Many-Objective Edge Computing Server Deployment Optimization for Vehicle Road Cooperation

Shanshan Fan * and Bin Cao

Posted Date: 31 October 2025

doi: 10.20944/preprints202510.2415.v1

Keywords: many-objective evolution algorithm; internet of vehicles (IoV); edge computing (EC); roadside units (RSUs); vehicle-road cooperation

 doi:10.20944/preprints202510.2415.v1

Article

Many-Objective Edge Computing Server Deployment Optimization for Vehicle Road Cooperation

Shanshan Fan [1,2,*] and Bin Cao [1,2]

[1] Hebei University of Technology, Tianjin 300401, China
[2] State Key Laboratory of Intelligent Power Distribution Equipment and System, Hebei University of Technology, Tianjin 300401, China
* Correspondence: 826905804@qq.com

Abstract

In the Internet of Vehicles (IoV), vehicles need to process a large amount of perception data to support tasks such as road navigation and autonomous driving. However, their computational resources are limited. Therefore, it is necessary to explore the combination of vehicle-road cooperation with edge computing. Roadside units (RSUs) can provide data access services for vehicles, and deploying edge servers on RSUs can improve the data processing capability in IoV environments and ensure the sustainability of vehicle communications, thus supporting complex traffic scenarios more effectively. In this work, we study the deployment of RSUs in vehicle-road cooperative systems. To balance the deployment cost of RSUs and the quality of service (QoS) of vehicle users, we propose an RSU deployment optimization model with six objectives, including time delay, energy consumption and security when vehicles offload their tasks to RSUs, as well as load balancing and the number and communication coverage area of RSUs. In addition, we propose a Wasserstein generative adversarial network (WGAN)-based Two_Arch2 (WGTwo_Arch2) to solve this many-objective optimization problem to better ensure the diversity and convergence of the solutions. In addition, a polynomial variation strategy based on Lecy's flight mechanism and a diversity archive selection strategy with an adaptive Lp-norm are also proposed to balance the exploratory and exploitative capabilities of the algorithm. The effectiveness of the proposed algorithm WGTwo_Arch2 for 6-objective RSU deployment optimization is verified by comparisons with five different algorithms.

Keywords: many-objective evolution algorithm; internet of vehicles (IoV); edge computing (EC); roadside units (RSUs); vehicle-road cooperation

1. Introduction

In recent years, with the rapid development of new generation communication technologies such as 5G and 6G, the number of vehicles connected to the Internet of Vehicles (IoV) and the amount of data for in-vehicle applications have increased [1]. However, vehicles have limited computing power and storage to perform many computational tasks, requiring them to offload tasks to servers for processing. Cloud servers can provide powerful computational services. However, they are physical far away from vehicle users and incompatible with latency-sensitive tasks. Unlike traditional cloud processing, edge computing effectively reduces network latency, saves energy in vehicle transmissions by processing data locally, and reduces the risk of cybersecurity attacks when data are in the cloud. As an infrastructure in the IoV, roadside units (RSUs) are fixed devices installed near roads that have stronger communication and computation capabilities than vehicles do. RSUs provide wide-coverage network access for vehicles [2] and play a key role in vehicle-road cooperation [3]. In IoV, edge servers are usually deployed on RSUs to provide various real-time services for vehicles. Intelligent vehicles in real-world scenarios can provide a variety of intelligent services. For example, vehicle control and road traffic-type services require high real-time performance to improve road safety and avoid

 doi:10.20944/preprints202510.2415.v1

hazardous situations, and entertainment and media-type services require continuous data streams to maintain a better service experience [4], whereas RSUs are costly to install and maintain, and large-scale deployments can impose expensive overheads on operators. Therefore, the RSU deployment problem is modeled as a multiobjective optimization problem, aiming to balance between conflicting user service quality and deployment cost, which is more suitable for practical application scenarios.

Vehicle-road cooperation systems rely primarily on wireless communication; however, when the wireless transmission range is limited, RSUs can provide access services only to vehicles within the communication area. To reduce the energy consumption and time delay of vehicles, in this study, task computation in vehicles is not considered. Instead, the tasks are offloaded to edge or cloud servers to provide satisfactory quality of service (QoS) for users. In addition, considering vehicle mobility, the time required complete computational tasks may necessitate data transmissions across multiple RSUs during task execution [5]. Then, it is necessary to backhaul the data in the form of wireless multihop communication between multiple RSUs. However, this process can easily be used by attackers to steal sensitive information, thus exposing vehicle users to security threats. Therefore, the ability to ensure the privacy of vehicle users is a key factor affecting the QoS. Existing security algorithms typically rely on encryption for ensuring data privacy. However, the extensive encryption/decryption process leads to severe latency and energy consumption, which is a key challenge for resource-limited vehicles. In this study, instead of using security techniques such as encryption and authentication, the probability of eavesdropping during multihop transmission is used as an optimization objective to secure the network while minimizing latency and energy consumption. Providing higher QoS to more vehicles is the fundamental purpose of deploying RSUs. Therefore, the coverage area is also an important measure used to evaluate the QoS provided by the network. To fully utilize the computational potential of edge computing so that vehicle requests should be processed in RSUs as much as possible, in this study, the number of vehicles covered by RSUs is used as an objective to measure RSU coverage. In addition, RSUs are expensive to install and maintain. When RSUs are not load balanced, it prolonged high loads on certain servers can lead to overheating and hardware failure, whereas idle RSUs lead to resource waste. Therefore, this paper introduces the concept of maximizing the load balance of RSUs to reduce the maintenance cost and maximize the operator's investment. Most studies maximize the QoS within a limited budget [4,6,7]. Previous RSU deployment optimization studies usually consider deployment cost, coverage area, time delay, or energy consumption as a measure of QoS [8–10], but few consider factors such as transmission security and load balancing; therefore, in this paper, we construct a six-objective RSU deployment optimization model including network delay and energy consumption, network transmission security, load balancing, deployment cost, and coverage area.

The RSU deployment strategy has been shown to be an NP-hard problem [1,11]. The evolutionary algorithm (EA) is a population-based stochastic optimization algorithm inspired by the law of natural evolution that does not easily fall into local optimal solutions when solving NP-hard problems and has good performance in nondifferentiable nonconvex and multiobjective optimization problems [12–15]. However, when the number of objectives to be optimized increases, the convergence speed and diversity of traditional multiobjective evolutionary algorithms decrease. Therefore, when solving many-objective optimization problems (the number of objectives is greater than 3), the overall optimization often leads to many locally optimal solutions during the search process as the dimensionality of the decision variables grows rapidly, which not only prevents the algorithms from finding the globally optimal solution but also may trigger premature convergence or even stagnation. To address this challenge effectively, Wang et al. [12] proposed the Two_Arch2 algorithm for solving many-objective optimization problems. The algorithm preserves the diversity and convergence of candidate solutions by maintaining two different archives, each with a specific selection and updating strategy. The algorithm performs well in terms of convergence, diversity and complexity. In addition, by learning sufficiently from high-quality datasets, deep neural networks exhibit strong feature representation and nonlinear fitting capabilities. Data generation methods based on deep generative adversarial network (GAN) models have shown extremely superior performance in learning complex high-dimensional

 doi:10.20944/preprints202510.2415.v1

data distributions. The traditional GAN [16] is a generative and discriminative gaming process through which adversarial training ultimately leads to the generation of high-quality synthetic data. Wasserstein-GAN (WGAN) [17] is an improved GAN model that introduces the Wasserstein distance as an optimization objective to train generators and discriminators in a more stable and interpretable way. Therefore, this paper introduces the WGAN-based Two_Arch2 algorithm (WGTwo_Arch2), which uses a WGAN-based population initialization strategy and population mating selection strategy to improve the diversity of solutions in the population, as well as a polynomial mutation strategy based on Levy's flight mechanism and a diversity archive updating strategy based on the adaptive Lp-norm to increase the exploration efficiency and global convergence of the algorithm.

This study focuses on the MaOP algorithm and its application to the problem of deploying RSUs in edge computing-based vehicle-road cooperation. The main contributions of this study are as follows:

1. A six-objective optimization model incorporating time delay, energy consumption, security, load balancing, and number and communication coverage areas is proposed for the RSU deployment optimization problem.
2. The WGTwo_Arch2 algorithm is proposed to optimize the many-objective deployment model of RSUs. To enhance the algorithm's ability to identify diverse solutions, Kent chaotic mapping data is applied to train the WGAN to generate random individuals covering the entire distribution space during the population initialization process, making the initial population distribution more uniform. A mating selection strategy based on the WGAN is designed to generate more diverse solutions for offspring generation.
3. To improve the global search ability and convergence speed of the algorithm when dealing with many-objective problems, a polynomial variation strategy based on the Levy flight mechanism is proposed. The stochastic wandering mechanism of the Levy flight is used to generate variation probabilities, enabling better exploration of the global search space. An adaptive Lp-norm-based strategy for updating diversity archives is proposed to control exploration and exploitation. Finally, the effectiveness of the proposed algorithm in solving the RSU many-objective deployment optimization problem is experimentally verified.

The remainder of this paper is arranged as follows. The related work is given in Section 2. The architecture of the vehicle-road cooperation system based on RSUs and the RSU deployment optimization model are presented in Section 3.1. The proposed WGTwo_Arch2 algorithm is described in Section 3.2. Section 4 provides the experimental comparison results and analysis. Finally, Section 5 summarizes this article.

2. Related Work

2.1. Large-Scale Many-Objective Optimization Problem (LSMaOP)

Multiobjective optimization problems (MOPs) involve several conflicting objective functions that need to be optimized simultaneously to obtain a set of satisfactory solutions, which can be described as follows:

$$\begin{aligned} \text{Minimize } F(X) &= \{f_1(X), f_2(X), \ldots, f_m(X)\} \\ \text{s.t. } X &= (x_1, x_2, \ldots, x_n) \in \Omega \end{aligned} \tag{1}$$

where there are n dimensions of decision variables and m objective functions, in the decision space Ω, individual X contains n dimensions of decision variables. When the number of decision variables is greater than or equal to 1000 and the number of objective functions is greater than 3, the multi-objective optimization problem is classified as a large-scale many-objective optimization problem. The decision space Ω is the set of all possible decision variables in a multiobjective optimization problem, which contains the individuals x_1,x_2. The first solution x_1 is said to dominate the second solution x_2 if solution x_1 outperforms solution x_2 on all the objective functions or is better on at least one of the objective functions and is at least non-inferior to solution x_2 on the others. A Pareto-optimal solution is one in which there is no other solution in the decision space that is superior to it on all objective

 doi:10.20944/preprints202510.2415.v1

functions. That is, a Pareto optimal solution is nondominated, and a set of Pareto optimal solutions together form a Pareto Optimal Set (PS).

2.2. *Two_Arch2 Algorithm*

The Two_Arch algorithm divides the set of nondominated solutions into two archives: the convergence archive (CA) and the diversity archive (DA). The CA and DA operate under different update rules due to their different objectives (convergence and diversity, respectively). If a new candidate individual is a nondominated member of the population, i.e., no individual in the two archives can dominate it and the candidate can dominate the other individuals in the two archives, the candidate enters the CA, and the dominated individual is removed. If the candidate individual cannot dominate any other individual and is not dominated by any archived individual, it enters the DA. In the Two_Arch algorithm, the sizes of CA and DA are variable. However, the total number of individuals in both archives is kept constant. If the total archive size is exceeded, the Euclidean distance from an individual in the DA to an individual in the CA is calculated. Then, the individual with the shortest Euclidean distance in the DA is remove. This process is repeated until the total archive size is restored. The Two_Arch2 algorithm effectively improves the update strategies of the CA and DA on the basis of the two archives of the original Two_Arch algorithm. It introduces the $I_{\varepsilon+}$ metric [14] in the CA to accelerate convergence toward the true Pareto front (PF), while leveraging the Pareto advantage to increase the diversity of the population in the DA. Specifically, the sizes of the CA and DA are fixed individually. If the CA exceeds the fixed size, the quality indicator $I_{\varepsilon+}$ guides selection, and the solution with the smallest $I_{\varepsilon+}$ loss is removed. If the DA exceeds the fixed size, the solution with the largest or smallest objective value is removed first, and then the DA is updated using the similarity metric that is based on the Lp-norm (p<1) distance until the fixed size is restored. In addition, during the offspring generation process, crossover operations are performed in the CA and DA, but only mutation operations are performed in the CA, and finally, the DA with better diversity is output. Algorithm 1 shows the pseudo-code of the the Two_Arch2.

Algorithm 1: The Two-Archive2 Algorithm

Input: population size N, maximum number of iterations $MaxFE$;
Output: the diversity archive DA;

```
Population initialization;
Update CA and DA according to the empty set and population;
while t < MaxFE do
    Select ParentC and ParentM from CA and DA according the mating selection strategy;
    Apply crossover and mutation operators to ParentC and ParentM;
    while |CA| > N_CA do
        Delete individual x* with the minimal I_ε+ loss function;
    if |DA| > N_DA then
        Remain the individuals with maximal or minimal objective values;
    while |DA| ≤ N_DA do
        Calculate the minimal Lp-norm-based distance between other solutions and extremal
          solutions;
        Remain the individual with the maximum and minimum distance from the previous
          step;
    t = t + 1;
return DA
```

2.3. *Research Status of RSU Deployment Optimization*

With the rapid development of the IoV, achieving low time delay and high security have become important requirements for many electronic devices. By applying edge computing to vehicle-road cooperation, it becomes possible to provide stable access services for vehicles, minimize the distance

 doi:10.20944/preprints202510.2415.v1

and time delay of data transmission, and reduce the risk of cybersecurity attacks on the data in the cloud, ensuring that the vehicles can access networks stably and quickly to obtain the required services. This is highly important for improving the user experience and ensuring service continuity.

RSU deployment is considered a multiobjective optimization problem that has been studied by some scholars using EAs. For example, Ning et al. [2] proposed an energy-efficient scheduling (MEES) scheme based on heuristic algorithms supporting MEC to minimize the energy consumption of RSUs under task delay constraints. Considering the mobility of vehicles in the IoV, Talpur et al. [8] proposed a deep reinforcement learning-based dynamic service placement (DRLD-SP) framework to minimize the maximum edge resource usage and service latency. Chen et al. [18] solved the joint optimization problem of user coverage and reliability for edge application placement (EAP-CR) by using integer planning techniques. Lin et al. [9] constructed a multiobjective RSU deployment model (MORD) to minimize the data transmission delay, data transmission loss rate, RSU-aware overlap region and communication overhead of RSUs and maximize the average RSU coverage. They applied the nondominated sorting genetic algorithm II (NSGA-II) [19] to solve the proposed MORD optimization problem. However, these five objectives were merged into two objective functions, and energy consumption and mobility were not considered. Fan et al. [1] proposed a task offloading scheme for heterogeneous cellular networks that combines 5G base stations with 4G base stations and applied the linear relaxation improved branch-and-bound algorithm (BBA) to minimize two objectives, total delay and energy consumption, to identify the optimal task offloading and resource allocation scheme, thus improving the user experience. Chaudhary et al. [11] minimized power consumption and maximized the model secrecy capability using mixed integer nonlinear programming (MINLP) and proposed a lightweight resilient key exchange authentication protocol to mitigate the various IoV security threats. Zhang et al. [10] maximized RSU coverage with minimum deployment cost and constructed a multiobjective model using an enhanced MOQPSO algorithm. Massobrio et al. [4] optimized two objectives, QoS and RSU deployment cost. for vehicular ad hoc networks using an improved NSGA-II evolutionary algorithm to solve for the optimal location and type of RSU. The transmitted data rate and the end-to-end delay were utilized as QoS measures. Nikookaran et al. [6] considered minimizing the capital expenditure (CAPEX) installation/opening costs and long-term energy operating (OPEX) costs of RSUs and proposed the minimum cost route clustering algorithm to solve the integer linear program problem to detect RSU locations. Kumrai et al. [20] applied EAs to optimize the communication coverage area as well as the number of RSUs. Wang et al. [21] proposed a multiobjective differential evolution algorithm-based discrete elite orientation (MODE-deg) to optimize the number and coverage area of RSUs. Guerna et al. [22] proposed a new genetic intersection-coverage algorithm (GICA) based on the priority concept to optimize the number of and coverage of RSUs. Wu et al. [23] proposed a capacity maximization placement scheme for RSUs in which vehicles can communicate with RSUs either directly or through multiple hops. An integer linear programming model was developed to maximize the aggregate throughput in the network. By considering high traffic density and low traffic density scenarios, Yu et al. [7] established an optimization model for RSU deployment with the objectives of data transmission delay and RSU coverage and constructed a model using EA. Ghosh et al. [24] proposed the memetic framework-based optimal RSU deployment (MFRD) algorithm to maximize the coverage area of each RSU and minimize the overlap between the coverage areas of neighboring RSUs. Considering the problem that a fixed coverage distance cannot adapt to diverse V2R scenarios, Huo et al. [25] assumed that RSUs have different transmit powers, performed coverage analysis of RSUs on the basis of the packet delivery ratio model to obtain the coverage distance, and used the total coverage time as the evaluation metric for the QoS, considering the purchase and installation costs. Finally, the NSGA-II was applied to solve this RSU multiobjective deployment optimization problem to improve the QoS and reduce the deployment cost. Lu et al. [26] proposed a deep reinforcement learning-based ES placement policy (DESP) to maximize coverage and workload balancing and minimize the average latency. Shen et al. [27] proposed a dynamic ES placement method (DEP), which models the DEP as the optimization problem with the reconstruction

 doi:10.20944/preprints202510.2415.v1

cost, network delay, and load balancing as the objectives and the threshold of the number of ESs and the edge coverage as the constraints. The nondominated sorting genetic algorithm III (NSGA-III) [28] was applied to optimize the location and the number of ESs.

3. Methods

3.1. System Model and Problem Formulation

In this section, we introduce the architecture of an edge computing-based vehicle-road cooperation. Within this framework, RSUs equipped with edge servers play a central role by providing services to IoV devices. RSUs are not only responsible for storing massive amounts of data but also for analyzing and processing these data in depth to ensure that IoV devices can obtain accurate and timely service support. Therefore, to optimize the overall system performance, we propose a many-objective RSU deployment optimization model, which integrally considers six objectives, including the time delay, energy consumption, and security of vehicle offloading tasks, as well as load balancing, deployment cost, and coverage area of RSUs.

3.1.1. The Vehicle-Road Cooperation System for RSU-Based Deployment

The architecture of the edge computing-based vehicle-road cooperation is shown in Figure 1. It is assumed that vehicles and RSUs carrying edge servers are randomly distributed on a unidirectional road with a distance of 3 km. The RSUs are distributed along the road, and each RSU covers a communication area where vehicles can offload their computational tasks via a wireless local area network (WLAN). The RSUs are connected to each other via a 5G core network. Assume that there are m edge computing servers in the system and that these m RSUs are arranged in order from left to right as $E = \{e_1, e_2, \cdots, e_m\}$. The system includes n ($n = 50$) vehicles, denoted by $V = \{v_1, v_2, \cdots, v_n\}$. It is assumed that each vehicle has a task to process and offload to the nearest edge server or to a cloud server for processing. In addition, assume that each vehicle v_n has a computationally intensive task $Z_n = (b_n, d_n)$. [21], where b_n denotes the size of the input data (in bits) for this task, which is randomly set to the size of [400,600] (in kbits). d_n denotes the total number of CPU cycles required to complete the computational task, which is randomly set to [900,1100] (in megacycles). The number of CPU cycles d_n is positively correlated with the task size b_n.

During a certain period, vehicles simultaneously send task processing requests to RSUs. The offloading decision of user v_n is denoted by x_n^i; if $i \leq 1 \leq m$, then user v_n offloads the task to the edge server e_m, and the vehicle users assigned to e_m are denoted by the set $ES_m = \{v_1, v_2, \cdots, v_k\}(0 \leq k \leq n)$ according to the priority (i.e., v_1 is the first to be processed). The data that arrives at RSUs with priority are processed first, and the queuing priority can be known according to the transmission delay. In addition, the RSU has limited computing power and cannot serve all the user task requests it receives. The queuing time strongly affects the service experience of the vehicle user. If the queuing time of the vehicle user is greater than the time of transmission of the vehicle to the cloud server (i.e., $T_n^{\text{Que}} > T_n^{\text{cloud}}$), the vehicle task is directly offloaded to the cloud server. The maximum wireless communication range of the RSU is assumed to be d ($d = 250m$). If there is no RSU within this distance, the vehicle offloads the task directly to the cloud server, and the vehicle offloaded to the cloud server is denoted as $CL = \{v_1, v_2, \cdots, v_c\}(0 \leq c \leq n)$. In addition, owing to the mobility of the vehicle, the vehicle's original location (denoted by v_o) and destination location (denoted by v_d) affect the selection of the computational task offloading method [22]. Assuming that the vehicle is moving in one direction along the road at a constant speed, the destination position v_d^n is calculated from the initial position v_o^n of vehicle v_n and its speed v:

$$v_d^n = v_o^n + \left(T_n^t + T_n^{Que} + T_n^c\right) \cdot v \tag{2}$$

 doi:10.20944/preprints202510.2415.v1

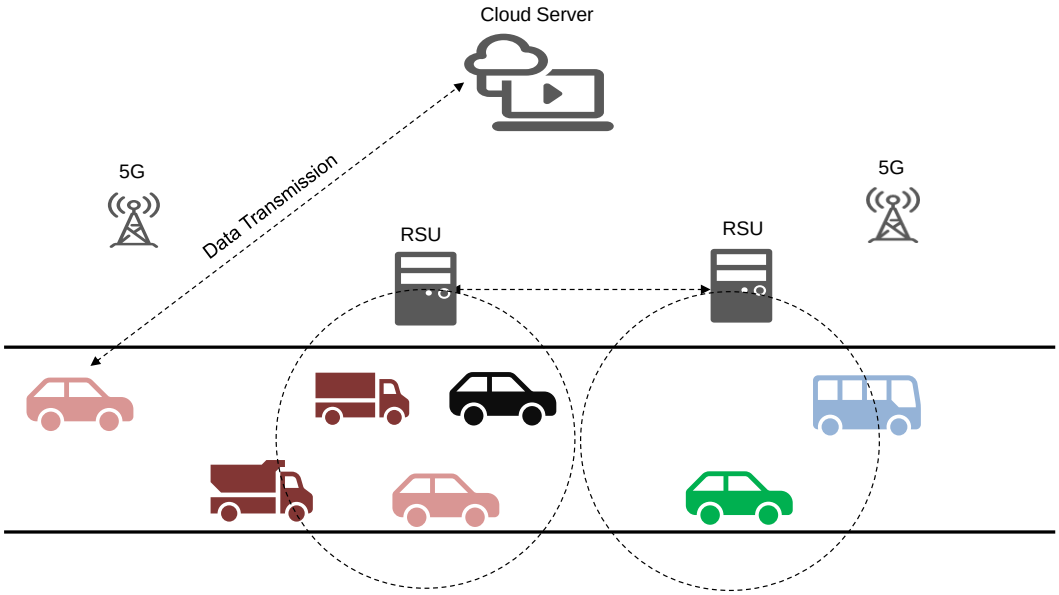


Figure 1. Vehicle-road cooperation based on edge-computing.

Assuming that v_o and v_d are in the same RSU coverage area d, v_o should offload the computational data to this RSU e_i for processing via wireless communication technology. At the end of the computation process, the data can be returned from e_i to the vehicle located at v_d. Assuming that the original and destination positions of the vehicle are within two RSUs e_i and e_j, respectively, the task is offloaded to the nearest RSU e_i that is connected to the v_o, and after the processing is completed, e_i transmits the data through multiple hops to e_j, which in turn transmits the processed data to the vehicle.

After processing the task, the RSU must transmit the processed data back to the vehicle. It determines whether the vehicle is still in the transmission range of RSU e_m after $T_n^t + T_{qc}$ time. If the vehicle is still in the transmission range, the data are directly transmitted back. Alternatively, if the vehicle is in the transmission range of another RSU e_q, e_m is transmitted to e_q via a multihop. Finally, if the vehicle is not in the transmission range of any of the RSUs, the processed data are sent via a multihop to the nearest RSU to the vehicle, and when the vehicle passes through the transmission range of that RSU, the data are transmitted back to the vehicle. Transmission between the RSUs is performed via a wireless backhaul link.

3.1.2. The RSUs Deployment Optimization Model

In this section, we focus on the optimization problem of RSU deployment in edge computing-based vehicle-road cooperation system. In this scenario, RSUs play a crucial role in that they not only reduce the computational burden of vehicular devices but also effectively scale down the transmission delay and guarantee real-time performance for high-bandwidth applications. Therefore, a six-objective optimization model that comprehensively considers total delay, energy consumption, security, load balancing, deployment cost, and coverage is constructed. The details are as follows:

1. Total delay
   In computationally intensive applications, the output data of a computation are often significantly smaller than the input data are. Therefore, the time required to return the computation results to the vehicle is negligible. If the vehicle transmits data to an RSU for processing, the user's time delay mainly consists of the transmission delay, queuing delay, and computation delay of the data in the RSU. If the vehicle transmits data to the cloud server for processing, the queuing time and the computation processing time are negligible because the cloud server has sufficient computational resources, and only the transmission delay needs to be considered.

 doi:10.20944/preprints202510.2415.v1

The time delay of the vehicle transmits data to an RSU: when vehicle v_n transmits task Z_n to RSU e_m, the transmission delay is generated at this time as follows:

$$T_n^t = \frac{b_n}{\lambda_{n,m}} \tag{3}$$

where $\lambda_{n,m}$ denotes the transmission rate (in bps) obtained by vehicle user v_n when it offloads the task to RSU e_m:

$$\lambda_{n,m} = W \times \log_2\left(1 + \frac{p_n \rho_0 l_{n,m}^{-\beta}}{\delta_0^2 + \sum_{v_i \in ES_m \cdot and v_i \neq v_n} p_i \rho_0 l_{i,m}^{-\beta}}\right) \tag{4}$$

where W is the channel bandwidth, which is set to 5 MHz. The transmission power p_n of the vehicle determines the strength of the signal sent by the vehicle, and the transmit power and idle power are set to 500 mW and 100 mW, respectively. The small-scale fading coefficient ρ_0 accounts for the rapid variation in the signal over short distances due to multipath effects, etc. The distance between vehicle user n and base station m is expressed as $l_(n, m)$, and β denotes the path loss coefficient (a constant), which is set to 3. δ_0^2 denotes the noise power spectral density, which is set to 10^{-9} W [29].

The vehicle's queuing delay in the RSUs: The vehicles in the set ES_m transmit messages to e_m at the same time and wait in line to be processed. The waiting time of vehicle v_n ($n > 1$) in e_m is computed as follows:

$$T_n^{\text{Que}} = \begin{cases} T_{n-1}^t + T_{n-1}^c + T_{n-1}^{Que} - T_n^t, & T_{n-1}^t + T_{n-1}^c + T_{n-1}^{Que} > T_n^t \\ 0, & T_{n-1}^t + T_{n-1}^c + T_{n-1}^{Que} \leq T_n^t \end{cases} \tag{5}$$

The vehicle's computation delay in the RSU: When a task arrives at the RSU, the RSU processes it, assuming that the computational power of the RSU is F_es, which is set to 5 GHz. The average computation time for all the vehicles in the system is

$$T_n^c = \frac{d_n}{F_{es}} \tag{6}$$

The vehicle's transmission delay in the cloud server: The time delay of the vehicle user v_n who is processing the task in the cloud server can be calculated as:

$$T_n^{\text{cloud}} = \frac{b_n}{\lambda_{n,\text{ cloud}}} \tag{7}$$

Therefore, the average time delay for all vehicles is:

$$\bar{T} = \frac{\sum_{j=1}^{m} \sum_{v_i \in ES_j} T_i^t + T_i^c + T_i^{Que} + \sum_{i \in CL} T_i^{cloud}}{n} \tag{8}$$

The objective function for minimizing the time delay can be normalized and expressed as:

$$\min f_{\text{delay}} = -\frac{1}{\bar{T}} \tag{9}$$

2. Energy consumption of vehicles

When user v_n transmits task Z_n to RSU e_m, the transmission energy consumption is:

$$E_n^{es} = p_n T_n^t = \frac{p_n b_n}{R_{n,m}} \tag{10}$$

 doi:10.20944/preprints202510.2415.v1

where p_n is set to 500 mW. The transmission energy consumption when user v_n transmits task Z_n to the cloud server is as follows:

$$E_n^{\text{cloud}} = \frac{p_n b_n}{R_{n,\text{cloud}}} \tag{11}$$

When the RSU processes the task of user v_n, user v_n is in the idle state, and the idle energy consumption of the vehicle can be calculated as follows:

$$E_n^c = p_{\text{idle}} T_n^c = \frac{p_{\text{idle}} d_n}{F_{es}} \tag{12}$$

where p_{idle} is the power consumption in the idle state, which is set to 100 mW. The cloud server has sufficient computational resources to process the task very quickly, so the idle energy consumption of the vehicle assigned to the cloud server is ignored, and only the transmission energy is consumed. In the last step, user v_n downloads the output data from the server. Since the size of the output data is much smaller than the size of the input data, the latency and energy consumption in this phase are intentionally ignored. The total energy consumption of the vehicle is defined as:

$$V_{\text{total}} = \begin{cases} E_n^{es} + E_n^c, \text{RSU} \\ E_n^{\text{cloud}}, \text{Cloud} \end{cases} \tag{13}$$

The average energy consumption objective function for all vehicles is:

$$E_{\text{total}} = \frac{\sum_{j=1}^{m} \sum_{v_i \in ES_j} E_n^{es} + E_n^c + \sum_{i \in CL} E_i^{\text{cloud}}}{n} \tag{14}$$

The objective function for minimizing energy consumption can be normalized and expressed as:

$$\min f_{\text{ene}} = -\frac{1}{E_{\text{total}}} \tag{15}$$

3. Security

Secure transmission is a key metric of QoS and is used to measure the probability of successful transmission. RSUs return data via multiple hops. Assuming that the probability that the link between two RSUs in the network is eavesdropped (i.e., transmitted packets may be intercepted) is p_d, the probability of successful forwarding between RSUs between each hop is $1 - p_d$. The probability of being compromised by an adversary is greater if the return path is longer. Therefore, the number of RSU hops passed through when returning data is used as a measure of security. If the vehicle's original and destination locations are within two RSUs e_m and e_i, respectively, and j RSUs need to be passed between e_m and e_i, the probability that e_m forward the data successfully to vehicle v_n is as follows:

$$p_n^m = (1 - p_d)^{j+1} \tag{16}$$

where $p_d = 10^{-9}$.

Therefore, the average safe forwarding rate for all vehicles is denoted as:

$$V_{\text{sec}} = \frac{\sum_{j=1}^{m} \sum_{v_i \in ES_j} p_i^J}{\sum_{j=1}^{m} \sum_{v_i \in ES_j} 1} \tag{17}$$

The minimization objective function is expressed as follows:

$$\min f_{\text{sec}} = 1 - V_{\text{sec}} \tag{18}$$

4. Load balancing of RSUs

 doi:10.20944/preprints202510.2415.v1

Load balancing improves both the availability of RSUs, by optimizing their resource allocation, and the reliability of the network by effectively handling congestion and failure events. The average load of the RSUs is denoted as:

$$\bar{C} = \frac{\sum_{i=1}^{m} |ES_i|}{m} \tag{19}$$

The vehicles covered by each RSU should be the same as possible. When an excessive number of vehicles are connected to certain RSUs, it often results in in a greater level of interference in vehicle data transmission. This also leads to stress overload in busy task RSUs and resource waste in idle RSUs. The standard deviation of the RSU load is used to denote the balanced load among them:

$$Std = \sqrt{\frac{\sum_{i=1}^{m} (|ES_i| - |\bar{C}|)^2}{m}} \tag{20}$$

The balanced load of the RSUs is normalized and expressed as a minimization objective function:

$$\min f_{\text{load}} = 1 - \frac{1}{1 + std} \tag{21}$$

5. Cost of deploying RSUs
The RSU cost is related mainly to the number of RSUs, so the minimized RSU cost is denoted as:

$$\min f_{\text{cost}} = \frac{N}{12} \tag{22}$$

6. Coverage area of RSUs
The more vehicles that RSUs can communicate with, the greater the coverage of the car network. Therefore, the coverage should be maximized and denoted as:

$$\text{Cov} = \frac{\sum_{i=1}^{m} |ES_i|}{n} \tag{23}$$

The objective function is expressed as follows:

$$\min f_{\text{cov}} = 1 - \text{Cov} \tag{24}$$

7. Six-objective optimization model
In summary, we construct a six-objective optimization model. That is, minimize total delay, energy consumption, balanced load, and deployment cost, and maximize security and coverage. The six-objective optimization model based on RSUs deployment is shown below:

$$\min \left\{ f_{\text{delay}}, f_{\text{ene}}, f_{\text{sec}}, f_{\text{load}}, f_{\text{cost}}, f_{\text{cov}} \right\} \tag{25}$$

3.2. The Proposed WGTwo_Arch2 Approach

This section presents the solution framework for the six-objective optimization model proposed in Section III. This approach is based on the Two_Arch2 algorithm. The proposed WGAN-based population initialization strategy and mating selection strategy, Levy distribution-based polynomial variation strategy and adaptive Lp-norm diversity archive updating strategy are first introduced, and then the proposed improvements are incorporated into the Two_Arch2 optimization framework to form the WGTwo_Arch2 algorithm. The pseudocode is shown in Algorithm 2. Finally, the superiority of WGTwo_Arch2 is verified by analyzing the experimental results through the objective function value and performance evaluation metrics.

 doi:10.20944/preprints202510.2415.v1

3.2.1. WGAN-Based Population Initialization Strategy and Mating Selection Strategy

GANs [27] mainly consist of a generator and a discriminator. The generator receives random noise vectors and generates new samples similar to the real data through an inverse convolutional network. The discriminator is a binary classifier that extracts features of the input data on the basis of a convolutional network, aiming at distinguishing whether the input samples are generated data or real data. The generator and discriminator are trained in an alternating adversarial game process, where the competition between the generator and the discriminator leads to an equilibrium process as the training progresses. Once training is complete, the generator can be used to generate high-quality synthetic data such that the discriminator cannot effectively distinguish between true and false data. The Jensen-Shannon (JS) scatter or Kullback-Leibler (KL) scatter used by traditional GANs as a loss function is as follows:

$$D_{\text{loss}} = -\frac{1}{m}\sum_{i=1}^{m}\left[\log\left(D\left(x^{(i)}\right)\right) + \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right)\right] \tag{26}$$

$$G_{\text{loss}} = -\frac{1}{m}\sum_{i=1}^{m}\log\left(D\left(G\left(z^{(i)}\right)\right)\right) \tag{27}$$

where D_loss is the discriminator loss, G_loss is the generator loss,$D(x)$ denotes the judgment probability of the discriminator on a real sample, and D(G(z)) denotes the judgment probability on a generated sample. With respect to the original loss function, problems such as training instability exist. The WGAN [28] introduces the Wasserstein distance as a loss function to improve the stability of the gradient, providing a better measure of the difference between the two distributions and makes it easier to generate high-quality data samples:

$$WD_{\text{loss}} = -\left(E_{x\sim p_{\text{data}}}[\mathrm{D(x)}] - E_{z\sim p_z}[\mathrm{D(G(z))}]\right) \tag{28}$$

$$WG_{\text{loss}} = -E_{z\sim p_z}[\mathrm{D(G(z))}] \tag{29}$$

where WD_{loss} is the discriminator loss, i.e., maximizing the difference between the real sample and the generated sample, WG_{loss} is the generator loss, i.e., minimizing the Wasserstein distance between the generated sample and the real sample, $D(x)$ denotes the judgment probability of the discriminator on the real sample, and D(G(z)) denotes the judgement probability on the generated sample. In the Two_Arch2 algorithm, the initial population is generated through randomization. However, this may lead to a nonuniform distribution of the initial population, thus reducing the population diversity and introducing high levels of uncertainty into the optimization process. The chaotic system is described as a nonlinear, stochastic-like deterministic bounded system that is neither periodic nor convergent and is highly sensitive to its initial parameters and conditions. Owing to its regularity, stochasticity, ergodicity and unpredictability, chaos is regarded as a reliable source of stochasticity that provides a more efficient search strategy for heuristic optimization algorithms than traditional random sequences do. The Kent chaotic map is a type of nonlinear chaotic mapping with a wireless number of mapping folding, which has good traversability, uniformity and randomness and can effectively prevent the algorithm from falling into local extremes. The Kent map is used to generates the solution sequence:

$$x_{i+1} = \begin{cases} \frac{x_i}{a}, & 0 \le x_i \le a \\ \frac{1-x_i}{1-a}, & a < x_i < 1 \end{cases} \tag{30}$$

where a is a control parameter, $a \in (0,1)$, which is set to a value of 0.4. The range of its chaotic orbital state values is (0, 1). Therefore, the population initialization formula is:

$$x_{i+1} = \begin{cases} \frac{x_i}{a} \times \left(\text{value}_{\text{high}} - \text{value}_{\text{low}}\right) + \text{value}_{\text{low}} \\ \frac{1-x_i}{1-a} \times \left(\text{value}_{\text{high}} - \text{value}_{\text{low}}\right) + \text{value}_{\text{low}} \end{cases} \tag{31}$$

 doi:10.20944/preprints202510.2415.v1

where value $_{high}$ and value $_{low}$ are the maximum and minimum values of the decision variables, respectively. we propose a Kent map-based WGAN initialization strategy to improve the uniformity of the distribution of individuals in the population and effectively enhance the search efficiency of the algorithm. The pseudocode is shown in Algorithm 2.

Algorithm 2: Population initialization strategy based on WGAN

Input: The dataset of size N generated by the Kent map;
Output: The initial population with N solutions generated by WGAN;
Train generators and discriminators of WGAN on the data generated by Kent map;
for i $<$ N **do**
    $Z_j \leftarrow$ Generate random noise;
    $x_i \leftarrow$ WGAN generates candidate solutions based on noise;
POP $\leftarrow \{x_1, x_2, \ldots, x_N\}$;
return POP;

Mating selection is employed to produce the next generation by selecting certain individuals from the CA and DA for crossover and mutation. An effective mating selection strategy for a population is crucial for maintaining the genetic diversity of the population and driving the search process toward the Pareto optimal frontier. In the Two_Arch2 algorithm, a random selection strategy and a Pareto dominance selection strategy are utilized to select individuals for offspring generation. To increase the diversity of the solutions, we propose a Pareto dominance selection strategy based on the WGAN, i.e., we use the nondominated solutions in the CA to train the generator and discriminator of the WGAN, and the WGAN-generated individuals are selected for mating. The pseudocode is shown in Algorithm 3.

Algorithm 3: The mating selection strategy based on WGAN

Input: CA,DA,Population size N;
Output: ParentC, ParentM;
ParentM $\leftarrow$ N - individuals randomly selected from CA;
ParentC $\leftarrow$ N/2 individuals randomly selected from DA;
NonDominate $\leftarrow$ N/2 individuals randomly selected from CA;
Training generators and discriminators of WGAN using nondominated solutions;
ParentC $\leftarrow$ N/2 solutions are generated using WGAN;
return ParentC and ParentM;

3.2.2. Polynomial Variation Strategy Based on the Levy Distribution

Variation operations are an important way for individuals in a population to explore the solution space. Polynomial variation is a probability-based variation operation that generates new individuals by randomly perturbing individual genes, effectively introducing diverse individuals and improving the global search ability of the algorithm. The Two_Arch2 algorithm enables better searches for the optimal solution by polynomially varying the individuals. Specifically, chromosomal variation is achieved by first randomly generating a variation parameter u, randomly selecting an individual for a particular gene, and then summing the variation value with the value of that gene. The parameters of polynomial variation include the variation probability, perturbation size and perturbation index. Among them, the mutation probability determines the probability of each individual being mutated,

 doi:10.20944/preprints202510.2415.v1

the perturbation size determines the size of the perturbation, and the perturbation index determines the direction of the perturbation. The mutation formula is as follows:

$$x_t' = \begin{cases} x_t + (u_t - l_t)\left[2r + (1-2r)\left(\frac{u_t - x_t}{u_t - l_t}\right)^{d+1}\right]^{\frac{1}{d+1}-1}, & r \le 0.5 \\ x_t + (u_t - l_t)\left[1 - \left[2(1-r) + 2(r-0.5)\left(\frac{x_t - l_t}{u_t - l_t}\right)^{d+1}\right]^{\frac{1}{d+1}}\right], & r > 0.5 \end{cases} \tag{32}$$

where x_t denotes a randomly selected locus, x_t' denotes the mutated locus, and r denotes a random number uniformly distributed in the interval (0,1). d denotes the probability of mutation, and u_t and l_t denote the upper and lower bounds of the variable, respectively.

To avoid blindly applying variation operations, the variation parameter r in polynomial variation is randomly generated. A polynomial variation strategy based on the Levy distribution is adopted to reasonably determine the variation range of the population. The step length in Levy flight is generated from the Levy distribution, which produces a random move distance in each step and occasionally has very long step lengths because of its heavy-tailed nature. Therefore, generating the variational parameter r from the Levy flight strategy not only increases the search region of the solution but also enhances the algorithm's ability to search for the solution. This strategy enables the algorithm to avoid falling into local optima, thus increasing its probability of finding the global optimum solution. The formula for generating the variational parameter r using the Levy flight strategy is as follows:

$$\sigma = \left(\frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\pi\beta\Gamma((1+\beta)/2)} \times \frac{2^{(\beta-1)/2}}{\beta}\right)^{1/\beta} \tag{33}$$

$$u = \mathrm{norm}(0, \sigma) \tag{34}$$

$$v = \mathrm{norm}(0, 1) \tag{35}$$

$$r = \frac{u}{|v|^{1/\beta}} \tag{36}$$

where Γ is the gamma function, β is the shape parameter of the step size distribution with a value of 1.5, and u and v denote the standard normal distributions $N(0, \quad \sigma^2)$ and $N(0,1)$ of random numbers, respectively.

3.2.3. Diversity Archive Update Based on the Adaptive Lp-Norm

The DA in Two_Arch2 complements the CA, which guides the population to converge to the PF. In contrast, the DA focuses on maintaining the diversity of the population, and its update is based on the Pareto dominance strategy. However, unlike traditional algorithms that use the crowding distance as a criterion for environment selection, the DA uses the Lp-norm-based distance as a similarity measure between solutions, prioritizing the retention of solutions in sparse regions. The setting of p can affect the evaluation of diversity and thus the distribution of solutions in the target space. In the original DA update strategy, p is set to 1/m (m is the number of targets). Although the distance based on the Lp-norm can largely improve the diversity of the algorithm, it is not conducive to the utilization of its own characteristics if it is used indiscriminately for all individuals in each iteration. Therefore, in this paper, the value of p is dynamically adjusted on the basis of the number of iterations. Therefore, the algorithm has a better global search ability to cover the potentially optimal region at the early stage of optimization and focuses on local depth-seeking at the later stage to achieve a wide distribution of the solution set and ensure the balance between exploration and exploitation of the algorithm, with the following formula:

$$p = \frac{1}{1 + e^{p \times \frac{t}{T}}} \tag{37}$$

where T is the maximum number of iterations and t is the current number of iterations.

 doi:10.20944/preprints202510.2415.v1

4. Experimental Evaluation

This section first describes the parameter settings of the edge computing-based vehicle-road cooperation architecture. Then, a comparison of the proposed algorithm with MaOEAIGD [30], TiGE2 [31], hpaEA [32], Two_Arch2 [12], and NSGA-III [28] on the basis of the values of hypervolume (HV) metric and the six optimization objectives proposed in this section.

4.1. Experimental Parameter Settings

Assuming that there are 50 vehicles on a one-way road with a range of 3 km, we set up experiments with 1 to 12 RSUs. We use four optimization algorithms to solve the RSU deployment optimization problem. Each algorithm is run 20 times, and the objective function is evaluated 100000 times. The other experimental parameters are shown in Table 1.

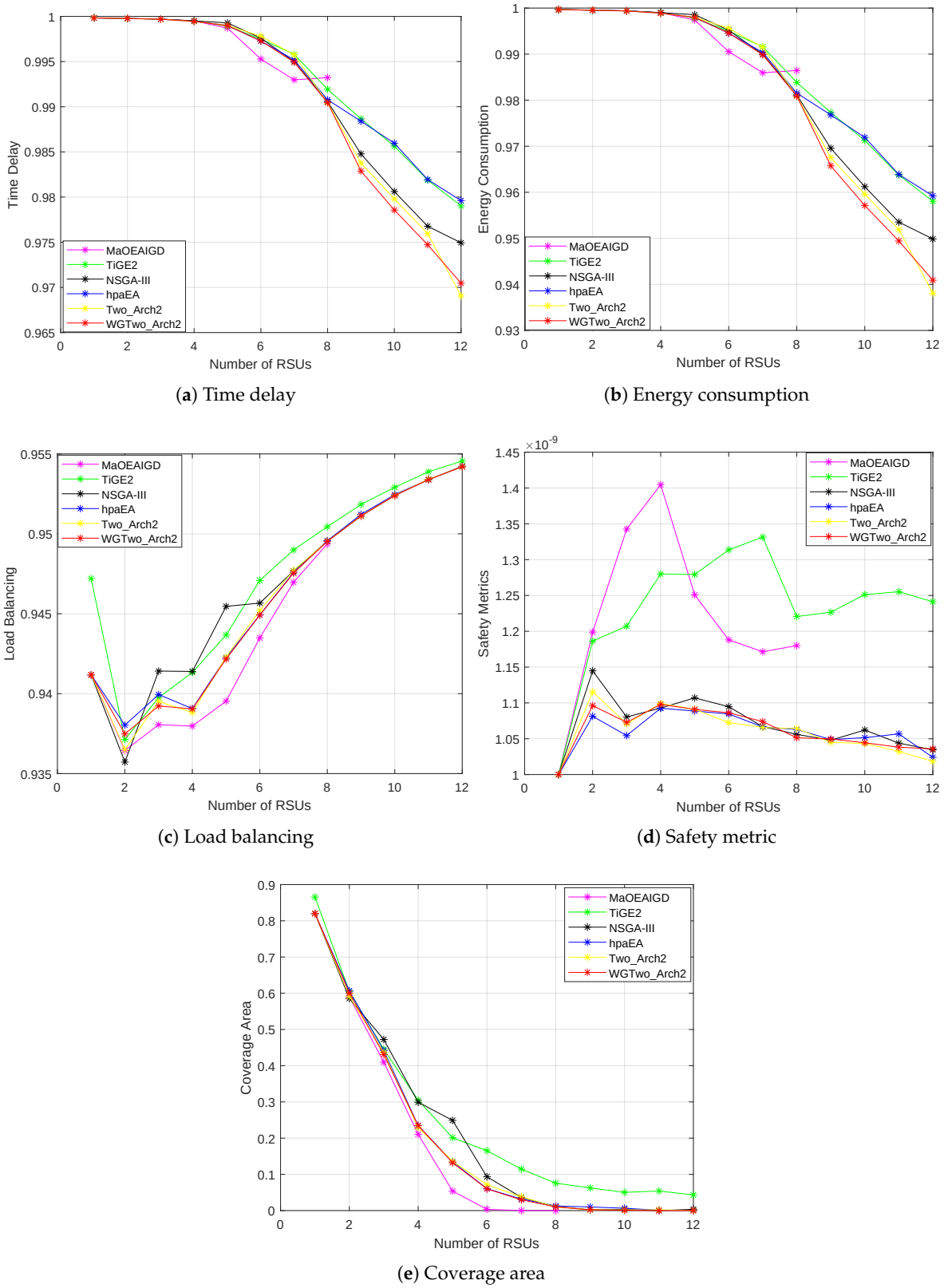
Table 1. Parameter settings.

Symbol	Description	Value
n	the maximum number of RSUs	12
m	the number of the vehicles	50
W	channel bandwidth	5MHz
b_n	the size of the input data	$[400, 600]$kbits
d_n	the required total number of CPU cycles	$[900, 1100]$megacycles
p_n	vehicle transmission power	500mW
$p_i dle$	vehicle transmission power	100mW
ρ_0	small-scale fading coefficient	10^{-5}
β	Path loss coefficient	3
d	the maximum communication range	250m
δ_0^2	the noise power spectral density	10^{-9}W
p_d	Probability of being eavesdropped	10^{-9}
$maxFE$	Maximum evaluation times	100000

4.2. Comparison and Analysis of Experimental Results

The variability in the number of RSUs deployed is analyzed on the basis of the set of Pareto optimal solutions obtained. Figure 2 presents a visualization of the experimental results for the the six algorithms on each optimization objective (the smaller the objective value is, the better the optimization). As shown in Figure 2 (a) (b) (d), the performance of the solution obtained by the improved algorithm WGTwo_Arch2 is significantly better than that of the remaining five algorithms on three objectives, time delay, energy consumption and security. The performance of the six algorithms under the load balancing and coverage area objective is shown in Figure 2 (c) (e), where the WGTwo_Arch2 algorithm is optimized after MaOEA-IGD. However, MaOEA-IGD can only find solution sets with RSU numbers less than 9, which has significant shortcomings in terms of diversity. In summary, the WGTwo_Arch2 algorithm demonstrates superior performance in solving the many-objective deployment optimization problem for RSUs in the IoV, achieving low latency, effectively reducing the energy overhead of the system and ensuring the security of data transmission.

 doi:10.20944/preprints202510.2415.v1



(**a**) Time delay

(**b**) Energy consumption

(**c**) Load balancing

(**d**) Safety metric

(**e**) Coverage area

Figure 2. Comparison of average objective values between WGTwo_Arch2 and other algorithms.

4.3. Comparison of HV Indicator Values

The HV, as a comprehensive evaluation metric, can be used to assess the performance of algorithm-generated PFs effectively in terms of diversity and convergence. This metric quantifies the performance of an algorithm by calculating the volume of the space enclosed by the PFs with respect to a predefined

 doi:10.20944/preprints202510.2415.v1

reference point. This reference point is usually set as the point of minimum value in each target direction, which serves as a benchmark for evaluation. If a larger value of HV is obtained during the calculation process, then the algorithm usually performs better in terms of convergence and diversity. The HV is calculated via the following equation:

$$HV(ps, ref) = VOL\left\{ \bigcup_{f \in ps} \left[f_1, f_1^{ref}\right] \times \ldots \times \left[f_m, f_m^{ref}\right] \right\} \tag{38}$$

where ps denotes the set of optimal solutions obtained by the algorithm, which contains solutions that perform well under multiple objective functions. ref is the set reference point, which is usually chosen as the minimum value point in each objective direction as the benchmark for calculating the hypervolume. m denotes the number of objectives, which determines the dimensions of the objective space that need to be accounted for during HV calculation. In this paper, the proposed algorithm is compared with the other five algorithms in terms of the HV term. The number of evaluations is set to 100000, and each algorithm is tested independently over 20 runs. The final HV value is computed as the average of these 20 runs. As shown in Table 2 and Figure 3, the HV metric value of the WGTwo_Arch2 algorithm is the largest, indicating its superior performance in solving the many-objective RSU deployment optimization problem.

Table 2. HV index values.

Algorithm	HV
WGTwo_Arch2	1.384E − 03
Two_Arch2	1.282E − 03
NSGA-III	1.240E − 03
hpaEA	1.243E − 03
TiGE2	1.297E − 03
MaOEA-IGD	1.037E − 03

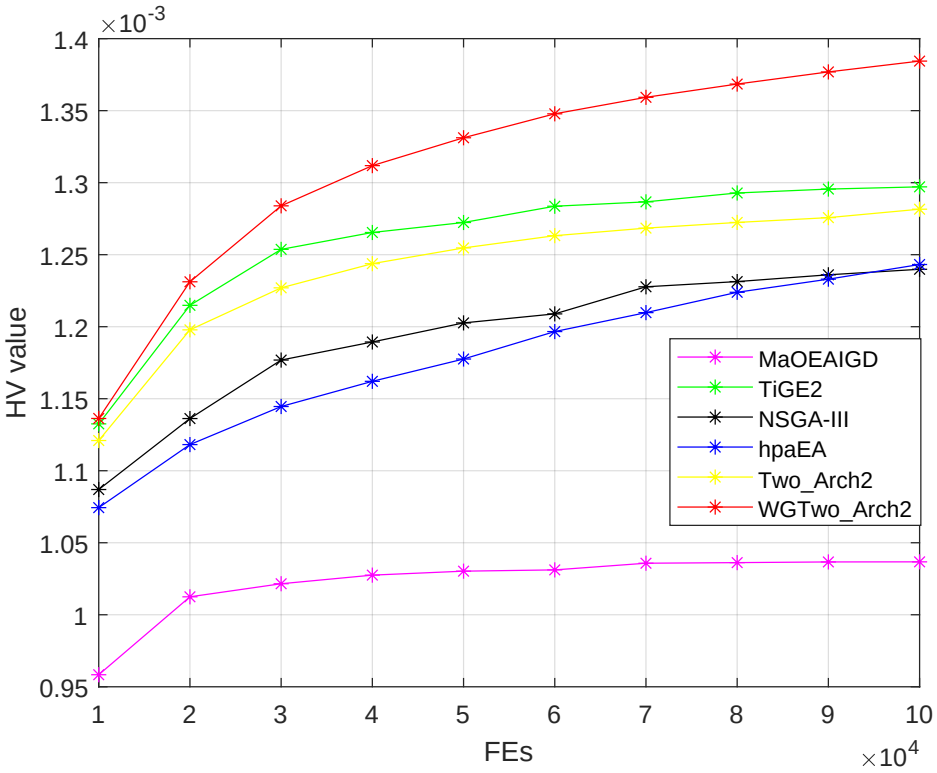


Figure 3. The HV value evolutionary curves of algorithms for the six-objective optimization model.

 doi:10.20944/preprints202510.2415.v1

5. Conclusions

Efficient IoV networks require optimal RSU deployment for a given road layout, which can reduce network latency and improve data transmission efficiency. However, deploying and maintaining RSUs is costly. Therefore, maximizing the QoS with a limited budget is a major challenge. In this paper, we describe the RSU deployment problem in edge computing-based vehicle-road cooperation and model it as a deployment optimization problem with multiple objectives. To evaluate the performance of the system architecture comprehensively, a six-objective optimization model is constructed that integrates several key factors, such as time delay, energy consumption, security, load balancing, deployment cost and coverage area. The WGTwo_Arch2 algorithm is subsequently proposed by incorporating the WGAN-based population initialization strategy and mating selection strategy, the Kent map-based population initialization strategy, the polynomial variation strategy based on the Levy distribution, and the diversity archive updating strategy based on the adaptive Lp norm into the Two_Arch2 framework. The execution process and pseudocode of the proposed enhancements are detailed. Finally, by comparing the objective function values and performance evaluation indices, it is verified that the WGTwo_Arch2 algorithm clearly outperforms the other algorithms in several aspects.

Author Contributions: Methodology, S.F.; software, S.F.; validation, S.F. and B.C.; formal analysis, S.F.; investigation, S.F.; writing—original draft preparation, S.F.; writing—review and editing, S.F.; visualization, S.F.; supervision, B.C.; project administration, B.C.; funding acquisition, B.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Key R&D Program of China under Grant No. 2023YFB4503000. This work was supported in part by National Natural Science Foundation of China (NSFC) under Grant No. 62473129, in part by the Natural Science Fund of Hebei Province for Distinguished Young Scholars under Grant No. F2021202010, in part by Science and Technology Project of Hebei Education Department under Grant No. JZX2023007, in part by the S&T Program of Hebei under Grant No. 24464401D. The APC was funded by the same grant.

Data Availability Statement: Data will be made available on request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Fan, X.; Gu, W.; Long, C.; Gu, C.; He, S. Optimizing task offloading and resource allocation in vehicular edge computing based on heterogeneous cellular networks. IEEE Transactions on Vehicular Technology 2023, 73, 7175-7187.
2. Ning, Z.; Huang, J.; Wang, X.; Rodrigues, J. J.; Guo, L. Mobile edge computing-enabled internet of vehicles: Toward energy-efficient scheduling. IEEE Network 2019, 33, 198-205.
3. Huo, Y.; Yang, R.; Jing, G.; Wang, X.; Mao, J. A multi-objective Roadside Units deployment strategy based on reliable coverage analysis in Internet of Vehicles. Ad Hoc Networks 2024, 164, 103630.
4. Massobrio, R.; Toutouh, J.; Nesmachnow, S.; Alba, E. Infrastructure deployment in vehicular communication networks using a parallel multiobjective evolutionary algorithm. International Journal of Intelligent Systems 2017, 32, 801-829.
5. Zhang, K.; Mao, Y.; Leng, S.; He, Y.; Zhang, Y. Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading. IEEE vehicular technology magazine 2017, 12, 36-44.
6. Nikookaran, N.; Karakostas, G.; Todd, T. D. Combining capital and operating expenditure costs in vehicular roadside unit placement. IEEE Transactions on Vehicular Technology 2017, 66, 7317-7331.
7. Yu, H.; Liu, R.; Li, Z.; Ren, Y.; Jiang, H. An RSU deployment strategy based on traffic demand in vehicular ad hoc networks (VANETs). IEEE Internet of Things Journal 2021, 9, 6496-6505.
8. Talpur, A.; Gurusamy, M. Drld-sp: A deep-reinforcement-learning-based dynamic service placement in edge-enabled internet of vehicles. IEEE Internet of Things Journal 2021, 9, 6239-6251.
9. Lin, M.; Huo, J.; Wang, L.; Yao, G.; Chen, Y.; Lu, Z. A Multi-Objective Optimization Approach for Roadside Unit Deployment Strategy in IoV. In 2024 IEEE Wireless Communications and Networking Conference (WCNC), Dubai, United Arab Emirates, 14-17 April 2024; pp. 1-6.

 doi:10.20944/preprints202510.2415.v1

10. Zhang, L.; Wang, L.; Zhang, L.; Zhang, X.; Sun, D. An RSU deployment scheme for vehicle-infrastructure cooperated autonomous driving. Sustainability 2023, 15, 3847.
11. Chaudhary, R.; Kumar, N. Secgreen: Secrecy ensured power optimization scheme for software-defined connected iov. IEEE Transactions on Mobile Computing 2023, 22, 2370–2386.
12. Wang, H.; Jiao, L.; Yao, X. Two_Arch2: An improved two-archive algorithm for many-objective optimization. IEEE transactions on evolutionary computation 2014, 19, 524-541.
13. Praditwong, K.; Yao, X. A new multi-objective evolutionary optimisation algorithm: The two-archive algorithm. In 2006 International Conference on Computational Intelligence and Security, Guangzhou, China, 2006; pp. 286-291.
14. Zitzler, E.; Künzli, S. Indicator-based selection in multiobjective search. In International conference on parallel problem solving from nature, Berlin, Heidelberg, September 2004; pp. 832-842.
15. Deb, K.; Goyal, M. A combined genetic adaptive search (GeneAS) for engineering design. Computer Science and informatics 1996, 26, 30-45.
16. Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Bengio, Y. Generative adversarial nets. Advances in neural information processing systems 2014, 27.
17. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein generative adversarial networks. In International conference on machine learning, Long Beach, California, USA, July 2017; pp. 214-223.
18. Chen, F.; Zhou, J.; Xia, X.; Xiang, Y.; Tao, X.; He, Q. Joint optimization of coverage and reliability for application placement in mobile edge computing. IEEE Transactions on Services Computing 2023, 16, 3946-3957.
19. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A. M. T. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation 2002, 6, 182-197.
20. Kumrai, T.; Ota, K.; Dong, M.; Champrasert, P. RSU placement optimization in vehicular participatory sensing networks. In 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Toronto, ON, Canada, April 2014; pp. 207-208.
21. Wang, F.; Wang, C.; Wang, K.; Jiang, Q.; Wang, B.; He, W. Multiobjective differential evolution with discrete elite guide in internet of vehicles roadside unit deployment. Wireless Communications and Mobile Computing 2021, 1, 4207130.
22. Guerna, A.; Bitam, S. GICA: An evolutionary strategy for roadside units deployment in vehicular networks. In 2019 International Conference on Networking and Advanced Systems (ICNAS), Annaba, Algeria, June 2019; pp. 1-6.
23. Wu, T. J.; Liao, W.; Chang, C. J. A cost-effective strategy for road-side unit placement in vehicular networks. IEEE Transactions on Communications 2012, 60, 2295-2303.
24. Ghosh, D.; Katehara, H.; Rawlley, O.; Gupta, S.; Arulselvan, N.; Chamola, V. Artificial intelligence-empowered optimal roadside unit (RSU) deployment mechanism for internet of vehicles (IoV). In 2022 IEEE 23rd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Belfast, United Kingdom, June 2022; pp. 495-500.
25. Huo, Y.; Yang, R.; Jing, G.; Wang, X.; Mao, J. A multi-objective Roadside Units deployment strategy based on reliable coverage analysis in Internet of Vehicles. Ad Hoc Networks 2024, 164, 103630.
26. Lu, J.; Jiang, J.; Balasubramanian, V.; Khosravi, M. R.; Xu, X. Deep reinforcement learning-based multi-objective edge server placement in Internet of Vehicles. Computer communications 2022, 187, 172-180.
27. Shen, B.; Xu, X.; Qi, L.; Zhang, X.; Srivastava, G. Dynamic server placement in edge computing toward internet of vehicles. Computer Communications 2021, 178, 114-123.
28. Deb, K.; Jain, H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. IEEE transactions on evolutionary computation 2013, 18, 577-601.
29. Nath, S.; Li, Y.; Wu, J.; Fan, P. Multi-user multi-channel computation offloading and resource allocation for mobile edge computing. In ICC 2020-2020 IEEE International Conference on Communications (ICC), Dublin, Ireland, June 2020; pp. 1-6.
30. Sun, Y.; Yen, G. G.; Yi, Z. IGD indicator-based evolutionary algorithm for many-objective optimization problems. IEEE Transactions on Evolutionary Computation 2018, 23, 173-187.

 doi:10.20944/preprints202510.2415.v1

31. Zhou, Y.; Zhu, M.; Wang, J.; Zhang, Z.; Xiang, Y.; Zhang, J. Tri-goal evolution framework for constrained many-objective optimization. IEEE Transactions on Systems, Man, and Cybernetics: Systems 2018, 50, 3086-3099.
32. Chen, H.; Tian, Y.; Pedrycz, W.; Wu, G.; Wang, R.; Wang, L. Hyperplane assisted evolutionary algorithm for many-objective optimization problems. IEEE Transactions on Cybernetics 2019, 50, 3367-3380.