ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
Received: 6 August 2025; Accepted: 8 October 2025

https://doi.org/10.17559/TV-20250806002885
Original scientific paper

Improved PSO-Based Task Offloading Model for Internet of Vehicles Edge Computing

ZhiXiong JIN

Abstract: The demand for computer resources for internet of vehicles services like autonomous driving, real-time navigation, and in-vehicle entertainment has grown rapidly due to the widespread deployment of intelligent transportation systems and the ongoing advancement of information and communication technologies. Therefore, a novel task offloading optimization allocation model for internet of vehicles edge computing is proposed. The model is based on mobile edge computing architecture. Through clustering algorithm, it intelligently clusters all nodes in the static parked vehicles edge computing architecture. Moreover, the PSO algorithm is coded and optimized, which improves the efficiency and resource utilization of internet of vehicles task offloading. The experimental results indicated that the model was able to realize obvious inter-cluster separation under 2 min, 10 min, 50 min, and 100 min time nodes. The vehicles inside the clusters were also more closely distributed, resulting in good internal consistency and external separation. When the number of tasks was increased to 60, the corresponding total system cost of the research model was only 198. When the task computation volume was 120 GHZ, the total system cost of the research model was only 214. In addition, the research model still maintained a high offloading success rate of 97.5%, 94.6%, and 92.8 in low-density, medium-density, and high-density environments. In summary, the research model not only can effectively improve the vehicle task processing efficiency and reduce the system overhead, but also shows strong adaptability and robustness, which has good prospects for practical applications.

Keywords: internet of vehicles; K-means; mobile edge computing; particle swarm optimization algorithm; task offloading

1 INTRODUCTION

With the rapid development of the new generation of information and communication technologies, the Internet of Vehicles (IoV) has gradually become an important component of intelligent transportation systems. IoV is able to realize the real-time interaction and sharing of various dynamic information in the road traffic environment through various communication modes such as vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I), and vehicle-to-network (V2N). Furthermore, this provides data support for applications such as autonomous driving, intelligent navigation, and remote diagnosis [1, 2]. However, with the sharp increase in the number of IoV terminal devices and the continuous complexity of application requirements, the amount of data that the system needs to process in real time has grown explosively. According to relevant predictions, self-driving vehicles can generate hundreds of gigabytes of data per hour during operation. Although traditional cloud computing has significant advantages in computing and storage capabilities, its centralized architecture leads to high latency and bandwidth occupation caused by long-distance data transmission, thus making it difficult to meet the strict requirements of IoV for low latency, strong computing power and high reliability [3]. To break through the above-mentioned bottlenecks, Mobile Edge Computing (MEC) has been introduced into the IoV environment. By sinking computing resources to the edge nodes of the network, MEC enables IoV data to be processed rapidly near the data source, thereby significantly reducing data transmission latency and bandwidth occupation, and enhancing system response speed and security [4]. Although MEC shows great potential in IoV, how to achieve reasonable and efficient offloading of tasks between vehicles and edge servers under the condition of limited edge computing resources remains the core issue restricting the performance improvement of IoV systems. If the unloading decision is unreasonable, it may lead to an imbalance in the distribution of computing resources, an increase in task execution delay, and even trigger security risks. Therefore, the task offloading problem in the IoV environment has gradually become a research focus of attention in both the academic and industrial fields.

2 RELATED WORKS

Aiming at the task offloading problem in high mobility IoV environments, Alam M. Z. et al. proposed a decentralized vehicle-assisted multi-access MEC network. Experimental results indicated that this network effectively overcame the limitation of deep reinforcement learning-based task offloading (DRL-TO), which is insufficiently robust in high mobility environments due to ignoring the independent actions of other agents during the training process [5]. Yang S. et al. proposed a fault-tolerant address event representation for spike information routing. According to experimental findings, the technique could successfully enhance the system's navigation fault-tolerant performance and achieve precise obstacle avoidance for edge IoV computing [6]. In addition, Jia Y. et al. suggested a multi-agent proximal policy optimization approach and structured the IoV task offloading problem as a time-averaged optimization problem with long-term constraints. Experimental results indicated that the algorithm outperformed other baseline algorithms in metrics such as average latency and task completion rate under different traffic load and MEC resource conditions [7]. Traditional MEC algorithms usually assume that the execution time of the resource allocation process is known. However, in real IoV environments, it is often difficult to accurately estimate the execution time of the edge servers (ESs) due to complex factors such as device heterogeneity and dynamic changes in the network. To solve this problem, Sun F. et al. proposed a joint learning-based optimal resource allocation method. Experimental results indicated that the method could achieve better resource utilization and task completion time (TCT) under different network topologies and task load conditions. This further improved the overall performance and adaptability of MEC systems in IoV environments [8].

Particle swarm optimization (PSO) is a population smart optimization algorithm. It has been widely used in IoV task offloading decision making problems in recent

years due to its simple implementation, fast convergence and global search capability [9, 10]. For example, Jamalzadeh M. et al. proposed an edge computing-assisted clustering routing algorithm based on multi-objective PSO for IoV applications. The experimental results indicated that this method had significant advantages over similar methods [11]. The computing and storage resources of MEC servers and the bandwidth of mobile networks are limited, making it impossible to offload all mobile computing tasks to MEC servers for processing. For this reason, Dong S et al. expressed the task offloading problem as an optimization problem and proposed a task offloading strategy based on PSO and quantum particle swarm optimization. The experimental results show that this algorithm can significantly reduce the energy consumption, task completion time and running time of the IoV system [12]. With the development of automobiles towards fully autonomous driving, IoV devices need to transmit massive amounts of data to the cloud and road side unit (RSU) in real time. However, frequent data transmission is very likely to trigger quality of service degradation, which in turn affects the vehicle user experience. Therefore, Sami H et al. used UAVs and RSUs to work together to achieve precise positioning of vehicles using PSO algorithms, which improved the data transmission efficiency and stability of the autonomous driving system [13]. To increase the effectiveness and security of IoV, safe protocols and effective routing methods are crucial because IoV incorporates real-time data. Sharma S et al. further explored the potential application of PSO algorithm to optimize the overall performance in IoV networks. The experimental results indicated that by introducing the PSO optimization mechanism, the IoV system was significantly enhanced in terms of routing efficiency, load balancing and communication security. The method effectively enhanced the quality of service and system robustness in the IoV environment [14].

In summary, existing research has made many advances in IoV edge computing and task offloading, but there are still problems such as poorly targeted algorithms, insufficient task allocation accuracy, and limited dynamic adaptation capability. Moreover, in computationally intensive task scenarios, IoV is prone to task execution delays, which leads to safety hazards during driving and reduces user experience. In view of this, the study first innovatively introduces the K-means clustering algorithm, which intelligently clusters all nodes in the edge computing architecture of statically parked vehicles to optimize the spatial distribution of computing resources. Subsequently, the PSO algorithm is coded and optimized for the actual high dynamics and heterogeneity problems in the IoV environment, and a task offloading optimization allocation model for edge computing based on improved PSO is further proposed. Through this new model, the research aims to enhance the accuracy and real-time performance of task offloading decisions in the IoV environment, reduce computing latency and system energy consumption, thereby effectively improving the overall performance of the IoV system and ensuring the safety of vehicle operation and user experience. The main contributions of the research are as follows:

(1) Optimization of resource distribution through static vehicle clustering: The K-means clustering algorithm is utilized to group static vehicles as edge computing nodes, thereby enhancing the rationality of resource spatial distribution.

(2) Improve PSO to adapt to dynamic environments: Propose a ring topology PSO algorithm to effectively avoid premature convergence and enhance adaptability in heterogeneous IoV environments.

(3) Comprehensive performance evaluation: The model was tested under different traffic density, task load and vehicle movement conditions. The results showed that it outperformed the existing advanced methods in terms of task completion delay, energy consumption and unloading success rate.

The remainder of this paper is organized as follows: Section 2 reviews related work on IoV task offloading and PSO variants. Section 3 introduces the system model, including K-means clustering and the improved PSO algorithm. Section 4 describes the simulation setup and presents results and discussion. Section 5 concludes with implications and future work.

3 METHODS AND MATERIALS

3.1 Vehicle Clustering Model Based on K-means with Edge Computing

In major cities at home and abroad, the number of parked vehicles on both sides of roads is usually much higher than the number of parked vehicles in centralized parking lots [15]. This distribution characteristic indicates that parked vehicles in a stationary state not only have stable physical location characteristics, but also have reliable availability of computing resources. Therefore, they can be integrated into the edge computing architecture as distributed computing nodes, thus significantly reducing the computational load pressure on ES clusters and remote cloud servers. Vehicles traveling on the road can also provide computing resources though. However, their mobility is affected by road conditions and driving states, resulting in significant fluctuations in the connection duration of MEC servers [16, 17]. This dynamic instability makes it difficult to guarantee the sharing of computing resources. Therefore, the study used the traveling vehicle as a secondary computational resource. If RSUs exist on the roadside, the static parked vehicles are considered as edge computing nodes along with the RSUs into the system architecture. The edge computing system (ECS) architecture is shown in Fig. 1.

In Fig. 1, the ECS architecture mainly consists of ESs, base station devices, static parked vehicle nodes, and mobile vehicle nodes (moving vehicles). The ES is deployed near the base station and is responsible for real-time processing of data from mobile vehicles and static parked vehicles. The static parked vehicles are responsible for sharing the computational load of the ES. The mobile vehicles communicate with each other through V2V to realize information exchange and cooperative computation, which further improves the resource utilization efficiency and task response speed of the overall network. The base station maintains a stable connection with each node through wireless communication. If necessary, it can be combined with RSU to further expand the system coverage and computing resource pool, so as to build a cooperative, efficient, and dynamically adapted edge computing

environment. To address the problem of insufficient ES resources, neighboring static parked vehicles and dynamic moving vehicles can be organized into vehicle clusters based on the road topology.

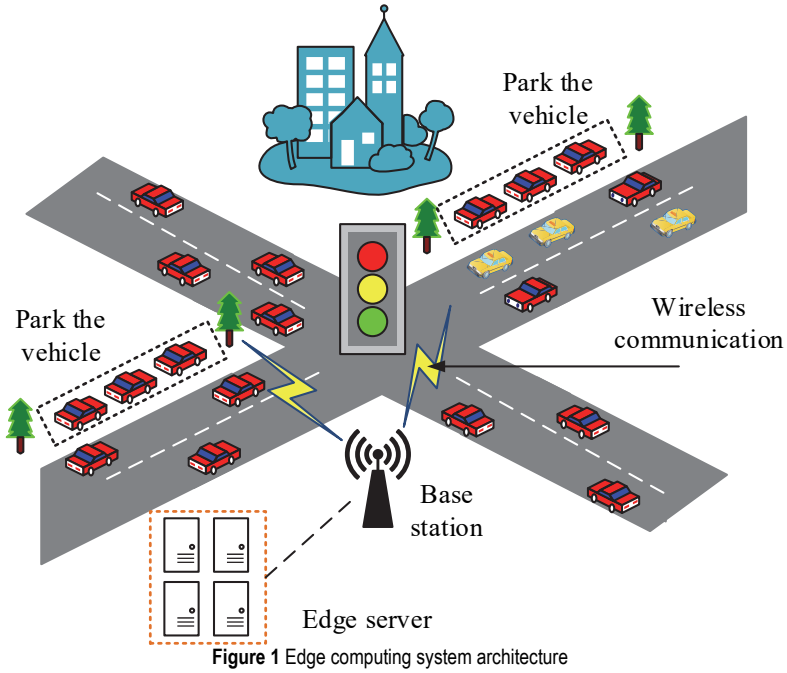


Figure 1 Edge computing system architecture

Most of the existing methods usually divide the cluster management process into two phases, i.e., cluster generation and cluster maintenance [18]. It also defaults to keep the mobile nodes stationary during the cluster generation process. The study combines this idea with the K-means algorithm for clustering static parked vehicles as well as mobile vehicles and proposes a vehicle clustering model based on K-means and edge computing. An unsupervised machine learning technique called the K-means algorithm can minimize the squared error between data points (DPs) inside clusters and split data samples into K clusters according to feature similarity [19, 20]. Fig. 2 depicts the vehicle clustering model process's structure.

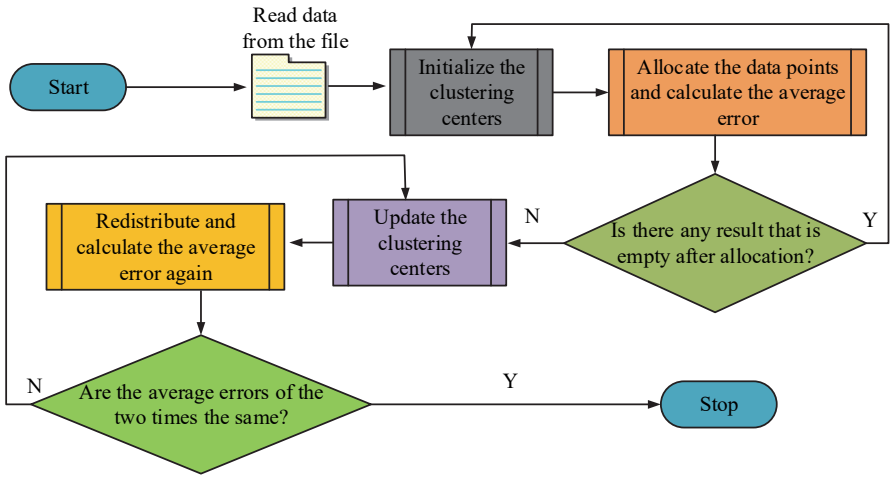


Figure 2 Schematic diagram of the process structure of vehicle clustering model

In Fig. 2, firstly, the required data are read from the specified file and K clustering centers are randomly initialized as the initial cluster representatives. Subsequently, each DP is assigned to the nearest cluster center based on the Euclidean distance, and the average error of the current division is calculated at the same time. After completing the data allocation, it is necessary to check whether there are empty clusters. If present, the cluster centers are reinitialized and adjusted to ensure that each cluster contains DPs. If there are no empty clusters, the clustering center is updated based on the mean value of the DPs within each cluster. After that, the data allocation is redone and the new mean error is calculated based on the updated clustering center. The cycle of iteration is repeated and the average errors of the two times before and after are compared after each iteration. When the average error of the two times reaches the preset convergence conditions (i.e., the error change tends to stabilize or is less than a certain threshold), the algorithm terminates and outputs the final clustering results. The vehicle clustering model cluster formation and vehicle node distribution structure are shown in Fig. 3.

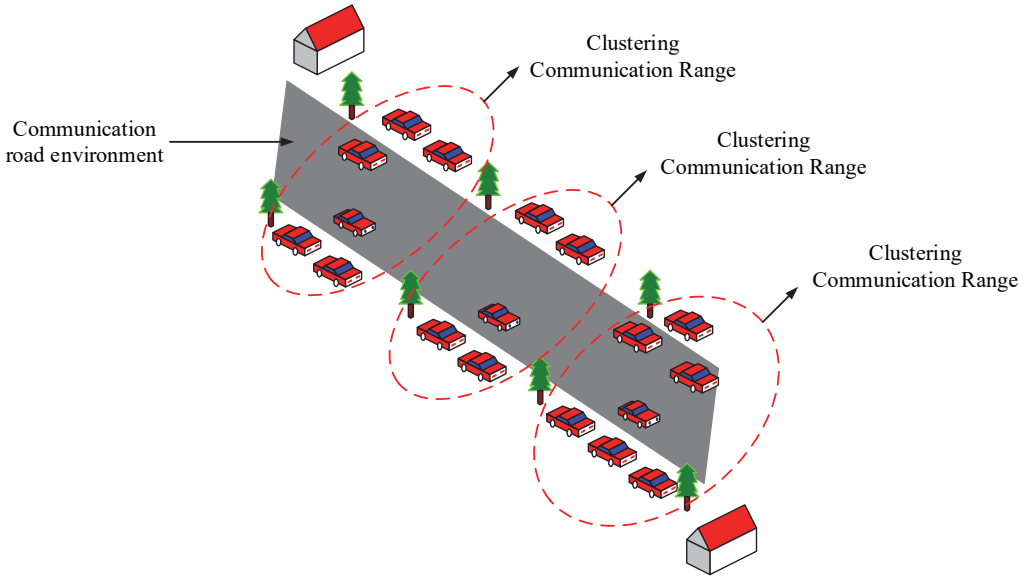


Figure 3 Schematic diagram of vehicle clustering

In Fig. 3, the clustering process usually includes two phases: cluster head node selection and cluster member selection. The cluster head, as a special node, is responsible for bandwidth allocation and member coordination. At least one cluster head needs to be configured in each cluster to ensure that the basic functions are realized. Cluster members are ordinary nodes, mainly responsible for uploading local messages to the cluster head and receiving broadcast messages from the cluster head. At the same time, they need to report their status (e.g., remaining energy, communication quality, etc.) regularly so that the cluster head can grasp the dynamics of the nodes in the cluster in real time. In terms of function positioning, the cluster head not only needs to effectively manage the cluster members and maintain the stability and efficiency of the cluster structure, but also needs to reasonably allocate the relevant resources. After determining the cluster head, the cluster members are selected from the one-hop neighbors of the cluster head to ensure the reliability of intra-cluster communication based on the criteria of communication quality and distance. When the cluster is formed, it then enters the cluster maintenance phase to maintain the stability of the cluster and provide security for subsequent data transmission between vehicles. Cluster members are required to interact with the cluster head on a regular basis for real-time attribute information. If the cluster head loses connection or is abnormal, the system will immediately and automatically start the reclusterization process. The continuity of service is guaranteed by quickly reconfiguring the network topology, thus providing reliable support for real-time data transmission in IoV environment. Eq. (1) illustrates how the clustering effect is measured in the study using sum of squared errors (*SSE*).

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} \left\| x - \mu_i \right\|^2 \tag{1}$$

In Eq. (1), K represents the number of clusters, i.e., the number of clusters after clustering. C_i represents the set of DPs in the i-th cluster. x represents any DP within the cluster. μ_i is the center of the i-th cluster. The SSE measure is the sum of squares of the distances from the DPs within each cluster to the center of the respective cluster [21]. The smaller the SSE, the closer the DPs within the same cluster are to the center and the better the clustering is [22]. The Euclidean distance formula is shown in Eq. (2).

$$d(x,y) = \sqrt{\sum_{j=1}^{n} (x_j - y_j)^2} \tag{2}$$

In Eq. (2), $d(x,y)$ represents the Euclidean distance between DPs x and y. n represents the quantity of feature dimensions. x_j and y_j then represent the values of x and y in the j-th dimension. The clustering center update formula is shown in Eq. (3).

$$\hat{\mu}_i = \frac{1}{|C_i|_{x \in C_i}} \sum x \tag{3}$$

In Eq. (3), $\hat{\mu}_i$ is the new center of the i-th cluster. In the vehicle clustering process, in order to measure the quality of intra-cluster communication, the study introduces the average communication distance between nodes. The average communication distance $D_{\text{avg}}(C_i)$ is calculated as shown in Eq. (4).

$$D_{\text{avg}}(C_i) = \frac{2}{|C_i|(|C_i|-1|)} \sum_{x,y \in C_i, x \neq y} d(x,y) \tag{4}$$

A cluster's structure is more compact and its communication efficiency is higher when the average distance between its nodes is smaller [23, 24]. The cluster head selection weight function W_i is calculated as shown in Eq. (5).

$$W_i = \alpha \times \frac{E_i}{E_{\max}} + \beta \times \frac{S_i}{S_{\max}} + \gamma \times \frac{D_{\min}}{D_i} \tag{5}$$

In Eq. (5), α, β, and γ are the weighting coefficients of each indicator. E_i and $E_{\max}$ represent the residual energy and maximum residual energy of node i, respectively. S_i and $S_{\max}$ represent the social influence metrics and maximum social influence of node i, respectively. D_i and $D_{\max}$ represent the average distance from node i to cluster members and the global minimum average distance, respectively.

3.2 Improved PSO-Based Optimal Allocation Model for Task Offloading in Edge Computing

The vehicle clustering model based on K-means with edge computing is able to rationalize the intelligent clustering of all nodes in the edge computing architecture for static parked vehicles. To further optimize the overall task scheduling, the study combines PSO on top of vehicle clustering model. It finally proposes an optimized allocation model for edge computing task offloading based on improved PSO. Kennedy and Eberhart proposed PSO in 1995. Inspired by the foraging behavior of bird flocks, the algorithm simulates the cooperation and information exchange of people in the search space to identify the best solution to the problem step by step [25, 26]. When applying PSO to the task offloading problem, the coding method needs to be reasonably designed. It makes the task allocation scheme closely correspond to the position, speed, and priority information of the particles to improve the search efficiency and accelerate the convergence process of the optimal solution (OS) [27, 28].

The study uses a two-dimensional floating-point encoding for the representation of PSO particles. In this case, the integer part corresponds to the specific task number, and the fractional part assumes a dual function. The first decimal is used to determine the offloading location of the target task (set to three optional offloading nodes). The second decimal defines the execution priority of the task. Larger values indicate higher priority. In response to the problem that high-speed vehicle movement may lead to out of the original intra-cluster communication range, which in turn affects the delivery and processing of unfinished tasks. The study further introduces vehicle speed into the priority rating. Faster vehicle tasks should be scheduled first to lower the chance of communication breakdowns, prevent traffic jams and accidents, and prevent other negative outcomes. The decoding method is shown in Fig. 4.

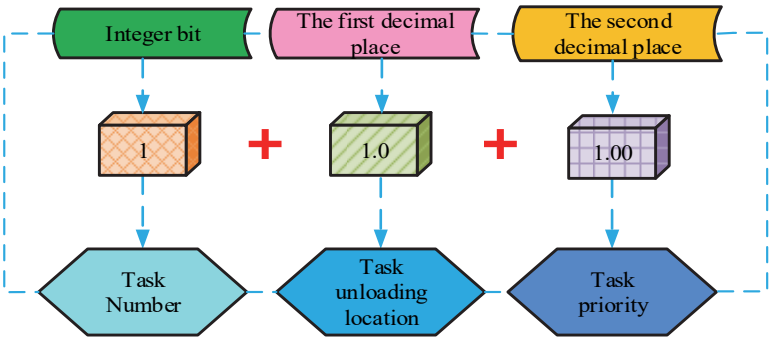


Figure 4 Decoding method

In Fig. 4, the decoding process maps the integer part and the fractional part to the task number and the offloading location by reverse parsing the floating-point encoding, respectively. This enables the correspondence between the optimization results and the executable task allocation scheme, which in turn determines the final offloading strategy. The work uses a ring topology to build the local neighborhood in an attempt to solve the issue of PSO's propensity for early convergence. The ring topology is shown in Fig. 5.

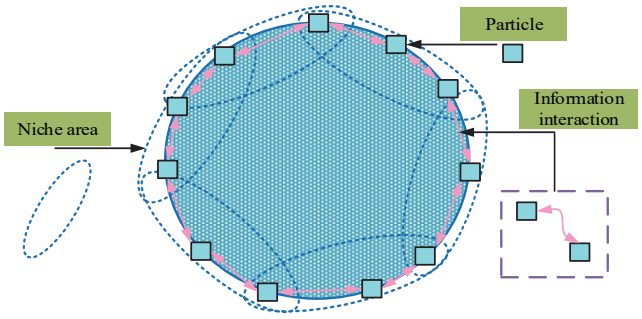


Figure 5 Ring topology structure

In Fig. 5, the ring topology is able to promote population diversity and evolution by restricting particles to interact with neighboring individuals only for information [29, 30]. The structure has self-organizing properties and can naturally form multiple relatively independent niche regions without additional parameters such as niche radius. The directional migration of particles and the sharing of OS information sustain the dynamic link between these regions, successfully preventing the PSO algorithm's early convergence issue. This makes it possible for the algorithm to better balance local exploitation with global exploration. Unlike prior PSO-based offloading methods (e.g., Dong S. et al. 2022), our improved PSO introduces a two-dimensional floating-point encoding that

jointly represents task location and priority while incorporating a vehicle speed factor for dynamic adjustment, and further adopts a ring-topology neighborhood structure for information exchange. This ensures accurate mapping of heterogeneous IoV tasks, robustness against task interruptions in highly dynamic environments, and a better balance between global exploration and local exploitation to avoid premature convergence. The maximal fitness value is the PSO algorithm's optimization goal [31, 32]. The formula for calculating the fitness function $f(x)$ is shown in Eq. (6).

$$f(x)=\frac{1}{1+\lambda\cdot y+\omega\cdot E} \tag{6}$$

In Eq. (6), λ and ω stand in for the weights of delay y and EC E, respectively. The inertia weight calculation formula is shown in Eq. (7).

$$\omega'=\omega_s-(\omega_s-\omega_e)\times\frac{1}{1-e^{a-bg}} \tag{7}$$

In Eq. (7), ω_s and ω_e represent the initial inertia weights and initial inertia weights, respectively. a and b represent the Sigmoid function translation parameter and decay rate coefficient, respectively. g represents the current iteration number. The structural flow of the improved PSO algorithm is as follows. First, basic parameters like population size and inertia weights are set after the parameters are initialized. A two-dimensional floating-point vector is used to encode the representation of the position information of each particle to adapt to the characteristics of the task offloading problem. Next, the objective function value of each particle in the population in the current iteration cycle is calculated. It also initializes its individual historical OS and the global OS of the whole population, respectively, to serve as the reference basis for subsequent iteration updates. Subsequently, the ring topology is used to construct the particle neighborhood. By adding the mutation operation, the algorithm improves the capacity to jump out of the local optimal by combining the global OS, the OS in the neighborhood, and the individual OS. Moreover, the effectiveness of the solution is ensured by the speed limit formula and task offloading constraints. After each update, the fitness values of the particles are re-evaluated to determine whether the individual or global OS records need to be updated. The process is executed continuously and iteratively until the preset termination conditions are satisfied. The final output is the optimal task offloading scheme after decoding process. The pseudo-code of the improved PSO algorithm is shown in Tab. 1.

Table 1 Pseudo-code

Input: Task set T, Vehicle clusters C, PSO parameters
1: Initialize particle positions and velocities based on task assignment
2: for each iteration do
3: for each particle do
4: Evaluate fitness considering delay, energy, success rate
5: Update personal and neighborhood best positions
6: Apply ring-topology information exchange
7: Apply mutation strategy if stagnation detected
8: end for
9: Handle vehicle mobility: reassign tasks from leaving vehicles
10: end for
Output: Optimized task-to-vehicle assignment

The final model task allocation process is shown in Fig. 6.

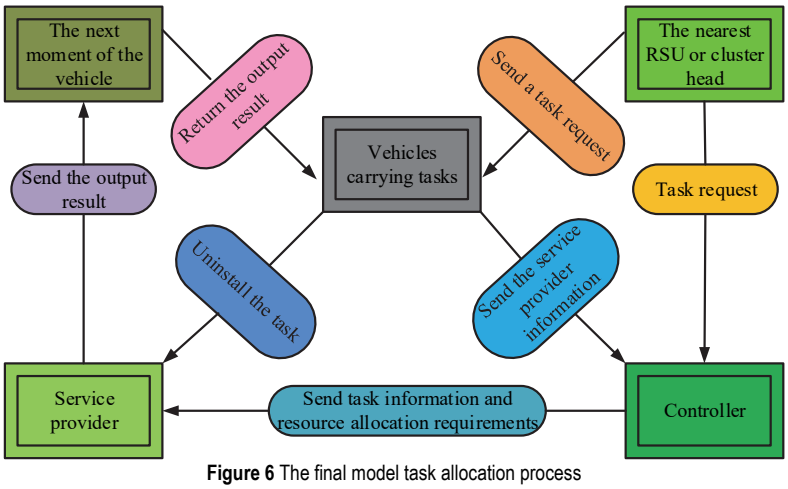


Figure 6 The final model task allocation process

In Fig. 6, first, the vehicle carrying the task sends a task request to the nearest RSU or cluster head. The request is then passed to the vehicular edge computing (VEC) controller for centralized processing. Based on the current system resource status and task characteristics, the VEC controller filters the suitable service providers and sends their information back to the vehicle. After receiving the candidate service provider information, the vehicle selects the appropriate service node and submits specific task information and resource allocation requirements. The service provider executes the processing operation after receiving the task and completes the task offloading by means of the improved PSO algorithm. After the task processing is completed, the service provider returns the output to the vehicle, which then uses the output in the next time slice. When generating a task, in addition to executing it locally or on the VEC controller, statically parked vehicles can be included in the pool of computational resources. This allows the task node to prioritize the most appropriate execution location among the three computing nodes based on execution latency and EC. The final model task offloading prioritization strategy is shown in Fig. 7.

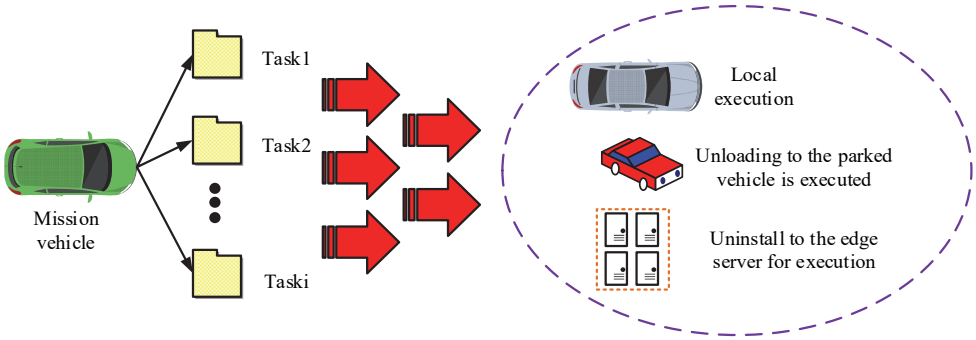


Figure 7 Final model task offloading priority strategy

In Fig. 7, in the task execution delay section, when the local execution mode is selected, the time delay of the task is the sum of the local task computation time and waiting time. In the V2V network architecture, vehicle nodes transmit sensory data via V2V to idle vehicle nodes in the network and these nodes complete the computation and analysis and transmit the results back. Eq. (8) provides the formula for determining the information transfer rate $C(P)$ for the communication between the vehicle unit and the neighboring vehicle units.

$$C(P) = \omega_v \log_2 (1 + \frac{p_a^2 (\frac{1}{l_o})^{p_b P}}{\omega_v \rho_o}) \tag{8}$$

In Eq. (8), ω_v represents the channel bandwidth in V2V communication. ρ_o represents the Gaussian noise power spectral density. p_a and p_a represent the channel fading coefficient and path loss coefficient in the upload link, respectively. l_o is the straight line distance between the vehicle and the nearby vehicles. The vehicle transmits its computational task to the nearby RSU with the help of V2R communication. The corresponding task is then processed by the edge computing server connected to the RSU. Eq. (9) provides the information transfer rate $C'(P)$ for the communication between the vehicle and the ES.

$$C'(P) = \omega_r \log_2 (1 + \frac{p_a^2 (\frac{1}{l_1})^{p_b P}}{\omega_r \rho_o}) \tag{9}$$

In Eq. (9), ω_r represents the channel bandwidth in V2R communication. l_1 represents the spatial linear distance between the vehicle and the edge computing node. Once the calculation is finished, the ES uses RSU to send the results back to the car device. When a task is assigned to a neighboring stationary vehicle for processing, its overall EC consists of three components: the EC for transmitting the task to the target in-vehicle device, the EC of the task during its transit in the in-vehicle device, and the EC generated by the execution of the task computation. The $h(x)$ calculation formula for the objective function of delay and EC is shown in Eq. (10).

$$h(x) = \min (\lambda \times \sum_{i=1}^{n} y + \omega \times \sum_{i=1}^{n} E) \tag{10}$$

In Eq. (10), the algebraic meaning is the same as before. When the vehicle leaves, unfinished tasks will be redistributed to adjacent static vehicles or edge server nodes based on the remaining resources within the cluster. During the algorithm iteration process, the particle fitness is dynamically updated to reflect the available resource situation after the vehicle moves, thereby ensuring the continuity and stability of task offloading optimization.

4 RESULTS

4.1 Performance Test of Vehicle Clustering Model Based on K-means with Edge Computing

To verify the performance effect of the proposed model, the study builds a suitable experimental environment. Windows 11 is used as the operating system, the CPU is Intel i7-9700K, the memory is 32 GB DDR4, and the GPU is NVIDIA RTX 3060. The public dataset of public parking lots and on-street parking spaces in Haidian District, Beijing, is used as the data source to extract the locations of parked vehicles in different time periods, form vehicle clusters using the vehicle clustering model, and analyze the clustering effect. The dataset is split in an 8:2 ratio between the training and test sets. The Haidian District dataset provided static parked vehicle distributions, which were used as fixed nodes in the Veins simulation. Dynamic vehicles and traffic flows were generated synthetically, ensuring realistic hybrid scenarios. In the Veins simulation, the data of Beijing parking lots and road parking Spaces are mapped to the initial positions of vehicles and task generation points. Static vehicles are used as edge nodes, while dynamic vehicles travel on the road and are connected to the simulation environment through V2V/V2R communication to achieve the integration of real data and simulation. The road network simulation visualization diagram is shown in Fig. 8.

In Fig. 8, the study plans eight travel lanes as a whole to enhance road capacity. It ensures that vehicles can still move efficiently and orderly during times of high traffic pressure such as the morning and evening peaks. The northern part of the simulation area corresponds to the real residential area. The east-west road in this area is equipped with three two-way traffic lanes to meet the relatively low traffic demand in the residential area. The southern part is constructed as an urban environment with intensive commercial activities. To cope with the high-frequency

entry and exit scenarios of pedestrian and vehicular traffic, five lanes are planned for the east-west roadway to further enhance the efficiency of access and flexibility of traffic scheduling. The driving direction of each lane at the intersection of the north side and the south side is set as shown in Tab. 2.

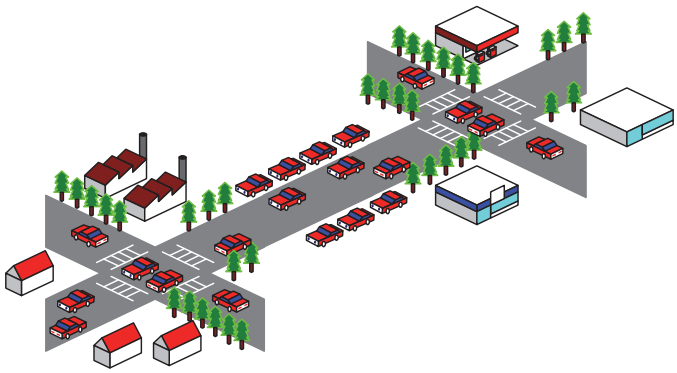


Figure 8 Road network simulation visualization diagram

Table 2 The driving directions of the lanes at the north and south intersections

Region	Direction	Straight lane	Turn right into the straight lane	Turn left into the straight lane	Right turn lane	Left-turn lane
The northern intersection	East to west	0	1	1	0	0
	West to east	0	1	1	0	0
	South to north	3	0	0	1	1
	North to south	3	0	0	1	1
The southern intersection	East to west	1	1	0	0	1
	West to east	1	0	0	1	1
	South to north	3	0	0	1	1
	North to south	3	0	0	1	1

Based on the travel directions in Tab. 2, the study simulates the traffic situation by generating traffic flows and injecting them into the road network. Out of 40 traffic streams, 16 traffic streams with parking demand are set up. These vehicles enter a parking lot for a short period of time while traveling. Subsequently, they drive out again and continue traveling along the predetermined traffic flow paths. In cluster analysis, the determination of the number of clusters is very important. Therefore, the study explores the optimal K value for the vehicular clustering model. The variation of vehicle clustering model SSE with K value at different times is shown in Fig. 9.

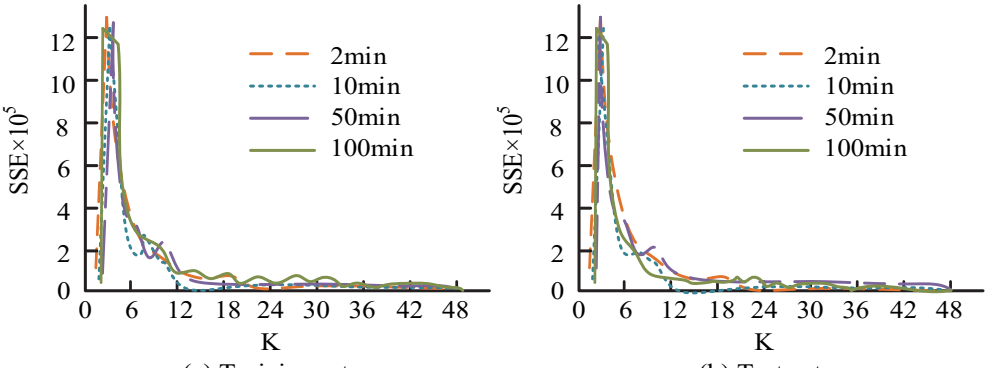


Figure 9 The variation curves of the SSE of the vehicle clustering model with the K value at different times

Fig. 9a to Fig. 9c show the variation curves of SSE of the vehicular clustering model with K value at 2 min, 10 min, 50 min, and 100 min, respectively. In Fig. 9, when the K value is less than 6, SSE decreases rapidly with the growth of K value. The SSE declines much more slowly when the K value is higher than 6. It indicates that the K value is close to the optimal interval at this time, and the effect of continuing to increase the K value on the model performance improvement is limited. Therefore, the study determines that the optimal K value is 6. For different road environments and traffic flow changes, the study selects 2 min, 10 min, 50 min, and 100 min time nodes for analysis. In the experimental process, the vehicle clustering model based on K-means and edge computing is used to divide the traveling and parked vehicles on the road into five clusters. The test results are shown in Fig. 10.

Fig. 10a and Fig. 10b shows the vehicle distribution characteristics and cluster separation results under 2 min, 10 min, 50 min, and 100 min time nodes. The degree of separation between the clusters is more obvious, and the vehicle distribution inside the clusters is also more tight, forming a good internal consistency and external separation. This clustering characteristic enables the system to effectively mobilize the computing resources of stationary vehicles within the cluster. Using it as an edge computing node, it can quickly respond to the

computational offloading requests made by the mobile vehicles within the cluster. Through local cooperative processing, the task TD is significantly reduced, and the overall task processing efficiency and the dynamic response capability of the system are improved to a certain extent. In summary, the research model has good grouping ability in practical application, and can effectively extract the characteristic differences between different vehicle aggregation areas. This can provide reliable data support for subsequent parking behavior analysis and traffic scheduling optimization.

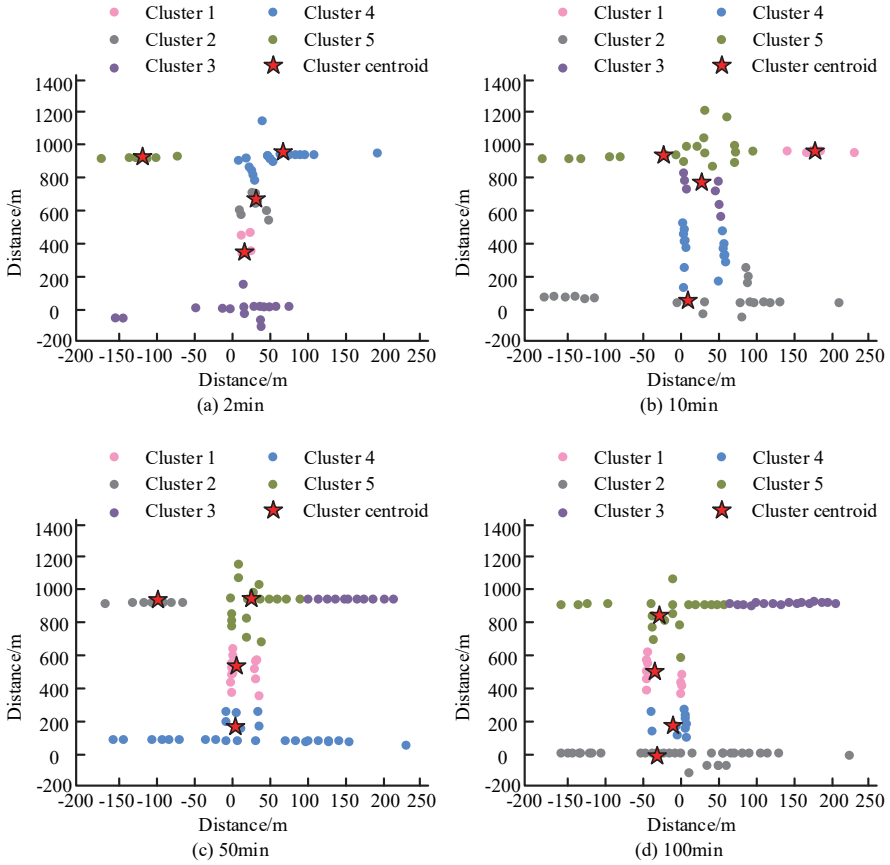


Figure 10 The distribution characteristics and clustering results of vehicles in different time periods

4.2 Simulation Test of Optimized Allocation Model for Edge Computing Task Offloading

We selected λ = 0.6 and ω = 0.4 after sensitivity analysis on validation datasets. We found that giving slightly higher weight to latency reduction than energy consumption produced more stable performance. Tab. 3 shows the impact of varying λ and ω on total system cost.

It introduces the advanced task offloading models at this stage. Namely, DRL-TO, federated learning-based task offloading (FL-TO) and deep Q-Network-based task offloading (DQN-TO) are tested as comparison models. All experiments were repeated 10 times with different random seeds. The final result is expressed as Mean ± standard deviation (Mean ± SD) to reflect the stability of the model's performance. The test results are shown in Fig.11.

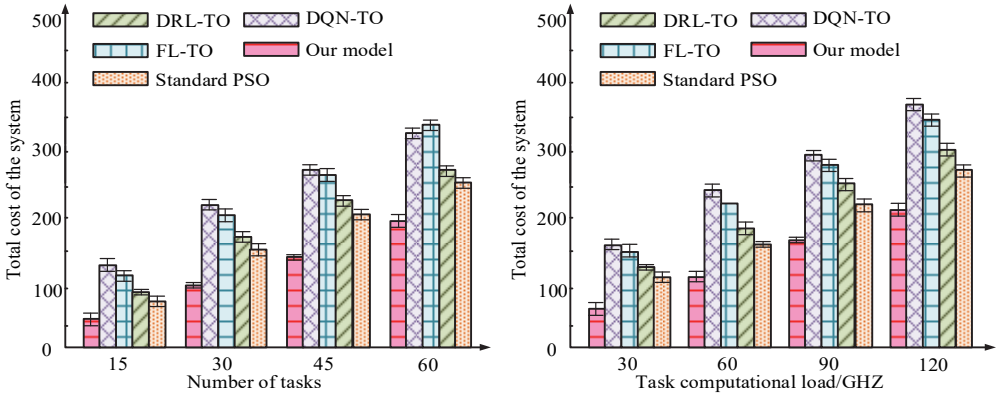
Figs. 11a and 11b illustrates how the quantity of computation and the number of tasks affect the system's overall cost. The adaptability values of different offloading strategies exhibit an overall declining trend in Fig. 11a as the number of tasks increases. This pattern of change is consistent with the trend that the total system cost rises with the increase of task size, indicating that the increase of the number of tasks has a certain negative impact on the system performance. Compared with the rest of the models, using the research model to optimize task offloading can obtain lower total system cost.

When the task is increased to 60, the total system cost corresponding to DRL-TO, FL-TO, DQN-TO, and the research model are 278 ± 5.2, 344 ± 6.1, 325 ± 5.8, and 198 ± 4.5, respectively. This is because the improved PSO algorithm is able to more fully utilize the global information when executing the offloading tasks, thus achieving a more reasonable task allocation and effectively reducing the communication overhead and computation overhead of the system. In Fig. 11b, the total system cost for all offloading strategies shows an increasing trend as the task computation volume continues to rise, and the growth gradually accelerates with the increase in computation volume. The research model consistently maintains a lower total cost at different levels of computation volume. When the task computation volume is 120 GHZ, the total system cost of the research model is 214 ± 5.0. Differences between our model and baselines were statistically significant (p & lt; 0.05, paired t-test). Its performance advantage over other models is especially noticeable in work circumstances involving large computation volumes. This verifies the good adaptability and stability of the research model in complex dynamic environments. In addition, to verify the task offloading

performance of the research model under different vehicle densities, the study sets three different density levels (low, medium, and high) corresponding to the number of 50, 100, and 150 vehicles, respectively. The computational volume is uniformly set to 100 GHz and the number of tasks is set to 30. The test results are shown in Tab. 4.

Table 3 Impact of varying λ and ω on total system cost

λ	ω	Avg. task delay / ms	Energy Consumption / J	Total system cost (Normalized)
0.2	0.8	128.6	14.8	1.12
0.4	0.6	112.4	15.6	1.05
0.6	0.4	95.7	16.9	0.98
0.8	0.2	83.2	18.4	1.01
1.0	0	75.5	19.6	1.08



(a) The influence of the number of tasks on the total cost of the system

(b) The influence of task computational complexity on the total cost of the system

Figure 11 The influence of the number of tasks and the amount of task computation on the total cost of the system

Table 4 Task offloading performance under different vehicle densities

Vehicle density	Models	Average task completion delay / ms	Average energy consumption / J	Offloading success rate / %
Low density	DRL-TO	120 ± 5	1.8 ± 0.1	87.5 ± 2.3
	FL-TO	110 ± 4	1.6 ± 0.1	90.2 ± 2.0
	DQN-TO	125 ± 6	2.0 ± 0.2	85.0 ± 3.0
	Standard PSO	105 ± 3	1.3 ± 0.1	91.0 ± 1.5
	Research model	95 ± 2	1.1 ± 0.1	97.5 ± 1.0
Medium density	DRL-TO	155 ± 7	2.2 ± 0.2	81.3 ± 2.8
	FL-TO	140 ± 5	2.0 ± 0.2	84.5 ± 2.5
	DQN-TO	165 ± 8	2.4 ± 0.2	78.6 ± 3.2
	Standard PSO	130 ± 4	1.7 ± 0.1	89.2 ± 1.8
	Research model	120 ± 3	1.5 ± 0.1	94.6 ± 1.2
High density	DRL-TO	190 ± 9	2.7 ± 0.3	74.5 ± 3.5
	FL-TO	175 ± 7	2.5 ± 0.2	77.2 ± 3.0
	DQN-TO	200 ± 10	2.9 ± 0.3	70.8 ± 3.8
	Standard PSO	160 ± 6	2.0 ± 0.2	84.3 ± 2.2
	Research model	145 ± 5	1.8 ± 0.1	92.8 ± 1.5

In Tab. 4, the research model outperforms the comparison model in average task completion delay, average EC and offloading success rate under different vehicle density levels. Under low, medium, and high density environments, the research model still maintains high offloading success rates of 97.5% ± 1.0%, 94.6% ± 1.2%, and 92.8% ± 1.5%, which fully verifies its robustness and efficiency under sparsely distributed vehicle environments. The task offloading tests are constructed with varying average vehicle speeds (low, medium, and high) to further validate the research model's performance under various mobility settings. The vehicle speeds are set to 20 km/h, 60 km/h, and 100 km/h. The test results are shown in Fig. 12.

The impact of varying average vehicle speeds on the research model's overall performance in the training and test sets is depicted in Figs. 12a and 12b. When the average vehicle speed is 20 km/h, the total system cost is the lowest, which is only 193 ± 2 and 199 ± 2, respectively. As the speed is increased to 60 km/h, the total system cost increases slightly, which is 201 ± 1 and 204 ± 2, respectively. When the average vehicle speed is further increased to 100 km/h, the total system cost rises significantly to 228 ± 4 and 224 ± 3. Differences between our model and baselines were statistically significant (p &*lt*; 0.05, paired t-test). This indicates that the faster the vehicles move, the less stable the connection between vehicles is, and more interruptions occur during task migration and collaborative computation, leading to an increase in the total cost of the system and a decrease in the task completion rate. It can be concluded that the research model is able to better optimize the task offloading performance under low and medium speed conditions. At high vehicle speeds, frequent topology changes increase communication instability, causing higher costs. This highlights the need for rapid reclustering and redundancy mechanisms in real-world IoV deployments. The total cost of the system gradually increases as the vehicle speed rises. The fundamental reason for this trend lies in the communication instability in highly mobile IoV

environments. As the vehicle speed increases, the hold time of V2V and V2R links shortens, the link handover frequency rises, and channel fading and path loss become more severe, which leads to a significant increase in data transmission failure rate and retransmission sales. Meanwhile, the high-speed movement of vehicles also intensifies the dynamic changes of the topology within the cluster. The average communication distance between cluster heads and cluster members fluctuates greatly, increasing the probability of task redistribution in the middle and further pushing up the system overhead. The task offloading model based on the improved PSO proposed by the research institute is superior to the traditional baseline algorithm because its coding mechanism and ring topology strategy can dynamically adapt to high-speed mobile scenarios. By prioritizing the scheduling of high-speed vehicle tasks, delay suppression and energy consumption balance, it effectively alleviates the additional costs caused by unstable communication. From the perspective of practical application, this model has potential value in the deployment of IoV edge computing. The introduction of statically parked vehicles enhances the stability of edge resources, but their computing and storage capabilities are relatively limited. Therefore, at the hardware level, the processor performance, energy consumption constraints, and storage capacity of the on-board unit need to be considered. Meanwhile, the computing power of the VEC controller and RSU will also directly affect the real-time performance and stability of task scheduling. If the model is applied to an actual IoV system, it must be comprehensively optimized by taking into account factors such as the differences in on-board hardware computing power of vehicles, the heterogeneity of edge nodes, and the coverage range of 5G/6G networks. For instance, custom AI accelerators can be adopted in VEC controllers, or energy efficiency optimization mechanisms can be introduced at the vehicle end to ensure the deployability and scalability of the model in large-scale IoV environments.

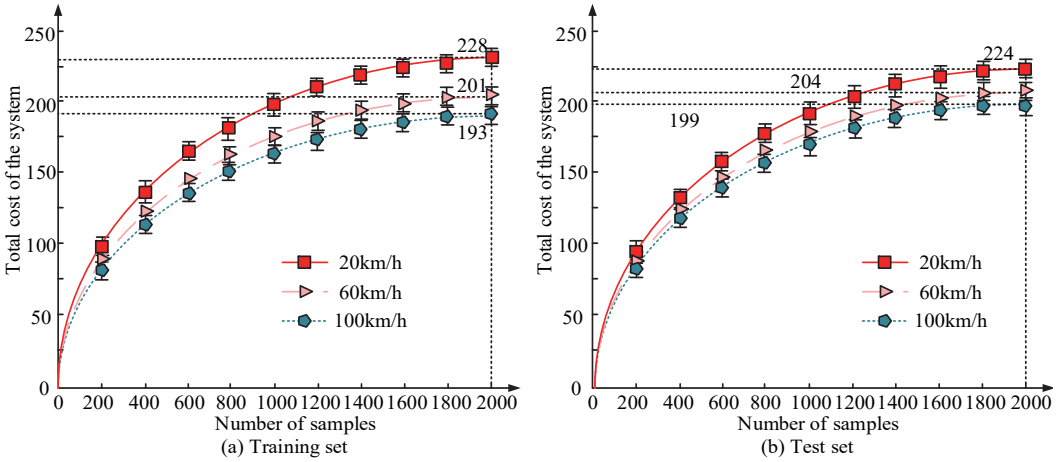


Figure 12 The influence of the average speed of different vehicles on the overall performance of the research model

Table 5 Offloading effects under different computing capabilities of ESs

Vehicle density	Models	Maximum task completion time / ms	Average task completion time / ms	Average utilization rate of nodes / %	Throughput / tasks / s	LBI	Fairness Index
Low ability (2 GHz)	DRL-TO	380 ± 12	210 ± 10	67.3 ± 3.1	4.5 ± 0.2	0.68 ± 0.03	0.72 ± 0.03
	FL-TO	360 ± 10	200 ± 9	68.9 ± 2.8	4.7 ± 0.2	0.70 ± 0.02	0.74 ± 0.03
	DQN-TO	400 ± 14	225 ± 12	65.7 ± 3.3	4.2 ± 0.2	0.66 ± 0.03	0.70 ± 0.03
	Standard PSO	350 ± 11	205 ± 10	69.0 ± 2.9	4.8 ± 0.2	0.71 ± 0.03	0.73 ± 0.03
	Research model	330 ± 9	185 ± 8	71.5 ± 2.5	5.1 ± 0.2	0.75 ± 0.02	0.78 ± 0.02
Medium ability (4 GHz)	DRL-TO	290 ± 9	160 ± 7	72.1 ± 2.6	5.8 ± 0.3	0.73 ± 0.02	0.76 ± 0.02
	FL-TO	270 ± 8	150 ± 6	73.5 ± 2.4	6.2 ± 0.3	0.75 ± 0.02	0.78 ± 0.02
	DQN-TO	310 ± 10	170 ± 8	70.8 ± 2.8	5.5 ± 0.3	0.71 ± 0.02	0.74 ± 0.02
	Standard PSO	280 ± 9	155 ± 7	74.0 ± 2.5	6.4 ± 0.3	0.76 ± 0.02	0.79 ± 0.02
	Research model	240 ± 7	135 ± 6	77.2 ± 2.3	7.0 ± 0.3	0.80 ± 0.02	0.83 ± 0.02
High ability (6 GHz)	DRL-TO	220 ± 7	120 ± 5	76.4 ± 2.1	7.5 ± 0.3	0.78 ± 0.02	0.81 ± 0.02
	FL-TO	200 ± 6	110 ± 4	77.9 ± 2.0	8.0 ± 0.3	0.80 ± 0.02	0.83 ± 0.02
	DQN-TO	230 ± 8	125 ± 5	75.3 ± 2.2	7.2 ± 0.3	0.77 ± 0.02	0.80 ± 0.02
	Standard PSO	210 ± 6	115 ± 4	78.5 ± 2.0	8.1 ± 0.3	0.81 ± 0.02	0.84 ± 0.02
	Research model	180 ± 5	95 ± 4	81.6 ± 1.9	8.8 ± 0.3	0.85 ± 0.02	0.87 ± 0.02

Although the overall performance of the research model is degraded in the high-speed environment, it still maintains a high task completion rate with a low total system cost. Finally, the study also explores the model's adaptability under different ES resource conditions. Three ES computing power levels are set (low power: 2GHz, medium power: 4GHz, and high power: 6GHz). The number of stationary vehicles in each condition is 100, the

number of tasks is 50, and the computational demand is 80 GHz. The test results are shown in Tab. 5.

In Tab. 5, as the ES's processing power rises, both the average and maximum job completion times for each model exhibit a declining trend. The average node utilization rate gradually increases, indicating that the abundance of computing resources helps to improve the overall task offloading performance of the models. Under the low-capability server condition, the maximum TCT of the research model is 330 ms ± 9 ms, the average TCT is 185 ms ± 8 ms, and the average node utilization rate reaches 71.5% ± 2.5%. In the medium-capability server condition, the maximum TCT of the research model is 240 ms ± 7ms, the average TCT is 135 ms ± 6 ms, and the average node utilization rate increases to 77.2% ± 2.3%. In the high-capability server environment, the maximum TCT of the research model is 180ms ± 5ms, the average TCT is only 95 ms ± 4 ms, and the average node utilization rate reaches 81.6% ± 1.9%. It is significantly better than DRL-TO, FL-TO and DQN-TO models. It can be concluded that the research model has good adaptability and stability under different server computing power conditions.

5 CONCLUSION

To further enhance the task processing efficiency and overall system performance in the IoV environment, the study proposed a novel IoV task offloading optimization allocation model based on the MEC architecture. The model firstly adopted the K-means clustering algorithm to reasonably cluster static parked vehicles and mobile vehicles, which effectively enhanced the internal consistency and external separation of the system. This provided a good structural foundation for the subsequent task allocation and scheduling. Subsequently, the traditional PSO algorithm was coded and optimized to better adapt to the high dynamics and node heterogeneity problems in the IoV environment. This increased the offloading decision's precision and resilience.

The outcomes indicated that at 2 min, 10 min, 50 min, and 100 min time nodes, the vehicle clusters were clearly separated from each other, and the vehicles within the clusters were tightly distributed. This confirmed the suggested model's efficacy in vehicle clustering. The research model always had a low total system cost under different levels of computation and number of tasks. In addition, the research model outperformed the comparison model in three metrics: average task completion delay, average EC, and offloading success rate under different vehicle density levels. Under low, medium, and high density environments, the research model still maintained high offload success rates of 97.5%, 94.6%, and 92.8%. In the high-capacity server environment, the maximum TCT of the research model was 180ms, the average TCT was only 95 ms, and the average node utilization reached 81.6%, which was significantly better than the rest of the state-of-the-art models. It can be concluded that the proposed model has significant advantages in terms of task offloading efficiency, EC optimization and system performance. This study is limited by its reliance on simulation with partial real-world data and absence of hardware-in-the-loop experiments. Future work will incorporate federated learning for dynamic adaptation, and evaluate the model in real vehicular testbeds.

6 REFERENCES

[1] Dalbah, L. M., Al-Betar, M. A., & Awadallah, M. A. (2024). Jaya clustering-based algorithm for multiobjective IoV network routing optimization. *Soft Computing*, *28*(6), 5639-5665. https://doi.org/10.1007/s00500-023-09350-y

[2] Wang, J., Zhu, K., & Hossain, E. (2021). Green Internet of Vehicles (IoV) in the 6G era: Toward sustainable vehicular communications and networking. *IEEE Transactions on Green Communications and Networking*, *6*(1), 391-423. https://doi.org/10.1109/TGCN.2021.3127923

[3] Mirza, M. A., Yu, J., & Ahmed, M. (2023). DRL-driven zero-RIS assisted energy-efficient task offloading in vehicular edge computing networks. *Journal of King Saud University-Computer and Information Sciences*, *35*(10), 101837-101845. https://doi.org/10.1016/j.jksuci.2023.101837

[4] Quan, H., Zhang, Q., & Zhao, J. (2024). Federated learning assisted intelligent IoV mobile edge computing. *IEEE Transactions on Green Communications and Networking*, *9*(1), 228-241. https://doi.org/10.1109/TGCN.2024.3421357

[5] Alam, M. Z. & Jamalipour, A. (2022). Multi-agent DRL-based Hungarian algorithm (MADRLHA) for task offloading in multi-access edge computing Internet of Vehicles (IoVs). *IEEE Transactions on Wireless Communications*, *21*(9), 7641-7652. https://doi.org/10.1109/TWC.2022.3160099

[6] Yang, S., Tan, J., & Lei, T. (2023). Smart traffic navigation system for fault-tolerant edge computing of internet of vehicle in intelligent transportation gateway. *IEEE Transactions on Intelligent Transportation Systems*, *24*(11), 13011-13022. https://doi.org/10.1109/TITS.2022.3232231

[7] Jia, Y., Zhang, C., & Huang, Y. (2022). Lyapunov optimization based mobile edge computing for Internet of Vehicles systems. *IEEE Transactions on Communications*, *70*(11), 7418-7433. https://doi.org/10.1109/TCOMM.2022.3206885

[8] Sun, F., Zhang, Z., & Zeadally, S. (2022). Edge computing-enabled Internet of Vehicles: Towards federated learning empowered scheduling. *IEEE Transactions on Vehicular Technology*, *71*(9), 10088-10103. https://doi.org/10.1109/TVT.2022.3182782

[9] Cao, B., Sun, Z., & Zhang, J. (2021). Resource allocation in 5G IoV architecture based on SDN and fog-cloud computing. *IEEE Transactions on Intelligent Transportation Systems*, *22*(6), 3832-3840. https://doi.org/10.1109/TITS.2020.3048844

[10] Xu, X., Jiang, Q., & Zhang, P. (2022). Game theory for distributed IoV task offloading with fuzzy neural network in edge computing. *IEEE Transactions on Fuzzy Systems*, *30*(11), 4593-4604. https://doi.org/10.1109/TFUZZ.2022.3158000

[11] Jamalzadeh, M., Maadani, M., & Mahdavi, M. (2022). EC-MOPSO: An edge computing-assisted hybrid cluster and MOPSO-based routing protocol for the Internet of Vehicles. *Annals of Telecommunications*, *77*(7), 491-503. -https://doi.org/10.1007/s12243-021-00892-6

[12] Dong, S., Xia, Y., & Kamruzzaman, J. (2022). Quantum particle swarm optimization for task offloading in mobile edge computing. *IEEE Transactions on Industrial Informatics*, *19*(8), 9113-9122. https://doi.org/10.1109/TII.2022.3185144

[13] Sami, H., Saado, R., & El Saoudi, A. (2023). Opportunistic UAV deployment for intelligent on-demand IoV service management. *IEEE Transactions on Network and Service Management*, *20*(3), 3428-3442. https://doi.org/10.1109/TNSM.2023.3242205

[14] Sharma, S. & Kaushik, B. (2021). A survey on nature-inspired algorithms and its applications in the internet of vehicles. *International Journal of Communication Systems*, *34*(12), e4895-e4902. https://doi.org/10.1002/dac.4895

[15] Raza, S., Wang, S., & Ahmed, M. (2021). Task offloading and resource allocation for IoV using 5G NR-V2X communication. *IEEE Internet of Things Journal*, *9*(13), 10397-10410. https://doi.org/10.1109/JIOT.2021.3121796

[16] Zhang, R., Wu, L., Cao, S. et al. (2021). Task offloading with task classification and offloading nodes selection for MEC-enabled IoV. *ACM Transactions on Internet Technology (TOIT)*, *22*(2), 1-24. https://doi.org/10.1145/3475871

[17] Hazarika, B., Singh, K., & Biswas, S. (2023). Multi-agent DRL-based task offloading in multiple RIS-aided IoV networks. *IEEE Transactions on Vehicular Technology*, *73*(1), 1175-1190. https://doi.org/10.1109/TVT.2023.3302010

[18] Karim, S. M., Habbal, A., & Chaudhry, S. A. (2023). BSDCE-IoV: Blockchain-based secure data collection and exchange scheme for IoV in 5G environment. *IEEE Access*, *11*(7), 36158-36175. https://doi.org/10.1109/ACCESS.2023.3265959

[19] Yang, C., Xu, X., & Zhou, X. (2022). Deep Q network–driven task offloading for efficient multimedia data analysis in edge computing-assisted IoV. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, *18*(2), 1-24. https://doi.org/10.1145/3548687

[20] Zheng, J., Zhang, Y., & Luan, T. H. (2023). Digital twin enabled task offloading for IoVs: A learning-based approach. *IEEE Transactions on Network Science and Engineering*, *11*(1), 659-672. https://doi.org/10.1109/TNSE.2023.3303461

[21] Chen, C., Li, H., & Li, H. (2022). Efficiency and fairness oriented dynamic task offloading in internet of vehicles. *IEEE Transactions on Green Communications and Networking*, *6*(3), 1481-1493. https://doi.org/10.1109/TGCN.2022.3167643

[22] Ren, H., Liu, K., & Liu, C. (2023). An approximation algorithm for joint data uploading and task offloading in IoV. *IEEE Transactions on Consumer Electronics*, *70*(1), 3018-3030. https://doi.org/10.1109/TCE.2023.3325319

[23] Sun, Y., Wu, Z., Meng, K. et al. (2023). Vehicular task offloading and job scheduling method based on cloud-edge computing. *IEEE Transactions on Intelligent Transportation Systems*, *24*(12), 14651-14662. https://doi.org/10.1109/TITS.2023.3300437

[24] Yuan, X., Chen, J., & Zhang, N. (2022). Digital twin-driven vehicular task offloading and IRS configuration in the Internet of Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, *23*(12), 24290-24304. https://doi.org/10.1109/TITS.2022.3204585

[25] Cui, Y., Li, H., & Zhang, D. (2023). Multiagent reinforcement learning-based cooperative multitype task offloading strategy for Internet of Vehicles in B5G/6G network. *IEEE Internet of Things Journal*, *10*(14), 12248-12260. https://doi.org/10.1109/JIOT.2023.3245721

[26] Fan, W., Su, Y., & Liu, J. (2023). Joint task offloading and resource allocation for vehicular edge computing based on V2I and V2V modes. *IEEE Transactions on Intelligent Transportation Systems*, *24*(4), 4277-4292. https://doi.org/10.1109/TITS.2022.3230430

[27] Yan, M., Xiong, R., & Wang, Y. (2023). Edge computing task offloading optimization for a UAV-assisted internet of vehicles via deep reinforcement learning. *IEEE Transactions on Vehicular Technology*, *73*(4), 5647-5658. https://doi.org/10.1109/TVT.2023.3331363

[28] Li, H., Chen, C., & Shan, H. (2023). Deep deterministic policy gradient-based algorithm for computation offloading in IoV. *IEEE Transactions on Intelligent Transportation Systems*, *25*(3), 2522-2533. https://doi.org/10.1109/TITS.2023.3325267

[29] Wang, S., Li, J., & Wu, G. (2021). Joint optimization of task offloading and resource allocation based on differential privacy in vehicular edge computing. *IEEE Transactions on Computational Social Systems*, *9*(1), 109-119. https://doi.org/10.1109/TCSS.2021.3074949

[30] Lai, Q., Xiong, C., & Chen, J. (2023). Improved transformer-based privacy-preserving architecture for intrusion detection in secure V2X communications. *IEEE Transactions on Consumer Electronics*, *70*(1), 1810-1820. https://doi.org/10.1109/TCE.2023.3324081

[31] Song, R., Xu, R., & Festag, A. (2023). FedBEVT: Federated learning bird's eye view perception transformer in road traffic systems. *IEEE Transactions on Intelligent Vehicles*, 9(1), 958-969. https://doi.org/10.1109/TIV.2023.3310674

[32] Li, H., Cai, Z., & Wang, J. (2023). FedTP: Federated learning by transformer personalization. *IEEE Transactions on Neural Networks and Learning Systems*, *35*(10), 13426-13440. https://doi.org/10.1109/TNNLS.2023.3269062

Contact information

ZhiXiong JIN
Geely University of China,
Jianyang City, Chengdu City, 641423, Sichuan Province, China
E-mail: jinzxiong8210@yeah.net