}$, $F^m_{k_n}(t)$ and $F^l_{k_n}$.
Let $\Phi = \mathcal{K}_n$.
for vehicle $k_n \in \Phi$ do
  Update the offloading decisions $\alpha^*_{k_n}(t), \beta^*_{k_n}(t)$ by Eq. (25);
  Update energy consumption $E_{k_n}(t)$ by Eq. (26);
  Update the packet drop strategies $D^m_{k_n}(t), D^l_{k_n}(t)$ by Eqs. (27) and (28), respectively;
  Update $A^m_{k_n}(t), A^l_{k_n}(t)$;
  if $A^m_{k_n}(t) = 0, A^l_{k_n}(t) = 0$ then
    break
  end if
  Update $Q^m_{k_n}(t), Q^l_{k_n}(t)$ by Eqs. (5), (6).
end for
Output optimal offloading decisions $\alpha^*_{k_n}(t), \beta^*_{k_n}(t)$, packet drop strategies $D^m_{k_n}(t), D^l_{k_n}(t)$ and energy consumption $E_{k_n}(t)$.
```

Table 2 Simulation parameters

Parameter	Value
Transmission power of vehicle k_n, $p^m_{k_n}$	100 mw
Local execution power of vehicle k_n, $p^l_{k_n}$	300 mw
Maximum number of drop packets in the transmission queue $D^{m,\max}_{k_n}$	200 packets
Maximum number of drop packets in the local queue $D^{l,\max}_{k_n}$	200 packets
CPU frequency of vehicle k_n, $F^l_{k_n}$	1 ∼ 3 GHz
CPU frequency of the MEC server, F^m	50 GHz
Traffic load, λ_{k_n}	5 ∼ 20 KB/t
Computation intensity, Z_{k_n}	500 ∼ 1000 cycle/b
Channel bandwidth B	1 MHz
Path loss model	ITU UMa/UMi [33]
Noise power N_0	− 100 dBm/Hz
Intra-interface I_{k_n}	d_0^{-4}
Packet length S	500 b

minimize the total delay. In the random offloading algorithm, vehicles randomly offload flexible subtasks to the MEC server, whereas, in the full offloading algorithm, all the flexible subtasks will be offloaded to the MEC server.

4.1 Parameter setting

Consider the coverage area of each RSU is a circle of radius 200 m, and $k_n = 10$ vehicles are randomly distributed in two unidirectional lanes within the coverage of RSU n, and they do not move out of the coverage of RSU n during ΔT. The speed of vehicles is 40 km/h, and the distance between adjacent vehicles in the same lane is not less than 10 m. Assume time period is 10 ms, the data traffic satisfies Poisson distribution. The main parameters used in the simulation are described in Table 2.

4.2 Performance analysis

In Fig. 3, we evaluate the average packet drop rate versus V with various traffic load for the LDOD algorithm, where the packet drop penalty factor $q^m_{k_n} = q^l_{k_n} = 2$. From the figure, the average packet drop rate decreases rapidly with the increasing V, then decreases slowly and approaches to zero finally. Obviously, when V is increasing, the queue delay constraint will be relaxed, resulting in smaller number of dropped packets. In addition, the average packet drop rate increases with the increasing traffic load λ_{k_n}, packets are more likely to violate queue delay constraints under the heavy traffic load.

Figure 4 shows the average energy consumption versus V with various traffic load for the LDOD algorithm. From the simulation results, we can seen that the average energy consumption for the LDOD decreases with the increases of V. Specifically, when V increases, energy consumption has higher priority in the offloading decision process. When V is large enough, the total average energy consumption decreases and approaches to a stable value. With the same V, the average energy consumption of the transmission and the local execution increases with the increase of the traffic load λ_{k_n}.

Figure 5 depicts the average queue length versus V with various traffic load for the LDOD algorithm. Obviously, it can be seen that the average queue length increases with the increasing V, and the queue delay constraint will be relaxed. Additionally, larger λ_{k_n} will result in more arriving packets at each time slot and the average queue length will be increased.

Figure 6 presents the average packet drop rate versus V with various packet drop penalty factor for the LDOD algorithm, where $q^m_{k_n} = q^l_{k_n}$ denoted as q, and $\lambda_{k_n} = 14$. With the same V, the increasing q relaxes the queue length constraint; hence, average packet drop rate is a decreasing function of q. Moreover, the average packet drop rate decreases rapidly with the increasing V, due to the relaxation of the delay constraint. Finally, the packet drop rate on the transmission queue will quickly drop to 0 and the packet drop rate on local queue decreases gradually, resulting in the decrease of the average packet drop rate.

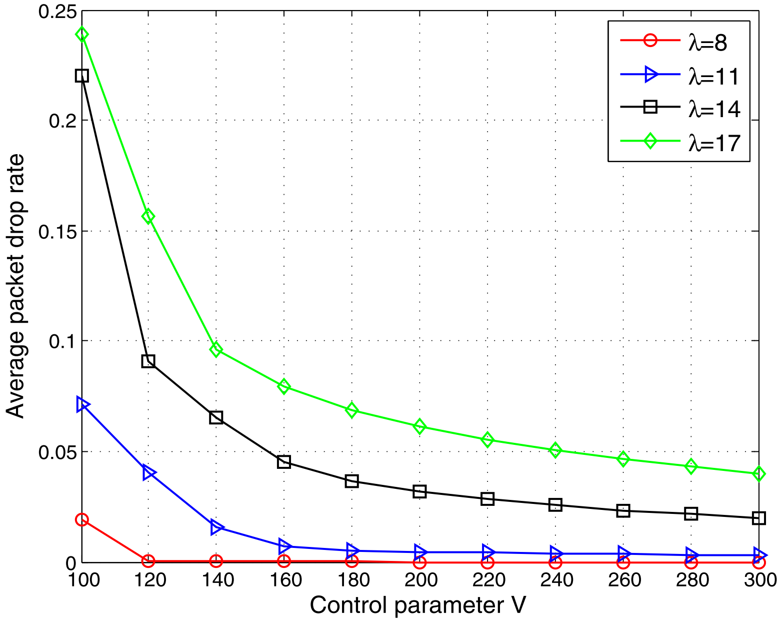

Fig. 3 The average packet drop rate versus the control parameter *V* with different λ

In Figs. 7 and 8, we evaluate the average energy consumption and the average queue length versus *V* with various packet drop penalty factor for the LDOD algorithm. From the simulation results, we can see that the average energy consumption is a decreasing function of *V*, whereas the average queue length is an increase function of *q*. With the same *V*, the average packet drop rate is a decreasing function of *q*, which can be obtained from Fig. 6. Therefore, the weight of packet drop rate in the network utility decreases with the increasing *q*, while the weight of the average energy consumption increases and more flexible subtasks will be offloaded to

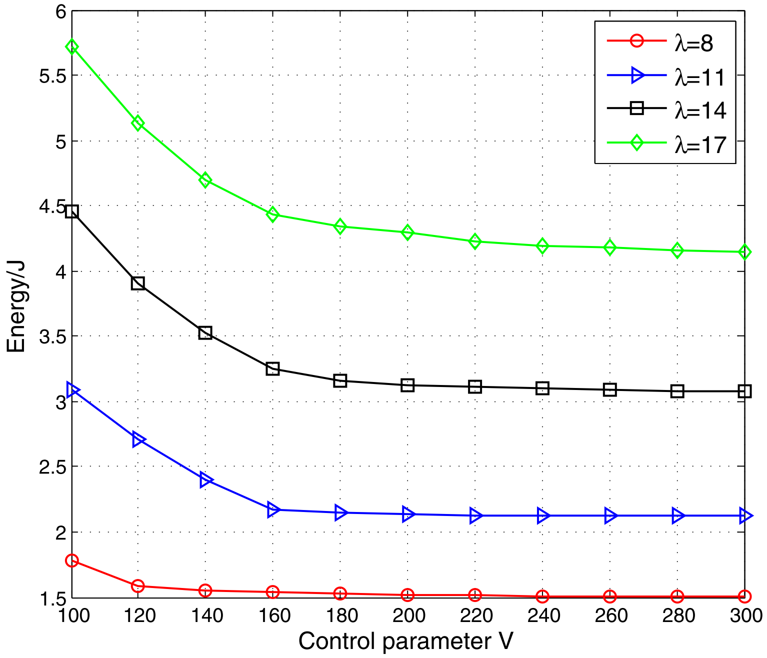

Fig. 4 The average energy consumption versus the control parameter *V* with different λ

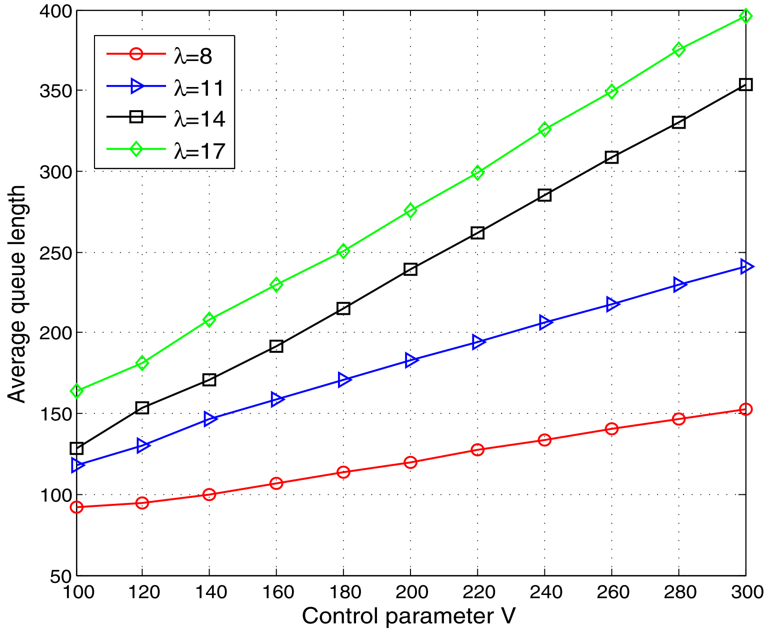

Fig. 5 The average queue length versus the control parameter *V* with different λ

the MEC server. Similarly, the increasing q could relax the queue delay constraint and increase the average queue length.

Figure 9 presents the computation resources allocated by MEC server to vehicles for the LDOD algorithm when $t = 2, V = 100, q = 2, \lambda_{k_n} \in [5, 20]$. The bars in the figure indicate the ratio of the allocated computation resources of MEC server. Figure 10 presents the offloading decisions of flexible subtasks of vehicle k_n at different time slots when $V = 100, q = 2, \lambda_{k_n} \in [5, 20]$. In the figure, the blue bar represents the proportion of the offloaded flexible subtasks to the MEC server, whereas the yellow bar shows the proportion of the locally executed flexible subtask.

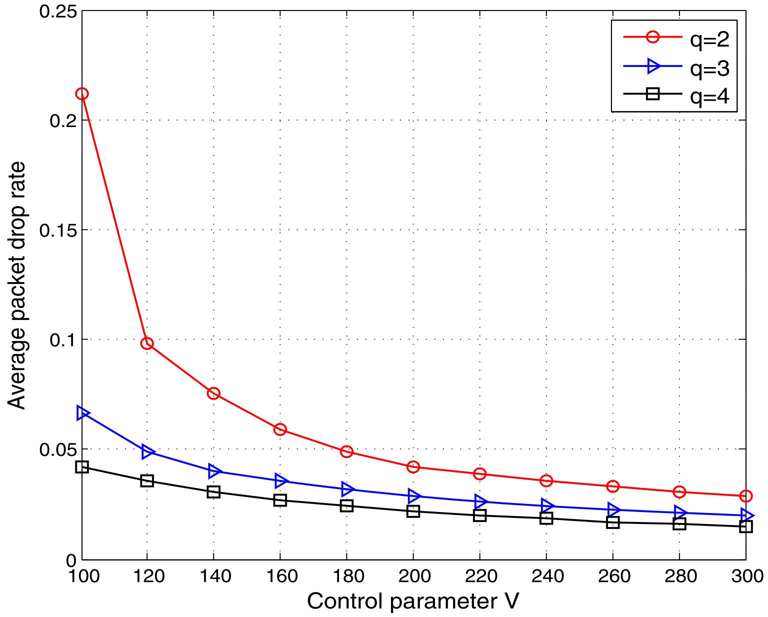

Fig. 6 The average packet drop rate versus the control parameter *V* with different *q*

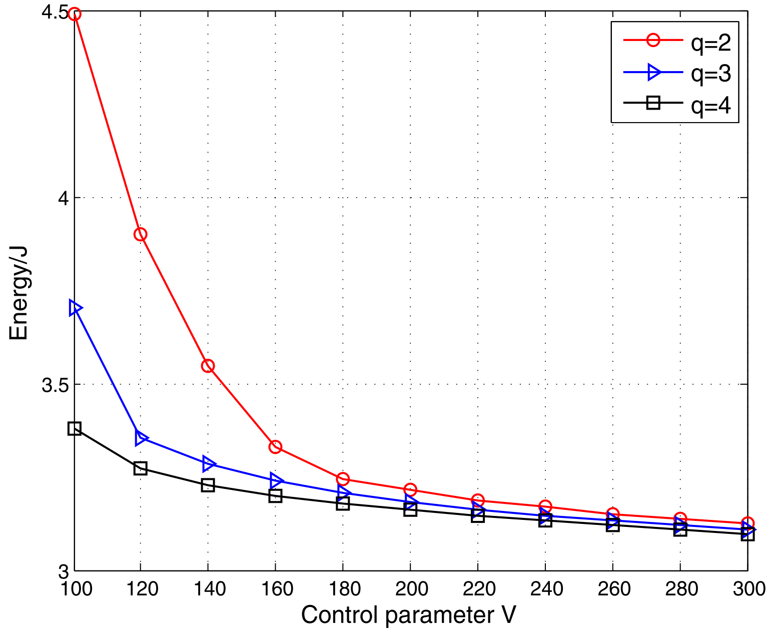

Fig. 7 The average energy consumption versus the control parameter *V* with different *q*

The time varying average length of the transmission queue and the local queue in the LDOD algorithm are shown in Figs. 11 and 12, respectively, where $q = 2, \lambda_{k_n} \in [5, 20]$. From the figures, the average length of the transmission queue and the local queue are increasing initially, and slowly approach to the convergence. Moreover, the length of the queue is an increasing function of *V*, the length of the queue at $V = 100$ is less than the length of the queue at $V = 300$. Meanwhile, the average length of the local queue is longer than the transmission queue with the same *V*, due to the higher execution rate of the transmission queue compared with that of the local queue.

Figure 13 is the comparison of the average packet drop rate of the LDOD, the FO, and the RO algorithms. It

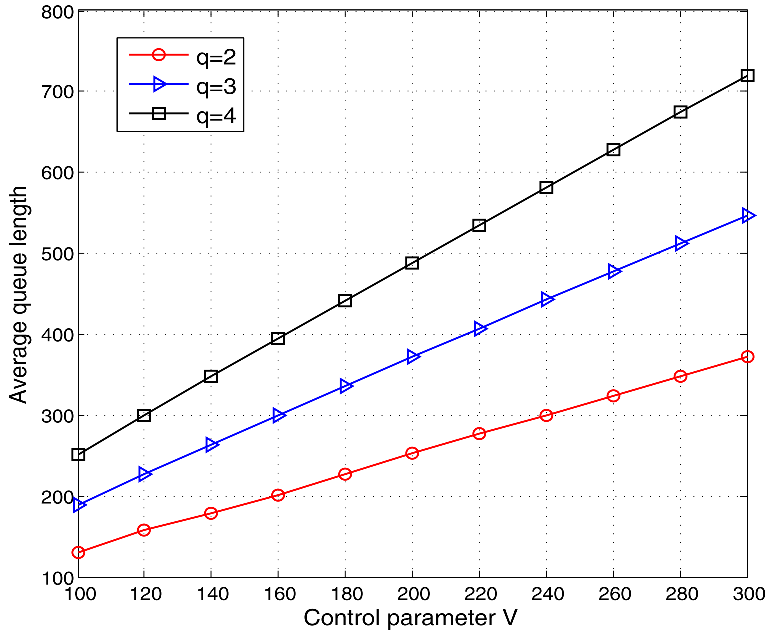

Fig. 8 The average queue length versus the control parameter *V* with different *q*

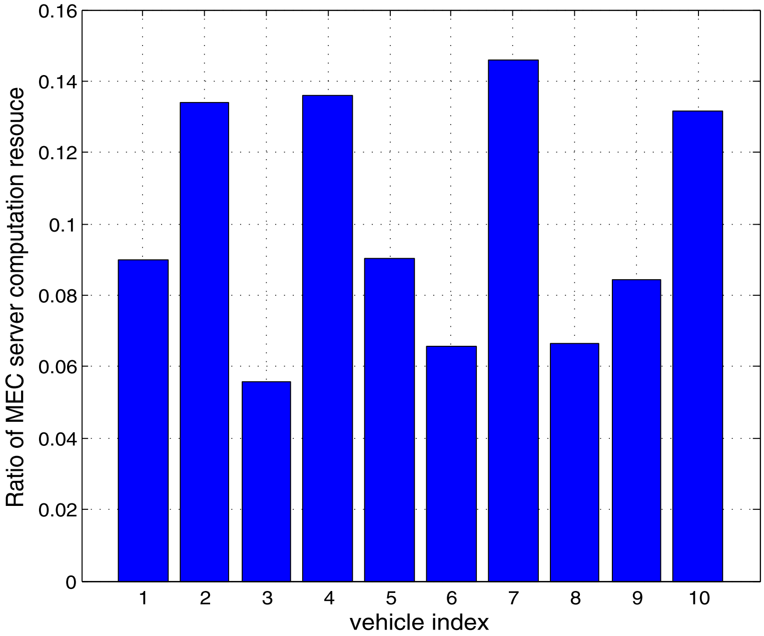

Fig. 9 The ratio of the computation resource allocated to vehicles to the total computation resource of the MEC server

can be seen that the average packet drop rate decreases with the increases of V for all algorithms. Meanwhile, in the FO algorithm, all the flexible packets are offloaded to the MEC server, resulting in a large packet drop rate in the transmission queue, with a small local queue. Thus, the packet drop rate of the FO algorithm is larger than that of the LDOD algorithm. When V is small, the packet drop rate of the MATO algorithm is smaller than that of the proposed algorithm. However, the weight of packet drop rate in offloading decisions also increases with increasing V, and the packet drop rate of the MATO algorithm is larger than that of the proposed algorithm with increasing V. Obviously, in the packet drop rate, the LDOD algorithm has better performance than the

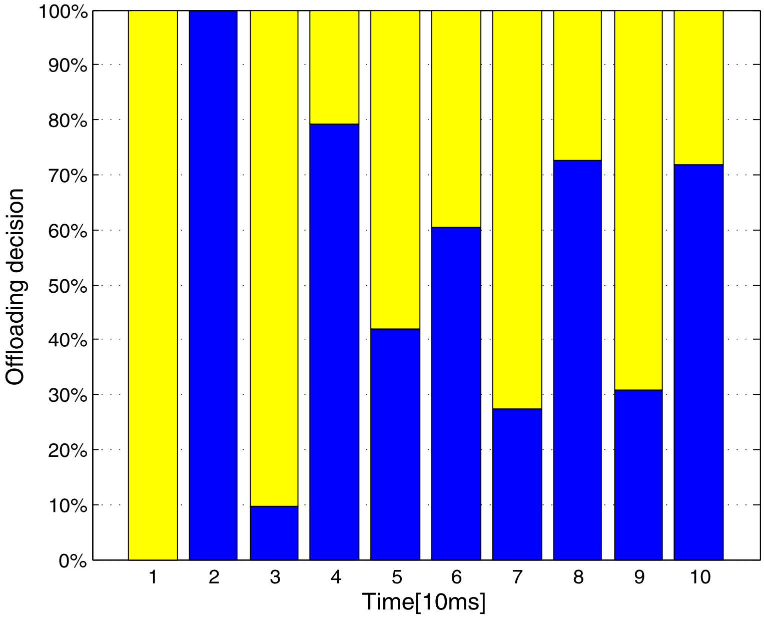

Fig. 10 The task offloading decisions of vehicle k_n at different time periods

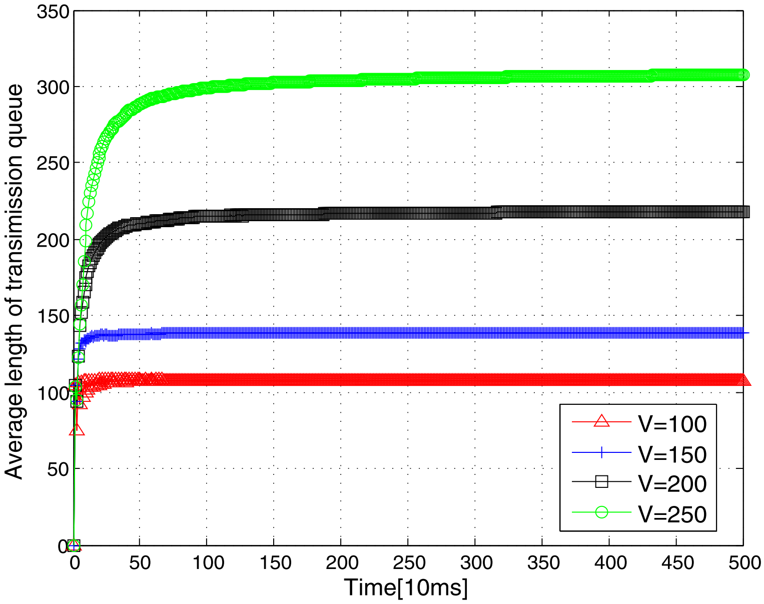

Fig. 11 The average length of the transmission queue versus time with different *V*

comparison algorithms, the RO algorithm has the worst performance.

The comparison of different algorithms with respect to the average energy consumption is given in Fig. 14. Among them, the average energy consumption of the LDOD algorithm decreases with the increasing *V*, whereas the RO algorithm and the FO algorithm obtain the highest and the lowest average energy consumption, respectively. In addition, based on the simulation results, the energy consumption of the MATO algorithm is larger than that of the proposed algorithm with increasing *V*. Moreover, the local execution power consumption is relatively larger than the transmission power consumption, as a result, the FO has the best performance in energy saving. Specifically, when vehicles decide to offload tasks simultaneously, the MEC server will be overloaded, resulting

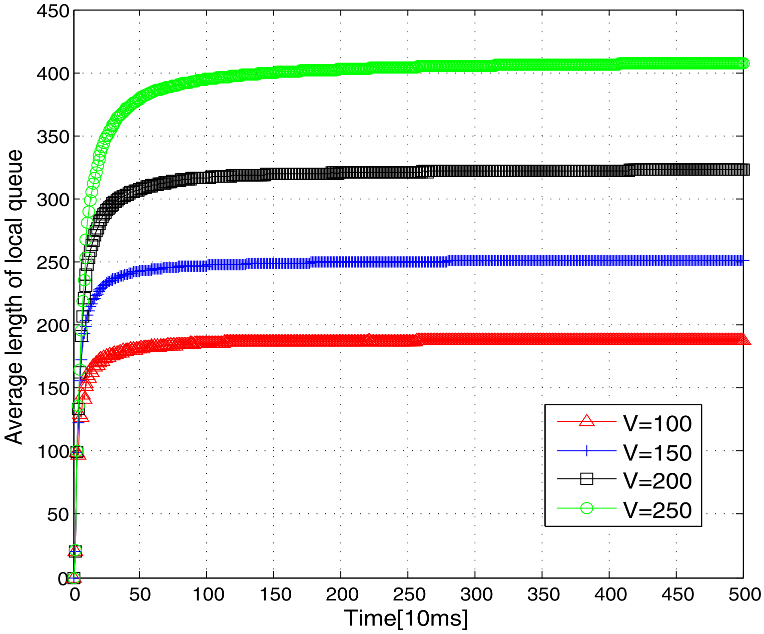

Fig. 12 The average length of the local queue versus time with different *V*

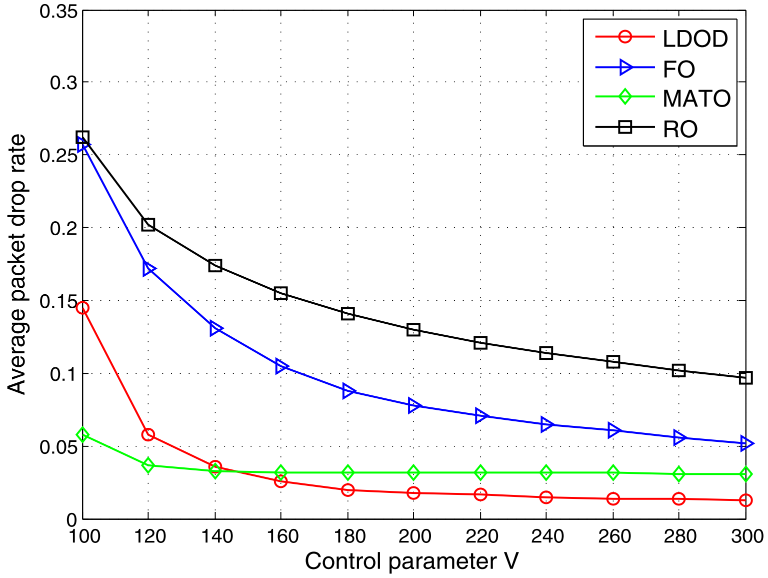

Fig. 13 The average packet drop rate versus *V* with the proposed algorithm, MATO algorithm, and RO and FO algorithm

in a high packet drop rate in the FO algorithm. Dynamically adjusting the control parameter *V* could optimize the tradeoff between the packet drop rate, energy consumption, and the stability of the queue.

5 Conclusion

In this paper, we investigated the task offloading decision in vehicular networks by jointly considering energy consumption and packet drop rate. The tasks of vehicles were classified into local subtasks and flexible subtasks according to the particular properties. Moreover, a dynamic task offloading scheme was proposed in the offloading process, which would execute flexible subtasks by the local terminals or the MEC server based on the offloading decision. To further improve the task offloading performance, a computation resource allocation scheme was proposed to allocate the computation resource of MEC server to vehicles. Based on that, the equivalent Lyapunov drift-plus-penalty minimization problem was proposed to minimize the utility, while ensuring the stability of the queue.

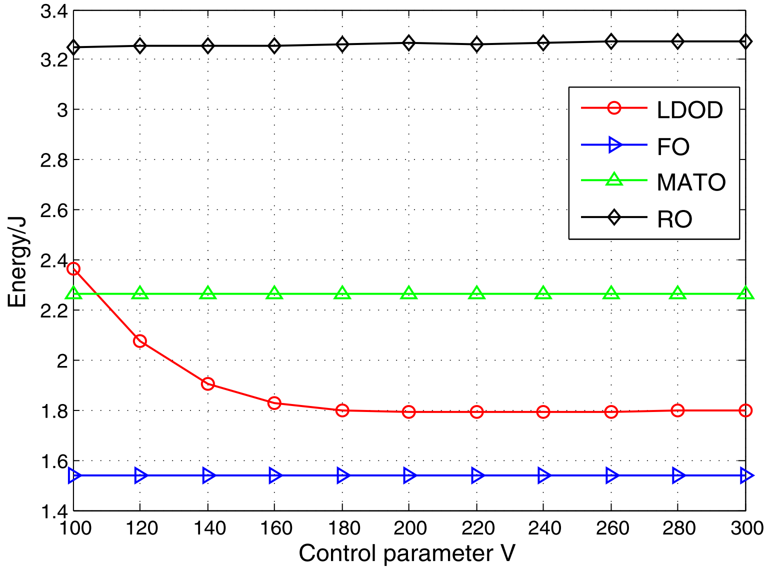

Fig. 14 The average energy consumption versus *V* with the proposed algorithm, MATO algorithm, and RO and FO algorithm

Abbreviations

AR: Augmented reality; FC: Fog computing; FO: Full-offloading; IoV: Internet of Vehicles; LDOD: Lyapunov-based dynamic offloading decision; MATO: Mobility aware task offloading MCC: Mobile cloud computing; MEC: Mobile edge computing; OBU: On-board unit; OFDM: Orthogonal frequency division multiplexing; QoE: Quality of experience; RO: Random offloading; RSU: Road side unit; SDNi-MEC: Software-defined network inside the mobile edge computing; SINR: Signal-to-interference-plus-noise ratio; V2I: Vehicle-to-infrastructure; V2V: Vehicle-to-vehicle; VANET: Vehicular ad hoc network; VEC: Sehicular edge computing; VR: Virtual reality

Acknowledgements

This work is supported by the National Natural Science Foundation of China (NSFC) (61831002,61401053), and Innovation Project of the Common Key Technology of Chongqing Science and Technology Industry (grant no. cstc2018jcyjAX0383)

Authors' contributions

XH designed the idea, main algorithms and the drafted manuscript. KX modify the manuscript and designed the simulations. CL worked with data analysis and encoding. QC and JZ gave some suggestion and reviewed the manuscript. All authors have made substantive intellectual contributions to this study. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Author details

[1]School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongwen Road, Chongqing, China. [2]The University of Sheffield, Sheffield, UK.

Received: 29 July 2019 Accepted: 16 January 2020
Published online: 05 February 2020

References

1. J. Ren, H. Guo, C. Xu, Y. Zhang, Serving at the edge: a scalable IoT architecture based on transparent computing. IEEE Netw. **31**(5), 96–105 (2017)
2. M. Zhou, Y. Wang, L. Liu, Z. Tian, An information-theoretic view of WLAN localization error bound in GPS-denied environment. IEEE Trans. Veh. Technol. **68**(4), 4089–4093 (2019)
3. Y. Li, B. Cao, C. Wang, Handover schemes in heterogeneous LTE networks: challenges and opportunities. IEEE Wirel. Commun. **23**(2), 112–117 (2016)
4. B. Awoyemi, B. Maharaj, A. Alfa, Optimal resource allocation solutions for heterogeneous cognitive radio networks. Digit. Commun. Netw. **3**(2), 129–139 (2017)
5. B. Cao, S. Xia, H. Han, Y. Li, A distributed game methodology for crowdsensing in uncertain wireless scenario. IEEE Trans. Mob. Comput. (2019). https://doi.org/10.1109/TMC.2019.2892953
6. K. Dolui, S. K. Datta, in *2017 Global Internet of Things Summit (GIoTS)*. Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing, (Geneva, 2017), pp. 1–6
7. C. Li, P. Liu, C. Zou, F. Sun, J. M. Cioffi, L. Yang, Spectral-efficient cellular communications with coexistent one- and two-hop transmissions. IEEE Trans. Veh. Technol. **65**(8), 6765–6772 (2016)
8. H. Wu, Y. Sun, K. Wolter, Energy-efficient decision making for mobile cloud offloading. IEEE Trans. Cloud Comput. (2018). https://doi.org/10.1109/TCC.2018.2789446
9. M. Zhou, et al., Calibrated data simplification for energy-efficient location sensing in Internet of Things. IEEE Internet Things J. **6**(4), 6125–6133 (2019)
10. B. Cao, et al., When internet of things meets blockchain: Challenges in distributed consensus (2019). https://doi.org/10.1109/JIOT.2018.2869671
11. J. Du, F. R. Yu, X. Chu, J. Feng, G. Lu, Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization. IEEE Trans. Veh. Technol. **68**(2), 1079–1092 (2019)
12. Q. Liu, Z. Su, Y. Hui, in *2018 10th International Conference on Wireless Communications and Signal Processing (WCSP)*. Computation Offloading Scheme to Improve QoE in Vehicular Networks with Mobile Edge Computing, (Hangzhou, 2018), pp. 1–5
13. S. Wu, et al., in *2018 10th International Conference on Wireless Communications and Signal Processing (WCSP)*. An Efficient Offloading Algorithm Based on Support Vector Machine for Mobile Edge Computing in Vehicular Networks, (Hangzhou, 2018), pp. 1–6
14. J. Zhou, F. Wu, K. Zhang, Y.M. Mao, S. Leng, in *2018 10th International Conference on Wireless Communications and Signal Processing (WCSP)*. Joint optimization of Offloading and Resource Allocation in Vehicular Networks with Mobile Edge Computing, (Hangzhou, 2018), pp. 1–6
15. C. You, K. Huang, H. Chae, B. Kim, Energy-efficient resource allocation for mobile-edge computation offloading. IEEE Trans. Wirel. Commun. **16**(3), 1397–1411 (2017)
16. F. Sun, et al., Cooperative task scheduling for computation offloading in vehicular cloud. IEEE Trans. Veh. Technol. **67**(11), 11049–11061 (2018)
17. K. Zhang, Y. Mao, S. Leng, S. Maharjan, Y. Zhang, in *2017 IEEE International Conference on Communications (ICC)*. Optimal delay constrained offloading for vehicular edge computing networks, (Paris, 2017), pp. 1–6
18. Y. Dai, D. Xu, S. Maharjan, Y. Zhang, Joint load balancing and offloading in vehicular edge computing and networks. IEEE Internet Things J. **6**(3), 4377–4387 (2019)
19. H. Guo, J. Zhang, J. Liu, FiWi-enhanced vehicular edge computing networks: Collaborative task offloading. IEEE Veh. Technol. Mag. **14**(1), 45–53 (2019)
20. Z. Wang, Z. Zhong, M. Ni, M. Hu, C. Chang, Bus-based content offloading for vehicular networks. J. Commun. Netw. **19**(3), 250–258 (2017)
21. C. Huang, M. Chiang, D. Dao, W. Su, S. Xu, H. Zhou, V2V data offloading for cellular network based on the software defined network (SDN) inside mobile edge computing (MEC) Architecture. IEEE Access. **6**, 17741–17755 (2018)
22. Z. Wang, Z. Zhong, D. Zhao, M. Ni, Vehicle-based cloudlet relaying for mobile computation offloading. IEEE Trans. Veh. Technol. **67**(11), 11181–11191 (2018)
23. Y. Sun, X. Guo, S. Zhou, Z. Jiang, X. Liu, Z. Niu, *Learning-Based Task Offloading for Vehicular Cloud Computing Systems*, (Kansas City, 2018), pp. 1–7
24. Q. Qi, et al., Knowledge-Driven Service offloading Decision for Vehicular Edge Computing: A Deep Reinforcement Learning Approach. IEEE Trans. Veh. Technol. **68**(5), 4192–4203 (2019)
25. Y. Sun, L. Xu, Y. Tang, W. Zhuang, Traffic offloading for online video service in vehicular networks: a cooperative approach. IEEE Trans. Veh. Technol. **67**(8), 7630–7642 (2018)
26. X. Zhu, Y. Li, D. Jin, J. Lu, Contact-aware optimal resource allocation for mobile data offloading in opportunistic vehicular networks. IEEE Trans. Veh. Technol. **66**(8), 7384–7399 (2017)
27. N. Cheng, N. Lu, N. Zhang, X. Zhang, X. S. Shen, J. W. Mark, Opportunistic WiFi offloading in vehicular environment: a game-theory approach. IEEE Trans. Intell. Transp. Syst. **17**(7), 1944–1955 (2016)
28. C. Li, S. Zhang, P. Liu, F. Sun, J. M. Cioffi, L. Yang, Overhearing protocol design exploiting intercell interference in cooperative green networks. IEEE Trans. Veh. Technol. **65**(1), 441–446 (2016)
29. C. Li, H. J. Yang, F. Sun, J. M. Cioffi, L. Yang, Multiuser overhearing for cooperative two-way multiantenna relays. IEEE Trans. Veh. Technol. **65**(5), 3796–3802 (2016)
30. C. Wang, F. R. Yu, C. Liang, Q. Chen, L. Tang, Joint computation offloading and interference management in wireless cellular networks with mobile edge computing. IEEE Trans. Veh. Technol. **66**(8), 7432–7445 (2017)
31. X. Huang, C. Cao, Y. Li, Q. Chen, Opportunistic resource scheduling for LTE-unlicensed with hybrid communications modes. IEEE Access. **6**, 47857–47869 (2018)
32. C. Yang, Y. Liu, X. Chen, W. Zhong, S. Xie, Efficient mobility-aware task offloading for vehicular edge computing networks. IEEE Access. **7**, 26652–26664 (2019)
33. Z. Xiao, et al., Spectrum resource sharing in heterogeneous vehicular networks: a noncooperative game-theoretic approach With correlated equilibrium. IEEE Trans. Veh. Technol. **67**(10), 9449–9458 (2018)

Publisher's Note