VEC System for Vehicle-to-Vehicle Communication Task Offloading Strategy Research

Xiang Yu, Zheyu Yu(✉), and Chi Zhang

School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing, China
1297154186@qq.com

Abstract. In the context of Vehicle Edge Computing (VEC) systems, optimizing the long-term return of the system is crucial, taking into account factors such as transmission delay, access delay, computing delay, available computing capacity, and the heterogeneity of vehicles and tasks. This paper introduces a task offloading scheme for Vehicle-to-Vehicle (V2V) communication in VEC systems. The proposed approach involves a systematic vehicle screening process based on signal strength. Following the screening, a task offloading model, employing a Semi-Markov Decision Process (SMDP-BOVS) derived from the vehicle screening, is established. An iterative approach utilizing Bellman's equation is then applied to approximate the optimal solution. Extensive simulations demonstrate that the algorithm model presented in this paper effectively enhances the system's sustained reward, meeting the requirements of vehicle task offloading.

Keywords: VEC · Task Offloading · Delay · Signal Strength

1 Introduction

Intelligent Transportation Systems rely significantly on the Internet of Vehicles (IOV), considering it a crucial element in their architecture. Traditional IOV still faces some technical issues in meeting computational and latency demands, such as intelligent information processing technology. To cater to the changing communication and computational requirements of vehicular systems, as well as the needs of Smart devices and people, the VEC system has been established and is now operational.

VEC utilizes connected vehicles as a new type of mobile terminal and deploys computing nodes or servers to provide rapid interactive responses to meet user demands for delay-sensitive tasks. Task offloading is a promising method that can transfer some computationally intensive activities to nearby servers, leveraging the computational capabilities of neighboring vehicles to alleviate the burden on edge servers. Therefore, vehicles can also execute computing tasks through vehicle-to-vehicle (V2V) communication [1]. Task delay is a critical communication metric for computing tasks, including computation delay, access delay, and transmission delay [2, 3]. In task offloading, it directly

F. Gao et al. (Eds.): Chinacom 2023, LNICST 590, pp. 247–262, 2024.
https://doi.org/10.1007/978-3-031-67162-3_16

affects the Quality of Service (QOS) of applications in the context of vehicles and determines whether vehicles can respond in a timely manner to different situations to guarantee the security of users. Furthermore, due to the random arrival of vehicles and offloaded tasks, as well as their temporary and variable nature within the VEC system, computing resources can frequently change.

The Received Signal Strength Indication (RSSI) in the system represents the strength of the received signal, and the size of the RSSI value indicates the distance between vehicles, with larger RSSI values indicating closer proximity. By adjusting the task offloading strategy based on the strength of the currently received RSSI values during vehicle travel, the current channel quality can be assessed.

Given the aforementioned factors, it is imperative to holistically take into account delays, Received Signal Strength Indicators (RSSI), computing resources, vehicle mobility, and task diversity characteristics as long-term rewards in task offloading. Consequently, designing a suitable offloading scheme becomes crucial to optimize the system's cumulative rewards over time.

2 Related Work

In the realm of Vehicle Edge Computing (VEC) in recent years, the optimization strategies for vehicle task offloading can be categorized into three main areas. The first focus is on delay optimization, primarily driven by the stringent delay requirements imposed by the Internet of Vehicles business. Due to the delay of the business, there may be a service that lacks the calculation results of its sub-services, resulting in the service not running normally, thereby degrading the user experience. The second category is optimization for computing resources, which mainly considers how to optimally allocate computing resources in the system. The third category involves optimizing for changes in vehicle states, where task offloading occurs within a system comprising rapidly moving vehicles on the road or vehicles stationed at the roadside.

The on-vehicle edge computing system consolidates the computational resources within a vehicle, offering computing services to other vehicles and pedestrians through the process of task offloading. In the first type of optimization for delay, the researchers in [4] designed an adaptive learning-based task offloading algorithm (ALTO), enabling a vehicle to learn about the offload latency performance of its neighbors while offloading computational tasks, reducing offload latency and improving QOS. Most of the offloading strategies regard the offloading task as a whole. In contrast, researchers in [5] proposed a framework centered on offloading tasks and constructed a finer-grained partition based on offloading tasks. A task offloading scheme is devised wherein each task is conceptualized as a graph without cycles and with directed edges. In this model, each element signifies a secondary task, and each connection signifies the interdependence of data flow between two secondary tasks. Vehicles and Mobile Edge Computing (MEC) servers within the proximity of communication are considered as potential offloading nodes. Subsequently, these computing nodes are allocated to specific subtasks. This method notably improves task performance in terms of execution time and throughput. In reference [6], an intelligent task offloading scheme (DQTF) is proposed, leveraging deep Q-learning to accommodate fast-moving vehicles and swiftly changing communication and computing resources. The scheme incorporates a software-defined network

for efficient collection and centralized management of network information, accelerating information processing, and enhancing the performance of offloaded tasks.

The dynamic nature of computing resources in VEC systems presents challenges for vehicle task offloading. Currently, there is considerable research focus on offloading strategies for computing resource allocation. In one category, as outlined in [7], researchers highlight concerns that relying on static edge server deployment may lead to "service loopholes" in the Internet of Things network. To address this, they propose utilizing vehicles as edge servers, leveraging the constantly changing computing resources. Their method, based on Q-learning and deep reinforcement learning, aims to derive the optimal strategy for task offloading and computing resource allocation. Another approach, presented in [8], introduces a scalable vehicle-assisted Mobile Edge Computing (SVMEC) paradigm. This paradigm not only mitigates resource constraints in MEC but also enhances the scalability of computing services for IoT devices while reducing computing resource costs. Task offload requesters can choose from three options: (1) offloading to local MEC, (2) offloading to the remote cloud, and (3) offloading to nearby vehicles. This is mathematically modeled as a Mixed Integer Nonlinear Programming (MINLP) problem with the aim of optimizing system overhead. In [9], researchers address the challenge of limited resources during peak hours by establishing a pricing model that varies with resource states and a flexible incentive model aligned with user needs. They formulate a UE and Stackelberg game between MEC service providers to determine the optimal strategy for task offloading and the pricing scheme.

Given the rapid movement of vehicles on the road within Vehicle Edge Computing (VEC) systems, researchers are actively exploring strategies to address the swift changes in vehicle entry and departure. In [10], the concept of parking edge computing is introduced, capitalizing on the abundance of resources in underutilized vehicles parked outside urban areas. The proposed approach utilizes parked vehicles to assist edge servers in offloading task processing. A task scheduling algorithm is designed to determine the selection of edge servers and resource allocation, ultimately providing enhanced and consistent offloading events. In the context of high-speed vehicle mobility, as discussed in [11], efforts are made to seamlessly integrate Mobile Edge Computing (MEC) and Internet of Vehicles (IOV) technologies. The introduction of a Cellular-V2X-based MEC offloading scenario enables collaboration between servers and vehicle nodes in 5G networks. For tasks sensitive to delays, a Mobile-Aware Dynamic Offloading Algorithm (MADO) is suggested. This algorithm minimizes the influence of vehicle mobility on offloading by continuously updating resource selection and offloading strategies based on the changing location of the vehicle. In [12], a task offloading scheme leveraging a Takagi–Sugeno fuzzy neural network (T–S FNN) and game theory is devised. The cloud server utilizes a T–S FNN to forecast upcoming traffic patterns on individual road segments, and these predictions are transmitted to the roadside unit (RSU). The RSU modifies the existing resource allocation strategy utilizing predicted traffic information, determining the optimal offloading strategy for users through game theory.

3 System Model

The system model established in this paper, as illustrated in Fig. 1, portrays the behavior of vehicles on the road. Their entry or exit from the VEC system adheres to a Poisson distribution with rates represented by μ_r and μ_l.Within the VEC system, vehicles communicate with each other, enabling them to understand the status of computational resources throughout the entire system.

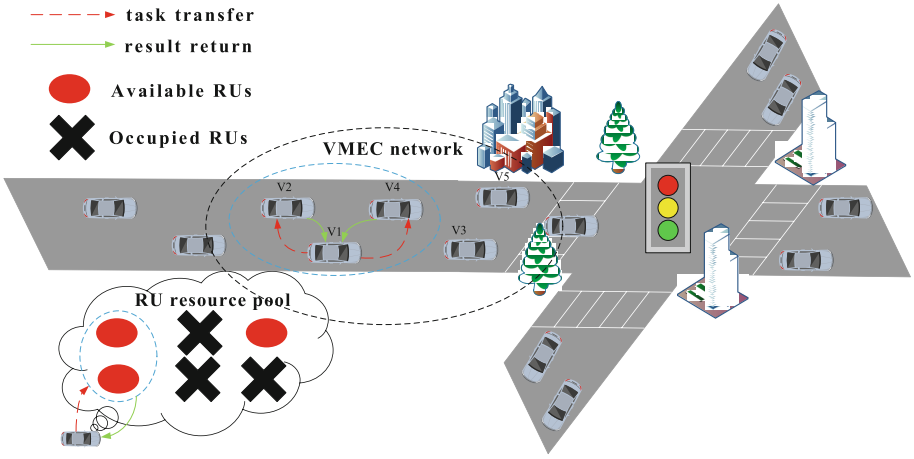

Fig. 1. System model of VMEC task offload

The initiating vehicle filters other vehicles based on their signal strength before sending a request to the system. Each vehicle possesses the same amount of computational resources unit (RU) after virtualization. Through mutual communication, each vehicle in the system is aware of the available RUs. When a requesting vehicle needs to unload a task, it first selects vehicles with RSSI that meets a certain threshold.

Subsequently, the system assesses the task's requirements and decides the appropriate number of RUs to allocate. The requesting vehicle further breaks down the task into evenly sized sub-tasks, transmitting them to the designated CRUs. Finally, the assigned RUs work in tandem to execute the task, providing feedback to the requesting vehicle on the computed results.

3.1 Signal Strength Model

We chose the urban street scene, where vehicle distance and signal propagation environment are important environmental factors affecting signal strength in V2V communication.

Line-Of-Sight (LOS) is a key element to describe the signal propagation environment [13]. A large number of studies regard non-Los and Los as the basic classification of communication environment. In the case of Non-Line-Of-Sight (NLos), due to the transmission signal Multiple reflections and scattering can cause severe degradation of

signal transmission. As in the real city street scene, the situation of Los will change rapidly. Therefore, a fuzzy method is adopted to determine the Loss condition, that is, the traffic area density is used to represent the probability that the signal path is blocked by the intermediate vehicle. In real-world scenarios, traffic density may also be an indication of multipath effects from signal reflections and scattering. The higher the traffic density, the greater the possibility of NLos.

In order to describe different RSSI values mathematically, the well-known large-scale fading model [14] is adopted to describe the RSSI-distance relationship. The prototype of the model is shown in Eq. (1). In this context, p represents the signal power at a specific location. d and d_0 denote the spatial gap and a specific reference position, respectively. θ is the path fading index, and x_σ signifies a stochastic variable conforming to a normal distribution $N(0, \sigma^2)$. The model is simplified as shown in Eq. (2), with model coefficients estimated from the Los characteristics of the vehicle.

$$p(d) = p(d_0) - 10\theta \cdot l_g \cdot (\frac{d}{d_0}) + x_\sigma \tag{1}$$

$$p(d) = p(1) - 10\theta \cdot l_g \cdot (d) + x_\sigma \tag{2}$$

3.2 Delay Model

In this study, we employ a stationary point model to characterize the effectiveness of Dedicated Short Range Communication (DSRC) in V2V communication. We utilize the Full Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) process within the Distributed Coordination Function (DCF) mechanism. Additionally, we make the assumption that the RUs in each vehicle possess identical computing power. The computing service rate of each RU is denoted as μ_t, and the calculation delay for all computing tasks handled by each RU is recorded as

$$T_p(i) = \frac{1}{i \cdot \mu_t} \tag{3}$$

Tasks are partitioned into subtasks of equal size based on the allocated number of individual Computational Resource Units (iCRUs). These subtasks are subsequently transmitted to the assigned individual Resource Units (iRUs). Consequently, the delay time for the task transfer to the iRU is determined as follows:

$$T_A(i) + T_t(i) = i \cdot (D_A(i) + D_t(i)) \tag{4}$$

where D_A(i) and D_t(i) denote the average access delay and transfer delay of transferring a subtask to one of the partitioned iRUs, respectively. The propagation delay is so that D_t(i) can be given by

$$D_t(i) = \frac{E[P]}{R_d} + T_H + \delta \tag{5}$$

In the equation, $E[P]$ stands for the typical length of a packet, R_d denotes the rate of data transfer, T_H signifies the sending time of the packet header, and represents the

transfer delay. In this study, In this paper $\delta = 0$, the $E[P]$ of each requesting vehicle, R_d and T_H are the same, so the transmission delay in the system is a constant value. When requesting a vehicle to start transmitting tasks to a target vehicle in the VEC system, two situations will occur:

1. When the requesting vehicle detects that the channel has been idle for a period of time, it will send a message immediately without fallback processing;
2. If the requesting vehicle detects channel occupancy, the packet will undergo a fallback procedure prior to transmission, and the associated probability is determined by Eq. (6).

In this analysis, we do not account for scenarios in which a previously queued packet, undergoing an extended back-off process, might be preempted by a newly arrived packet. This omission is justified by the deterministic nature of the time interval between consecutive packet arrivals, denoted as ($1/\lambda$ second), where is the packet transmission frequency, ensuring that it is significantly exceeds any potential data packet latency. The DCF employs a back-off scheme in discrete time slots. In cases involving the back-off process, transmission synchronizes with the initiation of a time slot. Consequently, only the second scenario discussed in this article may result in packet conflicts.

$$P_b = (M - 1)\lambda \cdot D_t(i) \cdot [1 - \frac{(n_c - 1)}{n_c} \cdot P_c] \tag{6}$$

In the equation, n_c refers to the mean number of packets involved in a collision incident, and P_c signifies the likelihood of a collision. A collision is triggered when one or more vehicles send packets during the assigned time slot for the target vehicle. Consequently, the likelihood of collisions is expressed by Eq. (7).

$$P_c = P_b[1 - (1 - \rho\pi_0)^{M-1}] \tag{7}$$

Among them, the probability of sending a message in the second case is $\rho\pi_0$, ρ is defined as the probability that the data packet stays in the buffer zone of each vehicle, and π_0 is characterized as the likelihood that a vehicle initiates message transmission when the counter reaches 0, as detailed in Eq. (8), where CW signifies the fixed contention window.

$$\pi_0 = \frac{2}{1 + CW} \tag{8}$$

The total delay encountered by a message scheduled for transmission by the vehicle comprises both the transmission delay D_t(i) and the access delay D_A(i). The access delay is specifically expressed as the duration between the instance when the message reaches the moment when the message arrives at the head of the queue and starts transmitting, it commences transmission within the time interval.

In the initial scenario, the access delay corresponds to one *DIFS*, given that the packet bypasses the backoff process. In the second scenario, the message must await the completion of the ongoing message transmission and subsequently execute the backoff process:

$$D_A(i) = \begin{cases} DIFS, 1 - P_b \\ T_{res} + DIFS + T_{,B}, P_b \end{cases} \tag{9}$$

The remaining time of ongoing message transmission is denoted as T_{res} and T_B represents the backoff time. The process of performing backoff, comprising slots, can be disrupted by transmissions from transmissions of other packets. During these interruptions, the backoff timer is halted. Upon reset, delayed by one DIFS period, it recommences from the beginning of the interrupted slot. The backoff time T_B can be expressed as shown in Eq. (10).

$$T_B = \sum_{n=1}^{L} (\phi + T_I) \tag{10}$$

The average access delay can be determined as:

$$E[D_A(i)] = DIFS + P_b(E[T_B] + E[T_{res}]) \tag{11}$$

Among them, $E[T_B]$ and $E[T_{res}]$ respectively formulate the mean value of the fallback time and the remaining time of the ongoing message transmission, which obey the uniform distribution, so the two can be expressed as:

$$E[T_B] = (\phi + E[T_I]) \cdot E[L] \tag{12}$$

$$E[T_{res}] = \frac{D_A(i)}{2} + DIFS \tag{13}$$

The mean value of the interruption duration T_I can be expressed as

$$E[T_I] = [1 - (1 - \rho\pi_0)^{M-1}] \cdot (D_t(i) + DIFS) \tag{14}$$

In summary, the average delay can be determined, equivalent to the summation of the average access delay, average computation delay and average transmission delay, as shown in Eq. (15).

$$T_A(i) + T_t(i) + T_p(i) = i \cdot (E[D_A(i)] + D_t(i)) + \frac{1}{i \cdot \mu_t} \tag{15}$$

3.3 Offloading Model

In a VEC system, the quantity of available RUs is influenced by factors such as signal strength, and events including vehicle arrivals and departures, as well as task arrivals and completions. Each vehicle can dynamically access the latest information regarding the available RUs in real-time. Upon the initiation of a task request in the system, VEC system determines decisions on the allocation of a specific number of RUs for task processing. This decision is based on considerations such as signal strength, transmission delay, computation delay, access delay, the current availability of RUs, and these varied attributes of both vehicles and tasks. Subsequently, these system receives corresponding rewards.

To address this inquiry, we employ the SMDP-BOVS model. Within the model, the vehicle state is defined as a set in Eq. (16), encompassing the count of tasks assigned with varied quantities of Rus, and the count of RUs in various circumstances. Events are represented by e, with each event categorized within the set $\{A, K_1, ..., K_n, L_{+1}, L_{+1}\}$. A in set represents the initiation of tasks, n signifies the maximum quantity of iRUs that the system can assign, K_i represents the completion of tasks processed by iRUs, L_{+1} represents the entry of vehicles, and L_{-1} represents the completion of vehicles. The condition of e is represented by $s = (R, r_1, ..., r_n, e)$, where is R the quantity of RUs in system, that is, the quantity of vehicles in the VEC system, and r_i represents the count of tasks handled by iRUs.

$$S = \{s | s = (R, r_1, ..., r_n, e)\} \tag{16}$$

The actions taken by the system reflect the quantity of allocated RUs in various scenarios. The action taken is represented as $a(s)$, which is an element of the set $\{0, 1, 2, ..., n\}$. Among them, when $a(s) = i > 0$, it means that the system divides *iRUs* to process the task. In the case of $a(s) = -1$, it means that the system does not take any action. The correlation between these events and the feasible A_{+1} is depicted in Eq. (17).

$$A_{+1} = \begin{cases} \{-1\}, e \in \{K_1, ..., K_n, L_{+1}, L_{-1}\} \\ \{0, 1, 2, ..., n\}, e \in A \end{cases} \tag{17}$$

The rewards obtained by the VEC system are related to the costs and benefits of system expenditures. When the state s takes action a, the system generates revenue $I(S, A)$. s persists for a duration of events until another event arises, leading to the transition of state s to the subsequent state. Throughout this interval, the system incurs a cost denoted as $C(S, A)$. When a is executed in s, the calculated rewards obtained by the system is $R(S, A)$, expressed as the distinction between revenue income and cost, as shown in Eq. (18). Since the state will change with the event, the income will be different in varied actions and events.

$$R(S, A) = C(S, A) - I(S, A) \tag{18}$$

4 Solution

To tackle the previously mentioned challenges, this section employs two algorithms: the vehicle screening algorithm and the iterative algorithm, aiming to optimize the cumulative reward over time within the framework of SMDP-BOVS. In the vehicle screening algorithm, the initial step involves filtering out RUs that meet the RSSI strength criteria. Subsequently, an iterative algorithm is employed to compute the maximum value function across diverse actions and states. This iterative process involves the application of the Bellman equation, iterating until convergence of the maximum value function is achieved. To enhance comprehension of the algorithm introduced in this paper, a detailed description of its workflow will be provided below.

4.1 Vehicle Screening Algorithm

Since the RSSI and the mobility of the vehicle node will affect the vehicle task unloading rate, that is, requesting the vehicle to stay in the system for a short time due to speed or other reasons may cause the task unloading to be forcibly interrupted, etc. Therefore, Selecting the optimal RU for edge computing and discarding task offloading requests for vehicles with too short a dwell time are key to ensure a good user experience.

First, RSSI-based pre-screening is performed on multiple available RUs within the offload decision slot. When task offloading is performed on the selected RU, if its RSSI value is lower than the threshold for smooth task offloading, it may not be guaranteed that the RU can meet the vehicle's task offloading requirements for highly sensitive delays. In this case, the utility value of the RU is set to 0; otherwise, When the RSSI value surpasses the threshold, the utility value of the RU is established as 1, as indicated in Eq. (20).

$$F_{n_nRSSI} = \begin{cases} 1, P_{n_nRSSI} > P_{th} \\ 0, P_{n_nRSSI} \leq P_{th} \end{cases} \tag{19}$$

where T_{n_n} is used to represent the expected residence time of the requesting vehicle in the system, and T_{th} is used to represent the threshold of the residence time. If the residence time is less than the threshold, the unloading request of the requesting vehicle is rejected, as shown in Eq. (21):

$$F_{n_nt} = \begin{cases} 1, T_{n_n} > T_{th} \\ 0, T_{n_n} \leq T_{th} \end{cases} \tag{20}$$

Among them, F_{n_nRSSI} represents the utility function of RU, and $F(n_n)$ represents the screening result of the nth RU. F_{n_nRSSI}, F_{n_nt}, $F(n_n)$ are all 0 or 1, and when the result is 1, the RU will remain in the candidate network of the vehicle node, requesting the vehicle to perform unloading tasks, otherwise it will RU delete, rejecting the unload request of the requesting vehicle. The vehicle screening algorithm (VSA) is presented in Algorithm 1.

Algorithm1. Vehicle Screening Algorithm

input: P_{n_nRSSI}; P_{th};RU scan list $m_M = \{m_1, m_2, m_3, \ldots, m_n\}$;$T_{n_n}$;$T_{th}$
output: The filtered RU list $\widetilde{m}$,the aggregate number of RUs in the system M'
While RU scan list $m_M \neq \phi$, **do**
  For $m_n \in m_M$ **do**
    if $T_{n_n} < T_{th}$ **then**
      $F_{n_nt} = 0$
    else
      $F_{n_nt} = 1$
    end if
    if $P_{n_nRSSI} < P_{th}$
      $F_{n_nRSSI} = 0$
    else
      $F_{n_nRSSI} = 1$
    end if
    $F\left(n_n\right) = F_{n_nRSSI} * F_{n_nt}$
  end for
  if $F\left(n_n\right) = 1$**then**
    $m_n \in \widetilde{m}$
  end if
  $\widetilde{m} \in M'$
end while

4.2 Value Iteration Algorithm

After the vehicle screening algorithm, in each state, the value iteration algorithm is used to calculate the maximum value of the function. At the beginning, the iteration count is set to 0, and each state's function value is set to 0. During the initial iteration, the value for each state is calculated based on the Bellman's optimality equation. Utilizing the initial State-value function, along with the Reinforcement and transition rewards from the preceding k iterations, the maximum function for the k + 1 iteration can be computed.

$$V_{k+1}(s) = \max_{a \in A_C}[R(s,a) + \gamma \sum \sum_{s' \in S} P(s'|s,a) V_k(s')] \tag{21}$$

Here, γ represents a discount factor, employed to decrease the function output of the subsequent state s_i, followed by the normalization of the calculated reward, transition probability, and discount factor. This process facilitates the conversion from discrete time to continuous time. Post-normalization of the dynamic programming equation of Bellman, it can be reformulated as:

$$V_{k+1}(s) = \max_{a \in A_C}[\widehat{R}(s,a) + \gamma \sum_{s' \in S} \widehat{P}(s'|s,a) \widehat{V}_k(s')] \tag{22}$$

During the next iteration of k, the algorithm computes the maximum long-term reward value for each state using Eq. (23). Following the determination of the maximum value for each state, compute the absolute difference between the maximum values across states. If the magnitude of the difference $|\widehat{V}_{k+1}(s') - \widehat{V}_k(s')|$ is below the cutoff point ε, Upon convergence of the iterative algorithm, ω^* represents the actions associated to the highest function value for each set of states, as shown in Eq. (24); on the contrary, If the absolute value falls below the threshold ε, increment the iteration count by 1, and continue to the next iteration until the optimal solution is reached. The iterative algorithm for value computation is detailed in Algorithm 2.

Algorithm2. Iteration Algorithm

1: **Input:** set of system states S;action A_C; reward $R(S,A)$; probability of state transition $P(s'|s,a)$; rate of convergence ε;
2: **Output:**the optimal strategy $\omega^*(s)$
3: Initialization pase: $V(s) = 0, s \in S, k = 0$, List of RUs offloaded in action set $A_C \in \overline{m}$
4: F**or** each state $s \in S$, **do**
5: $\hat{V}_{k+1}(s) = \max_{a \in A_C}[\hat{R}(s,a) + \gamma \sum_{s' \in S} \hat{P}(s'|s,a)\hat{V}_k(s')]$;
6: **if** $\left| \hat{V}_{k+1}(s) - \hat{V}_k(s') \right| < \varepsilon$ **then**
7: $\omega^*(s) = \max_{a \in A_C}[\hat{R}(s,a) + \gamma \sum_{s' \in S} \hat{P}(s'|s,a)\hat{V}_{k+1}(s')]$
8: **else**
9: $k++$
19: **end for**

5 Simulation and Analysis

The efficacy of this method is validated through simulation and comparative experiments. In the simulation, the experimental scene is set in the urban vehicle dense roads with low-speed vehicles, which belong to high density and high probability of causing non-Los situations. The performance of the designed SMDP-BOVS strategy is tested through simulation. The main comparison algorithms are the adaptive learning task offloading algorithm (ALTO), the deep reinforcement learning algorithm (DRL) and the mobile perception dynamic offloading algorithm currently used in the three offloading optimization directions. (MADO). The influence of different algorithms on the experimental results is considered from the three aspects of long-term reward, task offloading rate and task delay. Shown in Table 1 are the simulation parameters used.

Figure 2 illustrates the correlation between task offloading rate and varying vehicle numbers. The figure shows that the task offloading rate will be reduced by the number of vehicles. This trend is attributed to the increase in the number of vehicles leading to an increase in the probability of collision, consequently leading to increased transmission delays. As the task offload rate is inversely proportional to the transmission delay, this results in a reduction in the task offload rate. Notably, as the vehicle count surpasses

Table 1. Parameters of simulation

Parameters	value
Fading index θ	3.3993
Variance σ	5.0854
Slot time ϕ	16 μs
Packet length $E[P]$	400 bytes
Service rate μ_t	25, 50 $task/s$
Data rate R_d	12 Mbps
Propagation delay δ	0
Contention window CW	16
Packet transfer rate λ	2.10 pps
Distributed space $DIFS$	64 μs
Number of vehicles K	4–10
Entry speed μ_r	10 m/s
Departure speed μ_r	10 m/s

8, these tasks offload rate of the SMDP-BOVS exhibits a slower decline in comparison to alternative algorithms. This implies that the algorithm effectively optimizes RU allocation, maximizing the utilization of computing resources in the VEC.

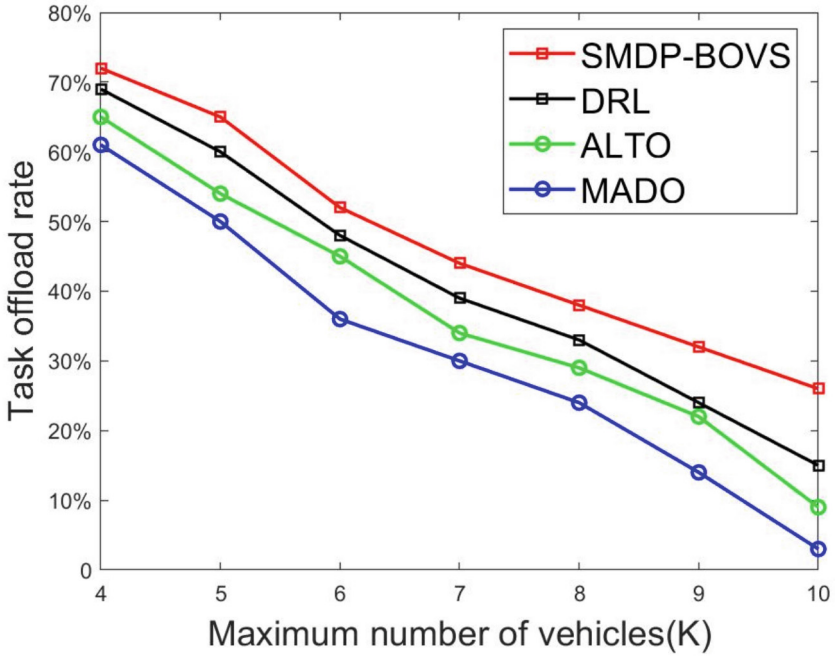

Fig. 2. Task unloading rate for different maximum vehicle

In Figs. 3 and 4, the variations in long-term rewards for the four algorithms with changing vehicle numbers are depicted. As the number of vehicles rises in the system, there is a concurrent rise in the system's rewards over an extended period. In this case of μ_t = 25. In comparison to the ALTO optimization scheme, the SMDP-BOVS-based optimization scheme achieves a 12% optimization rate in long-term rewards with 10 vehicles in the system., because the ALTO scheme requires Frequent communication and data transmission between the two have high requirements on network bandwidth and delay; compared with the optimization scheme based on DRL, the optimization scheme of SMDP-BOVS achieves a long-term reward optimization rate of 8% when there are 10 vehicles in the system. This is because DRL has high requirements for vehicle hardware resources and network bandwidth, which are difficult to meet for vehicles driving on urban streets; compared with the optimization scheme based on MADO, the optimization scheme of SMDP-BOVS has 10 vehicles in the system. A long-term reward optimization rate of 19% has been achieved. This is because MADO needs to accurately perceive and analyze the vehicle environment and user needs, and has high requirements for algorithm robustness and real-time performance. The above two figures respectively calculate the long-term reward when the service rat μ_t is 25 and 50, and the trend of change is the same, but the optimization rate of the SMDP-BOVS strategy in Fig. 3 exhibits a notable increase compared to the values depicted in Fig. 4. This is because compared to Fig. 3, the RU allocated by the system in Fig. 4 handles offloading tasks faster. In this scenario, when $\mu_t = 25$, the system aims to assign a greater number of RUs compared to $\mu_t = 50$. As a result, when $\mu_t = 25$, the optimization scheme based on SMDP-BOVS outperforms $\mu_t = 50$.

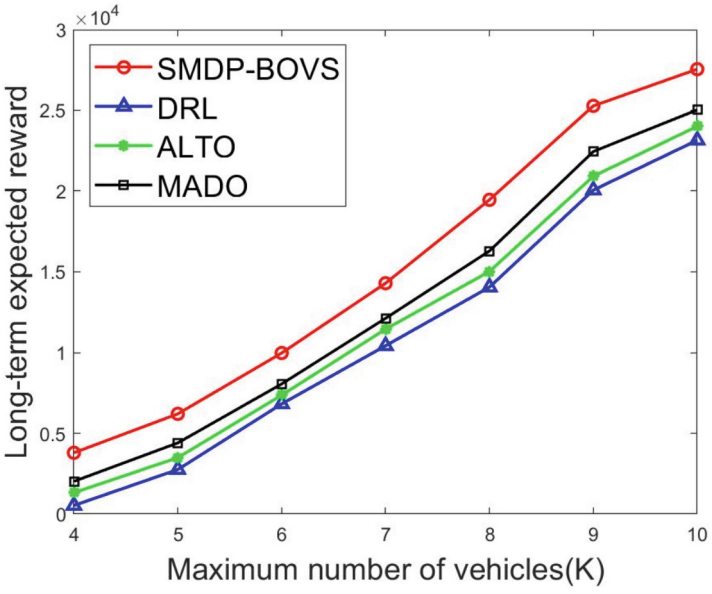

Fig. 3. Long-term rewards corresponding to varying numbers of vehicles ($\mu_t = 25$)

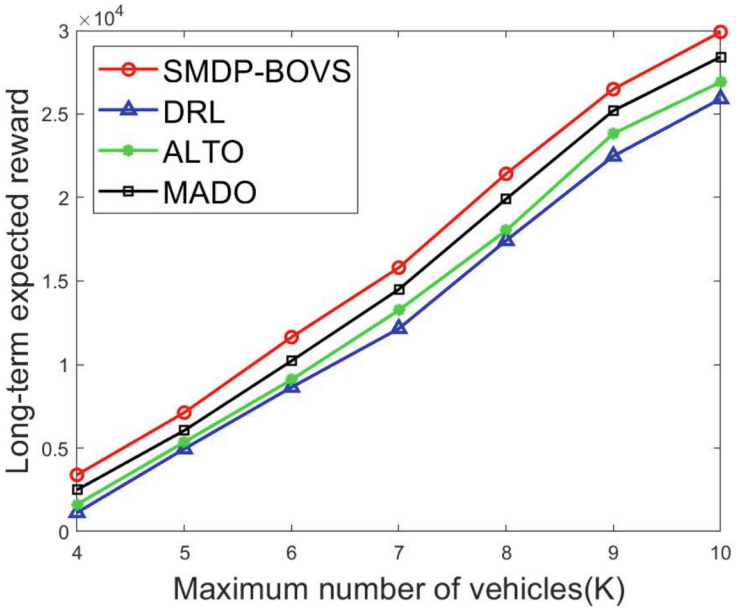

Fig. 4. Long-term rewards corresponding to varying numbers of vehicles ($\mu_t = 50$)

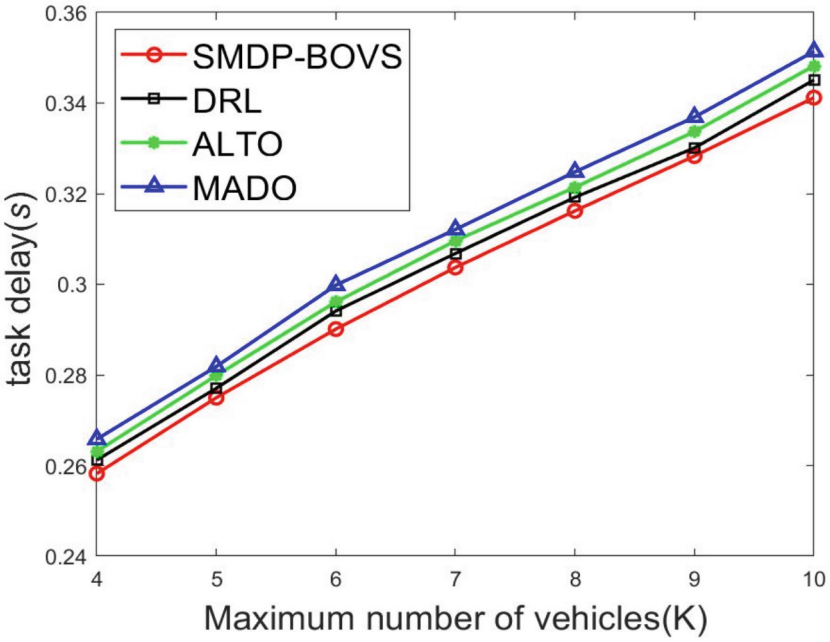

Fig. 5. Task delay for different vehicle ($\mu_t = 25$)

In Figs. 5 and 6, the graphs depict how task offloading delays within the VEC system correlate with the system's operational limit, highlighting variations across different algorithms. As the quantity of vehicles rises, the corresponding delay time also experiences an increase. As the overall quantity of automobiles within the system grows, there is a heightened likelihood of collisions due to the increased demand for task offloading. However, the SMDP-BOVS strategy outperforms the other three algorithms in delay optimization caused by task offloading, because as the number of vehicles escalates, the SMDP-BOVS strategy will comprehensively consider signal strength, transmission delay, access delay and calculation delay Allocate optimal RUs to requesting vehicles, improving task offload latency. The DRL algorithm only considers the lowest delay of unloading tasks within the time T, and does not take into account the departure and arrival of vehicles in the system, so it is difficult to achieve the expected effect; both the ALTO algorithm and the MADO algorithm will make a task unloading decision in each time slot, The ALTO algorithm allocates as many RUs as possible to the vehicle, so that the increased transmission delay may affect the delay of task unloading; the MADO algorithm will judge the unloading decision based on the vehicle's traffic conditions and road congestion, and it can quickly perform tasks in urban street scenes. Changing traffic conditions can affect the accuracy of the algorithm. Figure 5 and Fig. 6 respectively show that when the RU service rate is 25 and 50, the change trends of the two are the same. But the task offloading delay of the algorithm in Fig. 5 is inferior to that in Fig. 6, because when the RU service rate increases, the calculation delay is lower than the transmission delay, so the delay times will be reduced.

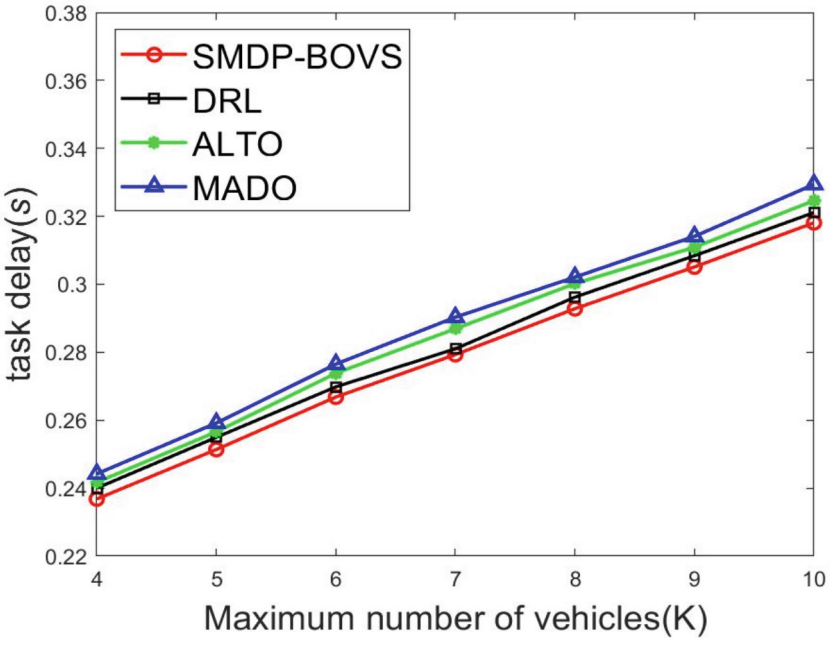

Fig. 6. Task delay for different vehicle ($\mu_t = 50$)

6 Conclusions

Aiming at the vehicle task offloading requirements in the VMEC system, this paper considers a variety of factors, including transmission delay, access delay, calculation delay, signal strength, RU quantity and usage, and the diversity characteristics of vehicles and tasks, etc., and designs a A SMDP-BOVS model. Simultaneously, to fulfill the signal strength prerequisites for on-board tasks, a vehicle screening algorithm is designed in this paper to select service vehicles. Finally, by solving the Bellman equation, the optimal solution for maximizing the system's long-term reward is obtained. In future work, the collaborative work of smart devices around the road and vehicles can be further considered to complete vehicle unloading tasks in a more efficient manner.

References

1. Sodhro, A.H., Obaidat, M.S., Abbasi, Q.H., et al.: Quality of service optimization in an IoT-driven intelligent transportation system. IEEE Wirel. Commun. **26**(6), 10–17 (2019)
2. Hussain, S.M., Yusof, K.M., Hussain, S.A., Khan, A.B.: An efficient interface selection scheme (DSRC/LTE) of vehicles for data dissemination enabling V2V communication to support internet of vehicles (IOV). In: Reddy, V.S., Prasad, V.K., Wang, J., Reddy, K.T.V. (eds.) Soft Computing and Signal Processing. AISC, vol. 1340, pp. 573–581. Springer, Singapore (2022). https://doi.org/10.1007/978-981-16-1249-7_54
3. Cao, L., Yin, H., Hu, J., et al.: Performance analysis and improvement on DSRC application for V2V communication. In: 2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall), pp. 1–6. IEEE (2020)
4. Sun, Y., Guo, X., Song, J., et al.: Adaptive learning-based task offloading for vehicular edge computing systems. IEEE Trans. Veh. Technol. **68**(4), 3061–3074 (2019)
5. Wu, L., Zhang, R., Li, Q., et al.: A mobile edge computing-based applications execution framework for internet of vehicles. Front. Comp. Sci. **16**(5), 165506 (2022)
6. Guo, H., Liu, J., Ren, J., et al.: Intelligent task offloading in vehicular edge computing networks. IEEE Wirel. Commun. **27**(4), 126–132 (2020)
7. Liu, Y., Yu, H., Xie, S., et al.: Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks. IEEE Trans. Veh. Technol. **68**(11), 11158–11168 (2019)
8. Pham, X.Q., Nguyen, T.D., Nguyen, V.D., et al.: Joint node selection and resource allocation for task offloading in scalable vehicle-assisted multi-access edge computing. Symmetry **11**(1), 58 (2019)
9. Xue, J., Hu, Q., An, Y., et al.: Joint task offloading and resource allocation in vehicle-assisted multi-access edge computing. Comput. Commun. **177**, 77–85 (2021)
10. Ma, C., Zhu, J., Liu, M., et al.: Parking edge computing: parked-vehicle-assisted task offloading for urban VANETs. IEEE Internet Things J. **8**(11), 9344–9358 (2021)
11. Li, B., Chen, F., Peng, Z., et al.: Mobility-aware dynamic offloading strategy for C-V2X under multi-access edge computing. Physical Communication **49**, 101446 (2021)
12. Shen, X., Chang, Z., Niu, S.: Mobile edge computing task offloading strategy based on parking cooperation in the internet of vehicles. Sensors **22**(13), 4959 (2022)
13. Farzamiyan, A.H., Gaitán, M.G., Sámano-Robles, R.: A multi-ray analysis of LOS V2V links for multiple antennas with ground reflection. In: 2020 AEIT International Annual Conference (AEIT), pp. 1–6. IEEE (2020)
14. Demir, Ö.T., Björnson, E.: Large-scale fading precoding for maximizing the product of SINRs. In: ICASSP 2020–2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5150–5154. IEEE (2020)