Energy-minimized Partial Computation Offloading in Cloud-assisted Vehicular Edge Computing Systems

Ziqi Wang
Faculty of Information Technology
Beijing University of Technology
Beijing, China
ziqi_wang@emails.bjut.edu.cn

Xinyang Li
Faculty of Information Technology
Beijing University of Technology
Beijing, China
lixinyang020826@126.com

Jiayi Wang
Faculty of Information Technology
Beijing University of Technology
Beijing, China
jacklinwang0109@gmail.com

Jiahui Zhai
Faculty of Information Technology
Beijing University of Technology
Beijing, China
zhaijiahui@emails.bjut.edu.cn

Shichao Chen
Institute of Automation
Chinese Academy of Sciences
Beijing, China
shichao.chen@ia.ac.cn

Jing Bi
Faculty of Information Technology
Beijing University of Technology
Beijing Huairou Academy of Parallel Sensing
Beijing, China
bijing@bjut.edu.cn

2023 International Conference on Networking, Sensing and Control (ICNSC) | 979-8-3503-6950-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICNSC58704.2023.10318997

***Abstract*—Nowadays, the rapid advancement of Connected and Automated Vehicles (CAVs) has led to their integration with various capabilities, encompassing environmental sensing, decision-making, and multi-level assisted driving. However, the integration of computationally intensive applications like navigation and autonomous driving challenges CAVs due to their limited computational resources, necessitating the timely completion of computations. Vehicular Edge Computing (VEC) offers a solution by enabling CAVs to partially offload computation-intensive tasks to Roadside Units (RSUs) embedded with Roadside Edge Servers (RESs). Nonetheless, RSUs have finite computational resources. Therefore, a Cloud-assisted Vehicular Edge Computing (CVEC) architecture is introduced to address this problem. In this paper, we first formulate a typical CVEC system and then formulate a constrained optimization problem based on the aforementioned system, which considers both communication latency and energy consumption. Finally, a novel optimization algorithm called Whale optimization embedded with Simulated-Annealing and Genetic-learning (WSAG) is proposed to solve the above optimization problem. WSAG simultaneously determines the resource allocation and optimizes the energy consumption of the system. Experiment results prove that WSAG significantly achieves lower energy consumption with faster convergence speed than state-of-the-art peers.**

***Index Terms*—Vehicular edge computing, computation offloading, cloud computing, intelligent optimization algorithms**

I. INTRODUCTION

In recent years, Connected and Automated Vehicles (CAVs) have attracted increasing attention in industry [1] [2]. However, their limited computing resources pose challenges, especially for computation-intensive tasks. To address this problem, two solutions are proposed: Mobile Cloud Computing (MCC) [3] and Vehicular Edge Computing (VEC) [4]. The former one utilizes remote Cloud Data Centers (CDCs) to enhance the computing performance of CAVs but suffer from delays and privacy concerns. The latter one offloads computationally intensive tasks to Roadside Units (RSUs) embedded with Roadside Edge Servers (RESs) that offer more computing capabilities. However, RSUs have limited resources compared to CDC. Based on the aforementioned analysis, a Cloud-assisted VEC (CVEC) system is constructed to handle computation-intensive CAV tasks with strict latency requirements [5]. Two key challenges remain, the first one is to optimize resource allocation [6] for timely task execution and the second one is to minimize energy consumption of the CVEC system [7].

Motivated by the above analysis, this work proposes an energy-minimized computation offloading technique within a fundamental unit architecture that includes a moving CAV, RSUs, and a CDC. A constrained optimization problem is formulated and a novel optimization algorithm called Whale optimization embedded with Simulated-Annealing and Genetic-learning (WSAG) is proposed to optimize energy consumption of the CVEC system. Experiments demonstrate WSAG's effectiveness in achieving energy-minimized computation offloading. Main contributions of this paper are summarized as follows.

1) A three-tier CVEC framework is constructed in this work. It consists of a dynamic CAV, multiple RSUs deployed with RESs, and a CDC. Moreover, tasks in this CAV can be split and dynamically offloaded to RSUs and/or CDC for parallel execution.
2) For the above CVEC system, a constrained Mixed Integer Nonlinear Programming problem (MINLP) is constructed. This problem takes into account the Central Processing Unit (CPU) computation frequency, data trans-

This work was supported in part by the Education and Teaching Research Projects of Beijing University of Technology (ER2022KCB05), Beijing Natural Science Foundation (L233005).

mission rate, as well as the upstream and downstream transmission channel resources of the CAVs, RSUs, and the CDC. The goal is to optimize the energy consumption of the system as well as satisfy the time limit defined by the user.

3) A novel optimization method called WSAG is presented to address the previously mentioned optimization challenge. It jointly optimizes task offloading and resource allocation among a CAV, RSUs, and the CDC, while ensuring that tasks are completed within the maximum constraint time.

II. Problem Formulation

To address the energy minimization challenge in the CVEC system, we propose a unit architecture as illustrated in Fig. 1. In this system, a CAV dynamically moves along a road and dynamically generates a set of K tasks, and partial offloads them to a RSU embedded with a RES. Each task is represented as a four-element tuple, *i,e.*, $\{\alpha_k, \beta_k, x_k, \delta_k\}$ where α_k denotes the size of task k, β_k denotes the processing density of task k (CPU cycles/bit), x_k denotes the memory density of task k (MB /bit) and δ_k denotes the ratio of returned data to input data after task k is executed at the edge. Moreover, each task in the CAV can be executed in CAV, a RSU and/or the CDC. Tasks are offloaded to RSU through wireless channels shared by the CAV. Additionally, tasks in RSU can be further offloaded to the CDC. Moreover, RSUs are connected to the CDC through high-speed fiber links. This work focuses on computation-intensive tasks that are independent of each other. For instance, navigation applications can be divided into multiple logically independent tasks, each of which can be executed in the CAV or RSU/CDC by using partial computation offloading.

In this proposed architecture, if the CAV drives into the coverage area of RSU i ($1 \le i \le N$), the CAV establishes a connection with this RSU, denoted as $\theta_i=1$; otherwise, $\theta_i=0$. Additionally, each task can be executed on CAV, RSU and/or the CDC. λ_0^k, λ_i^k and λ_{ic}^k denotes the proportions of task k executed in CAV, RSU i and the CDC. As a result, for task k offloaded to RSU i, the sum of λ_0^k, λ_i^k and λ_{ic}^k equals one, *i.e.*,

$$\lambda_0^k+\lambda_i^k+\lambda_{ic}^k=1 \tag{1}$$

Therefore, all tasks executed on the CAV should not exceed its maximum CPU capacity ($\hat{C}_L$), memory capacity ($\hat{M}_L$), and its maximum CPU running speed ($\hat{F}_0$), *i,e.*,

$$\sum_{k=1}^{K}\alpha_k\lambda_0^k\beta_k \le \hat{C}_L, \quad \sum_{k=1}^{K}\alpha_k\lambda_0^k x_k \le \hat{M}_L, \quad \sum_{k=1}^{K} f_0^k \le \hat{F}_0 \tag{2}$$

where f_0^k denotes the CPU running speed of a CAV for task k.

All tasks executed on the RSU i should not exceed its maximum CPU capacity ($\hat{C}_i$) and memory capacity ($\hat{M}_i$), as well as its maximum CPU running speed ($\hat{F}_i$), *i,e.*,

$$\sum_{k=1}^{K}\theta_i\alpha_k\lambda_i^k\beta_k \le \hat{C}_i, \quad \sum_{k=1}^{K}\theta_i\alpha_k\lambda_i^k x_k \le \hat{M}_i, \quad \sum_{k=1}^{K}\theta_i f_i^k \le \hat{F}_i \tag{3}$$

where f_i^k denotes the CPU running speed of the RSU i for task k.

Similarly, all offloading tasks from RSU i processed on the CDC should not exceed its maximum CPU capacity ($\hat{C}_c$), memory capacity ($\hat{M}_c$), and its maximum CPU running speed ($\hat{F}_c$), *i,e.*,

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\theta_i\alpha_k\lambda_{ic}^k\beta_k \le \hat{C}_c, \sum_{i=1}^{N}\sum_{k=1}^{K}\theta_i\alpha_k\lambda_{ic}^k x_k \le \hat{M}_c, \sum_{i=1}^{N}\sum_{k=1}^{K}\theta_i f_{ic}^k \le \hat{F}_c \tag{4}$$

where f_{ic}^k denotes the CPU running speed of the CDC connected to RSU i for task k.

The following three parts will formulate a bandwidth, latency and energy model based on the proposed CVEC system.

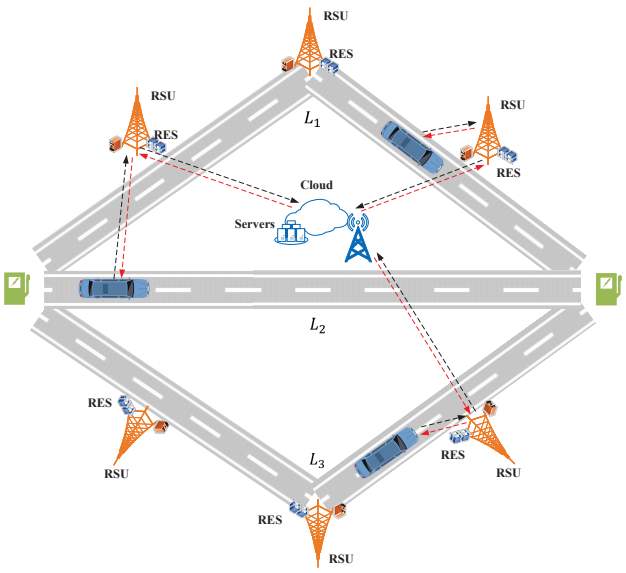

Fig. 1. Architecture of the CVEC system.

A. Bandwidth Model

This work considers one frequency division duplex mode in each uplink/downlink channel. Based on Shannon's theorem [8], the data transfer rate of CAV's k task on the uplink channel with SBS i ($\hat{r}_i^k$) can be calculated as:

$$\hat{r}_i^k=\theta_i W_1 log_2\left(1+\frac{P^L(d_i)^{-o}|\overline{f}|^2}{\omega_0}\right) \tag{5}$$

where W_1 denotes the bandwidth in RSU i's uplink channel used by the CAV. $\overline{f}$ denotes the fading coefficient of the uplink channel. ω_0 denotes the power parameter of white Gaussian noise, and d_i denotes the distance from the CAV to RSU i. P^L

represents the transmitting power of the CAV and is obtained as:

$$P^L{=}s_0(f_0^k)^3 \tag{6}$$

where s_0 is a constant determined by the chip architecture of the CAV.

Similarly, the data transfer rate of CAV's k task on the downlink channel with SBS i ($\check{r}_i^k$) can be obtained as:

$$\check{r}_i^k{=}\theta_i W_2 log_2(1{+}\frac{P^i(d_i)^{-o}|\widetilde{f}|^2}{\omega_0}) \tag{7}$$

where W_2 denotes the bandwidth in RSU i's downlink channel allocated for the CAV, and $\widetilde{f}$ denotes the fading coefficient of the downlink channel. P^i means the transmitting power of RSU i and it is obtained as:

$$P^i{=}s_i(f_i^k)^3 \tag{8}$$

where s_i is a constant and determined by the chip architecture of the RSU i.

B. Latency Model

The CVEC system consists of multiple time-consumption components, including computation and data transmission among CAV, RSU and the CDC. It is worth noting that tasks have to be completed within their delay limits and the coverage area of RSU. As a result, this section formulates the latency model of the system.

Let T_L^k denotes the execution time of task k on CAV and it is obtained as:

$$T_L^k{=}\frac{\alpha_k\lambda_0^k\beta_k}{f_0^k} \tag{9}$$

T_i^k denotes the total time of executing task k of CAV on RSU. It is obtained as:

$$T_i^k{=}\dot{T}_i^k{+}\ddot{T}_i^k \tag{10}$$

where $\dot{T}_i^k$ denotes the time of executing task k on RSU i and $\ddot{T}_i^k$ denotes the time of data uploading, downloading for task k between CAV and RSU i. They are obtained as follows:

$$\dot{T}_i^k{=}\frac{\theta_i\alpha_k\lambda_i^k\beta_k}{f_i^k} \tag{11}$$

$$\ddot{T}_i^k{=}\frac{\theta_i\alpha_k\lambda_i^k}{\hat{R}_i}{+}\frac{\theta_i\alpha_k\lambda_i^k}{\check{R}_i} \tag{12}$$

where $\hat{R}_i$ and $\check{R}_i$ denote the total upload/download transfer rate from CAV to RSU i, and they are obtained as follows:

$$\hat{R}_i{=}\sum_{k=1}^{K}\lambda_i^k\hat{r}_i^k \tag{13}$$

$$\check{R}_i{=}\sum_{k=1}^{K}\lambda_i^k\check{r}_i^k \tag{14}$$

Let T_{iC}^k denote the total time of executing task k of CAV on the CDC associated with RSU i and it is obtained as:

$$T_{iC}^k{=}\dot{T}_{iC}^k{+}\ddot{T}_{iC}^k \tag{15}$$

where $\dot{T}_{iC}^k$ denotes the time of executing task k on CDC associated with RSU i and $\ddot{T}_{iC}^k$ denotes the time of data uploading, downloading for task k between RSU i and the CDC. They are obtained as follows:

$$\dot{T}_{iC}^k{=}\frac{\theta_i\alpha_k\lambda_{ic}^k\beta_k}{f_{ic}^k} \tag{16}$$

$$\ddot{T}_{iC}^k{=}\frac{\theta_i\alpha_k\lambda_{ic}^k}{\hat{R}_{ic}}{+}\frac{\theta_i\alpha_k\lambda_{ic}^k\delta_k}{\check{R}_{ic}} \tag{17}$$

where $\hat{R}_{ic}$ and $\check{R}_{ic}$ denote the total upload/download transfer rate from RSU i to the CDC.

It is worth noting that the computation in CAV and edge (RSU and CDC) works in parallel. As a result, the time by executing task k from the CAV (T_k) is the maximum value of local computation and edge computation, which is obtained as:

$$T_k{=}\textbf{max}(T_L^k, T_i^k{+}T_{iC}^k) \tag{18}$$

Moreover, the total computational time cannot exceed the upper limit required by the user and a RSU must return results to the CAV before it moves beyond the coverage area of this RSU, *i.e.*,

$$T_k \le \hat{T}_k \tag{19}$$

where $\hat{T}_k$ denotes the minimum value of the user's required time limit and the time for the CAV to leave the coverage area of the associated RSU, *i.e.*,

$$\hat{T}_k{=}\textbf{min}(\hat{T}_k^u, \frac{L_i}{v}) \tag{20}$$

where $\hat{T}_k^u$ represents the user's required time limit. L_i denotes the coverage length of RSU i along the route, and v is the speed of the CAV.

C. Energy Model

This section formulates the energy consumption model of the CVEC system. E_L^k denotes the energy consumption of locally executed task k by the CAV and it is obtained as:

$$E_L^k{=}P^L T_L^k{=}s_0(f_0^k)^2\alpha_k\lambda_0^k\beta_k \tag{21}$$

Let E_i^k denote the energy consumption of executing task k associated with RSU i and it comprises two parts, *i.e.*,

$$E_i^k{=}\dot{E}_i^k{+}\ddot{E}_i^k \tag{22}$$

where the first term denotes the energy of executing the offloading task k on RSU i, while the second term denotes the transmission energy between the CAV and RSU i. They are obtained as follows:

$$\dot{E}_i^k{=}P^i\dot{T}_i^k{=}\theta_i s_i(f_i^k)^2\alpha_k\lambda_i^k\beta_k \tag{23}$$

$$\ddot{E}_i^k{=}P^L\frac{\theta_i\alpha_i^k\lambda_i^k}{\hat{R}_i}{+}P^i\frac{\theta_i\alpha_i^k\lambda_i^k}{\check{R}_i} \tag{24}$$

Let E_{iC}^k denote the energy consumption of executing task k related to CDC and it comprises two parts, *i.e.*,

$$E_{iC}^k{=}\dot{E}_{iC}^k{+}\ddot{E}_{iC}^k \tag{25}$$

where the first term denotes the energy of executing the offloading task k on the CDC and the second term denotes the transmission energy between CDC and RSU i. They are obtained as:

$$\dot{E}_{iC}^{k}=P^{C}\dot{T}_{iC}^{k}=P^{C}\frac{\theta_i\alpha_k\lambda_{ic}^{k}\beta_k}{f_{ic}^{k}} \tag{26}$$

$$\ddot{E}_{iC}^{k}=P^{i}\frac{\theta_i\alpha_k\lambda_{ic}^{k}}{\hat{R}_{ic}}+P^{C}\frac{\theta_i\alpha_k\lambda_{ic}^{k}\delta_k}{\check{R}_{ic}} \tag{27}$$

where P^C is the transmitting power of the CDC.

Finally, F denotes the total energy consumption of the CVEC system, which consists of three parts, *i.e.*,

$$F=\sum_{i=1}^{N}\sum_{k=1}^{K}(E_L^k+E_i^k+E_{iC}^k) \tag{28}$$

III. Whale optimization embedded with Simulated-Annealing and Genetic-learning (WSAG)

To sum up, decision variables are λ_0^k, λ_i^k, λ_{ic}^k, θ_i, f_0^k, f_i^k, f_{ic}^k, P^L, P^i, and the optimization objective is to minimize $\mathcal{F}$, *i.e.*,

$$\underset{\Psi}{\textbf{Min}}\ \mathcal{F}$$

where Ψ denotes a vector of decision variables, which are subject to (1), (2), (3), (4) and (19).

Given the constrained optimization problem (F), our proposed WSAG is given as follows. F is nonlinear with respect to decision variables. To handle the above constraints, a penalty function method [9] is adopted to transform all constraints into the penalty and convert this problem into an unconstrained optimization one.

WSAG uses chaotic mapping [10] to initialize the population that can well cover the search space. M denotes the number of individuals of the population and D denotes the dimension of each individual. The population (X) is initialized as:

$$\begin{aligned} Z_i&=4\times Z_{i-1}\times(1-Z_{i-1}), i\in[2,3,...,M]\\ X_i&=\check{b}_d+(\check{b}_d-\hat{b}_d)\times Z_i, i\in[1,2,...,M]\end{aligned} \tag{29}$$

where Z denotes a $M\times D$ zero matrix. $\check{b}_d$ and $\hat{b}_d$ represent lower and upper bounds of each dimension d.

During optimization, Whale Optimization Algorithm (WOA) assumes that each whale has a 50% probability of choosing either the shrinking encircling model or the spiral model. However, the fixed probability of choosing each model may trap into local optima at the later stage of optimization when solving complex optimization problems. As a result, WSAG allows more whales to choose shrinking encircling model at the early stage to enhance the exploration ability and choose the spiral model at the later stage to enhance the exploitation ability. t denotes the current iteration number and $\hat{t}$ denotes the maximum number of iterations. Therefore, when $\frac{t}{\hat{t}}<0.5$, the possibility of choosing shrinking model is 0.8, *i.e.*, $p_1=0.8$; otherwise, $p_1=0.3$.

For the shrinking encircling model, A and C denote the coefficient vectors and they are updated as illustrated in [11]. $|A|<1$ denotes the exploitation phase and each individual is updated as:

$$\begin{aligned} X^i(t+1)&=X^*(t)-A\cdot D\\ D&=|CX^*(t)-X^i(t)|\end{aligned} \tag{30}$$

where $X^i(t)$ and $X^i(t+1)$ is the position of individual i in iteration t and $t+1$. $X^*(t)$ is the best individual of iteration t, and $|\cdot|$ denotes the absolute value.

On the other hand, $|A|\geq 1$ denotes the exploration phase and individuals are updated as:

$$\begin{aligned} X^i(t+1)&=X^r(t)-A\cdot D\\ D&=|C\cdot X^r(t)-X^i(t)|\end{aligned} \tag{31}$$

where $X^r(t)$ is a random individual of the population in iteration t.

Moreover, for the spiral model, individuals are updated as:

$$X^i(t+1)=D^{'}\cdot e^{bl}\cdot cod(2\pi l)+X^*(t) \tag{32}$$

where $D^{'}=|X^*(t)-X(t)|$, b is a constant for defining the shape of the logarithmic spiral, l is a random number in $[-1,1]$.

WSAG adopts SA's Metropolis acceptance rule when selecting individuals for the next iteration. The Metropolis acceptance rule allows for the exploration of directions that may worsen objective function values. This property enables the algorithm to escape local optima and effectively search for the global optimum by adjusting the cooling rate of temperature. The possibility of acceptance is given as:

$$p_a=e^{-\frac{\psi}{T}} \tag{33}$$

where ψ is the difference of fitness values before and after the update. T is the initial temperature in SA, and p_a is the possibility of acceptance.

WSAG incorporates the crossover operation of genetic learning to maintain population diversity. After the update, each individual performs crossover operation. To be specific, if the updated individual is better than a randomly selected individual from the population, $X(t+1)$ is updated as Eq. 34; otherwise, it is updated as Eq. 35.

$$X_d^i(t+1)=r\cdot X_d^*(t)+(1-r_1-r_2)\cdot X_d^k(t) \tag{34}$$

$$X_d^i(t+1)=X_d^k(t) \tag{35}$$

where $X_d^i(t+1)$ denotes dimension d of individual i in iteration $t+1$. $X_d^*(t)$ denotes dimension d of the best individual in iteration t. $X_d^k(t)$ denotes dimension d of individual k in iteration t. r_1 and r_2 are two random numbers between $[-1,1]$. k is a random integer between $[1,M]$.

Finally, a mutation operation is adopted in the later stage of WSAG to enhance the exploration of the search space. To be specific, the individual is mutated and the mutation probability is controlled by p_m. It is worth noting that mutation

is only allowed in the later stage of the optimization process for avoiding to mislead the optimization direction in the early stage. The details of WSAG are provided in Algorithm 1.

Algorithm 1 WSAG

Input: $\hat{t}$, $\check{b}_d$ and $\hat{b}_d$, T, p_m
Output: Final population X

Initialize X with (29)
Calculate the fitness values of all individuals and choose the best one as X^*
for t=1:$\hat{t}$ **do**
  for i=1:M **do**
    Update a, A, C, l with
    if t<0.5 × $\hat{t}$ **then**
      $p_1 = 0.8$
    else
      $p_1 = 0.3$
    end if
    if p<p_1 **then**
      if $|A|$<1 **then**
        Update $X^i(t+1)$ with (30)
      else
        Select a random individual X^r
        Update $X^i(t+1)$ with (31)
      end if
    else
      Update $X^i(t+1)$ with (32)
    end if
    Calculate the acceptance probability with (33)
    if $\mathbf{f}(X^i(t+1)) < \mathbf{f}(X^i(t))$ **then**
      $X^i(t+1) = X^i(t+1)$
    else
      if p>r_1 **then**
        $X^i(t+1) = X^i(t+1)$
      else
        $X^i(t+1) = X^i(t)$
      end if
    end if
    for d=1:D **do**
      Select k as a random number between [1,M]
      if $f(X^i(t+1)) < f(X^i(t))$ **then**
        Update $X^i_d(t+1)$ with (34)
      else
        Update $X^i_d(t+1)$ with (35)
      end if
    end for
    if t>0.8 × $\hat{t}$ **then**
      for d=1:D **do**
        if r<p_m **then**
          X^i_d=$\check{b}_d$+($\hat{b}_d$−$\check{b}_d$)
        end if
      end for
    end if
  end for
  t=t+1
end for
return X

IV. Experimental Results and Discussion

Parameter setting of the CVEC system is shown in Table I. Moreover, the parameters of WSAG are set as follows. p_m=0.4, and T=10^7. We compare WSAG with four benchmark algorithms including genetic algorithm (GA) [12], WOA [13], particle swarm optimization (PSO) [14], and SA [15]. All parameters of these algorithms are set by using their optimal values.

Figs. 2 and 3 illustrate the energy consumption and the penalty comparisons among WSAG, GA, WOA, PSO and SA in each iteration, respectively when there are 5 RSUs in the system. It is shown that WSAG achieves the lowest energy consumption (16.39 J). Moreover, it is shown in Fig. 3 that the penalty of WSAG is zero at the end of the iteration, which proves that the solution obtained by WSAG is valid. The penalties of GA, WOA and PSO cannot decrease to zero and PSO yields a satisfied solution after 860 iterations, which is longer than WSAG. Moreover, we compare WSAG with three offloading strategies including local computing, full offloading and random offloading. The total system energy consumption of the system with different distances of the CAV traveled on the road is shown in Fig. 4. It is shown that WSAG achieves the lowest energy consumption at almost all distances and it increases in a linear manner. Moreover, Table II demonstrates the penalty of the four strategies at different distances and it is shown that none of the strategies except WSAG produces a valid solution, which proves that WSAG outperforms other compared algorithms. Finally, Fig. 5 shows the penalty of WSAG with different number of RSUs. It is shown that the penalty of WSAG is zero, which proves that it can always find valid solutions with different number of RSUs.

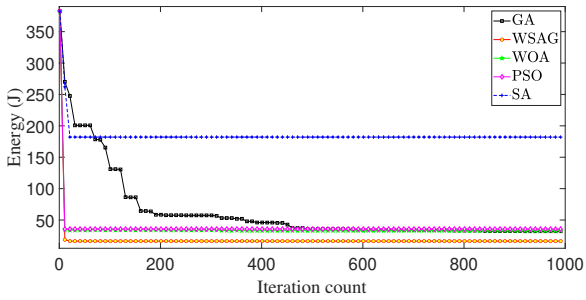

Fig. 2. Energy in each iteration for each algorithm.

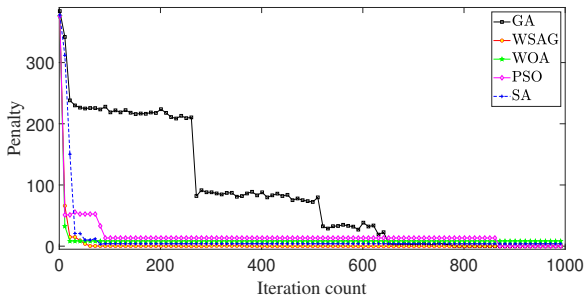

Fig. 3. Penalty for each algorithm.

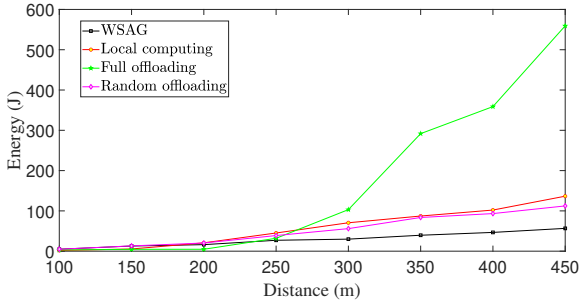

Fig. 4. Energy v.s. distance for each algorithm.

TABLE I
PARAMETER SETTING

δ_k	K	α_k	B_1	B_2	W_i	d_i	S_0	S_i	S_c	$\hat{f}_0$	$\hat{f}_i$	$\hat{f}_c$
0.2	[1,8]	[100 MB-1 GB]	10 MHZ	10 MHZ	1	[50,100] m	10^{-27}	10^{-29}	10^{-31}	10^6 cycles/s	10^8 cycles/s	10^{12} cycles/s

TABLE II
PENALTY OF EACH STRATEGY FOR DIFFERENT DISTANCES

Strategies \ D	100	150	200	250	300	350	400	450
WSAG	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Local computing	1.76×10^{-04}	2.87×10^{-04}	1.84×10^{-02}	6.68×10^{-02}	$7.10\times10^{+01}$	$1.43\times10^{+02}$	$3.81\times10^{+02}$	$1.58\times10^{+04}$
Full offloading	4.12×10^{-04}	4.32×10^{-04}	5.64×10^{-03}	4.67×10^{-01}	$2.90\times10^{+01}$	$3.71\times10^{+02}$	$5.57\times10^{+03}$	$7.41\times10^{+04}$
Random offloading	1.91×10^{-04}	2.29×10^{-04}	2.33×10^{-02}	5.55×10^{-02}	$9.40\times10^{+01}$	$1.17\times10^{+02}$	$1.03\times10^{+02}$	$1.65\times10^{+04}$

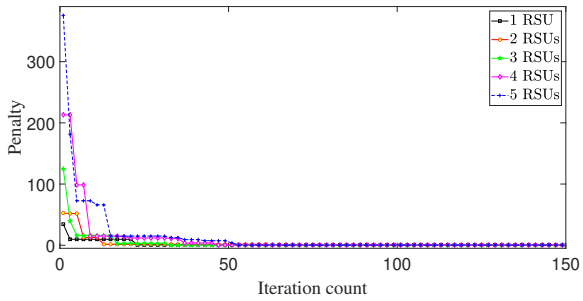

Fig. 5. Penalty of WSAG for different number of RSUs.

V. CONCLUSIONS

Nowadays, there has been a rapid proliferation of Connected and Automated Vehicles (CAVs) distinguished by their advanced capabilities, which have significantly improved people's lives. However, the inherent limitation of computational resources in CAVs poses a challenge when dealing with numerous computationally intensive tasks under strict time constraints. To address this problem, this paper constructs a typical Cloud-assisted Vehicular Edge Computing (CVEC) system. In this CVEC system, CAVs can partial offload their computational tasks to Roadside Units (RSUs) embedded with Roadside Edge Servers (RESs) for processing. Moreover, due to the limited computational resources of RSUs, they can further offload tasks to the Cloud Data Center (CDC) for execution. Additionally, a novel optimization algorithm called Whale optimization embedded with Simulated-Annealing and Genetic-learning (WSAG) is proposed to determine the resource allocation between a CAV, RSUs and the CDC. It also optimizes the energy consumption of the system. Experiments prove that WSAG achieves significantly lower energy consumption with faster convergence speed than state-of-the-art peers.

REFERENCES

[1] M. Shen, C. He, T. Molnar, *et al.*, "Energy-Efficient Connected Cruise Control With Lean Penetration of Connected Vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 4, pp. 4320–4332, Apr. 2023.

[2] S. S. Avedisov, G. Bansal and G. Orosz, "Impacts of Connected Automated Vehicles on Freeway Traffic Patterns at Different Penetration Levels," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 5, pp. 4305–4318, May 2022.

[3] Z. Sharif, L. T. Jung, I. Razzak, *et al.*, "Adaptive and Priority-Based Resource Allocation for Efficient Resources Utilization in Mobile-Edge Computing," *IEEE Internet of Things Journal*, vol. 10, no. 4, pp. 3079–3093, Feb. 2023.

[4] Z. Xue, C. Liu, C. Liao, *et al.*, "Joint Service Caching and Computation Offloading Scheme Based on Deep Reinforcement Learning in Vehicular Edge Computing Systems," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 5, pp. 6709–6722, May 2023.

[5] J. Bi, K. Zhang, H. Yuan and J. Zhang, *et al.*, "Energy-Efficient Computation Offloading for Static and Dynamic Applications in Hybrid Mobile Edge Cloud System," *IEEE Transactions on Sustainable Computing*, vol. 8, no. 2, pp. 232–244, Apr. 2023.

[6] K. Mawatwal, R. Roy and D. Sen, *et al.*, "A State Based Resource Allocation Game for Distributed Optimization in 5G Small-Cell Networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 11, pp. 12072–12087, Nov. 2021.

[7] J. Park and Y. Lim, "Toward Adaptive Energy Management for Mobile Edge Networks," *2022 IEEE 4th Eurasia Conference on IOT, Communication and Engineering*, Yunlin, Taiwan, 2022.

[8] G. Han and S. Shamai, "On Sampling Continuous-Time AWGN Channels," *IEEE Transactions on Information Theory*, vol. 68, no. 2, pp. 782–794, Feb. 2022.

[9] J. Bi, Z. Wang, H. Yuan, *et al.*, "Self-adaptive Teaching-learning-based Optimizer with Improved RBF and Sparse Autoencoder for High-dimensional Problems," *Information Sciences*, vol. 630, pp. 463–481, Jun. 2023.

[10] Xun Yi, "Hash function based on chaotic tent maps," *IEEE Transactions on Circuits and Systems*, vol. 52, no. 6, pp. 354-357, Jun. 2005.

[11] Q. Yang, J. Liu, Z. Wu, *et al.*, "A Fusion Algorithm Based On Whale and Grey Wolf Optimization Algorithm for Solving Real-world Optimization Problems," *Applied Soft Computing*, vol. 146, Oct. 2023.

[12] Q. Zhang, S. Yang, M. Liu, *et al.*, "A New Crossover Mechanism for Genetic Algorithms for Steiner Tree Optimization," *IEEE Transactions on Cybernetics*, vol. 52, no. 5, pp. 3147–3158, May 2022.

[13] Y. Hang and J. Guo, "Research on Scientific Data Mining Algorithms based on WOA-BP Neural Networks," *2023 3rd International Conference on Consumer Electronics and Computer Engineering*, Guangzhou, China, pp. 667–672, 2023.

[14] J. Zhang, Y. Lu, L. Che and M. Zhou, *et al.*, "Moving-Distance-Minimized PSO for Mobile Robot Swarm," *IEEE Transactions on Cybernetics*, vol. 52, no. 9, pp. 9871–9881, Sept. 2022.

[15] M. D. Mahardika and Z. K. A. Baizal, "Recommender System for Tourist Routes in Yogyakarta Using Simulated Annealing Algorithm," *2023 IEEE 8th International Conference for Convergence in Technology*, Lonavla, India, pp. 1-6, Sept. 2023.