Efficient Task Offloading for Mobile Edge Computing in Vehicular Networks

Xiao Han https://orcid.org/0000-0002-4589-5085 Harbin Engineering University, China

Huiqiang Wang Harbin Engineering University, China

Guoliang Yang Harbin Engineering University, China

Chengbo Wang Harbin Engineering University, China

ABSTRACT

In vechcular networks, a promising approach to enhance vehicle task processing capabilities involves using a combination of roadside base stations or vehicles, there are two challenges when integrating the two offloading modeth: 1) the high mobility of vehicles can easily lead to connectivity interruptions between nodes, which in turn affects the processing of the tasks that are being offloaded; and 2) vehicles on the road are not completely trustworthy, and vehicle tasks that contain private information may suffer from result errors or privacy leakage and other problems. This paper investigates the computing offloading problem for minimizing task completion delay in vehicular networks. Specifically, we design a trust model for mobile in-vehicle networks and construct a migration decision problem to minimize the overall delay of task execution for all vehicle users. The simulation results show that the scheme proposed in this paper can effectively reduce the execution delay of the task compared to the baseline scheme.

KEYWORDS

Computing Offloading, Resource Allocation, Trust Model, Vehicular Edge Computing

With the rapid development of new technologies such as autonomous driving and in-vehicle communication, application services that require high computing power or have high latency requirements are widely used in in-vehicle networks, including autonomous driving (Ren et al., 2020; Zhu et al., 2023; Zhou et al., 2019) intelligent traffic control (Xu et al., 2023) and image- or video-assisted real-time navigation (Fan et al., 2023). However, the limited computational resources of vehicles seriously hinder the realization of the above applications, and it is difficult to achieve good performance by relying only on the vehicle itself (Gao et al., 2023); thus, edge computing is regarded as a promising approach (Chen et al., 2023). By utilizing edge computing technology, vehicles can offload the complex computational tasks of applications to platforms with more resources, thus shortening the response time of applications and improving the service quality of applications. When there are multiple resource-rich edge nodes in a network, choosing which edge node to offload the task becomes the primary problem to be solved by the vehicle. Currently, there are two approaches for solving this problem: roadside server-assisted migration and road vehicle migration.

DOI: 10.4018/IJDCF.349133

This article published as an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited. For the first approach, the vast majority of research has focused on server-based migration strategies, and the more common scheme is to achieve migration based on the distance of the requester or node availability. Zhu et al. (2023) considered the constraints of service latency as well as the quality of the task and used the binary particle swarm optimization method to construct a task offloading scheme. Similarly, Hou et al. (2020) introduced partial offloading and redistribution mechanisms into the regular offloading process, using heuristics to maximize reliability under delay constraints. Fan et al. (2023) further refined the task offloading process through collaboration between edges to achieve balanced computational loads in the network. Lin et al. (2020) added an authentication mechanism to ensure security during the migration process.

Due to facility costs, it is impractical to deploy servers along all highways. To address this lack of infrastructure, using other vehicles to achieve task offloading has become an effective approach (Ma et al., 2021). Shi et al. (2020) provided a dynamic pricing scheme that optimizes the gains of vehicles to achieve the migration of computational tasks between vehicles through a deep reinforcement learning approach. Fan et al. (2023) proposed a heterogeneous migration scheme that uses both vehicles and edge servers to achieve migration. Chao et al. (2019), Hou et al. (2020) and Lin et al. (2020) introduced parked vehicles as nodes into the network. Fan et al. (2023) and Xue et al. (2023) utilized parked vehicles as forwarding relays to expand the service range of mobile vehicles as well as edge servers, which minimized the processing delay of weighted tasks within the system. Shi et al. (2020) and Fan et al. (2022) then formed computing clusters of parked vehicles to participate in the task offloading process. Fan et al. (2023) and Dai et al. (2019) utilized only vehicles waiting for traffic lights as temporary computing nodes to assist in migration.

However, problems still exist for in-vehicle network task migration. First, existing task offloading research has mainly focused on offloading between vehicle nodes and roadside base stations or vehicles, and it is rare to consider both offloading modes at the same time. Considering two offloading modes further increases the complexity of the migration decision problem. Second, the above work mostly assumes that the devices in the network are trustworthy, and any node can perform task offloading. However, in an open network environment, malicious nodes are inevitable. These malicious nodes could steal private information or interfere with task offloading. When tasks are offloaded to malicious nodes, the accuracy of task results cannot be guaranteed. Although the traditional authentication system can guarantee the security of task offloading in vehicular networks to a certain extent, it is difficult to filter for malicious nodes with legitimate identities within the network. Constructing trust relationships between nodes in a network is a challenging problem. To address the above issues, we propose a trusted and efficient task offloading scheme for vehicular networks, and our contributions are summarized as follows.

In this paper, we consider a hybrid migration scenario in which computational tasks can be migrated to edge servers as well as road vehicles and design a node trust value evaluation model considering the impacts of vehicle mobility.

In this paper, four parameters – internode transmission delay, network communication link stability, trust assessment, and computation delay – are combined to model the trusted computing offloading problem of service nodes in a vehicle network, and an offloading strategy decision problem with average delay minimization, link stability, and trust assessment maximization is constructed.

We analyze the complexity of the above problem and then design an algorithm based on an alternating direction method of multipliers (ADMM) and a convex difference algorithm to find a set of approximate solutions. Simulation experiments show that the proposed scheme in this paper can effectively reduce task delay and improve service quality.

SYSTEM MODELLING

In this paper, we provide mobile edge computing services to users in a time-varying vehicular network, as shown in the scenario in Figure 1. Mobile vehicles with spare computing resources and



Figure 1. Trusted computing offloading network model scenario

roadside infrastructure together form a service unit to process computing tasks for users. The set of vehicles with task requests is $V_s = \{V_{s1}, V_{s2}, \dots, V_{sn}\}$, and the set of vehicles that can be used as auxiliary computing nodes is $V_e = \{V_{e1}, V_{e2}, \dots, V_{en}\}$.

All nodes can communicate with each other within the communication range for resource sharing and trust evaluation. A node's task request can be migrated to only one node within its communication range. The computational tasks commonly carry real-time road information and authentication information for the user and are highly sensitive to latency. The main parameters used in the model proposed in this paper are shown in Table 1

Assuming that the service range of roadside unit *r* is fixed and that a mobile vehicle is associated with only one service node at moment *t*, each vehicle node in V_s can offload its computational tasks to an auxiliary vehicle node in its area or to roadside unit *r*. Each vehicle node $V_s \in V_s$ transmits mobile data of size k_{v_s} at a time. In this paper, we use the ternary $p_v^t = \{x_v^t, y_v^t, v_v\}$ to denote the operating state of the mobile vehicle at moment *t*, where $\langle x_v^t, y_v^t \rangle$ denotes the position of the mobile vehicle at moment *t* and v_v denotes the speed of the vehicle at moment *t*. $p_r^t = \{x_r^t, y_r^t\}$ indicates the position of the roadside unit at moment *t*. The *n*th computing task received by each service unit is denoted as $M_{sn} = (\omega_{sn} k_{sn}), n = \{1, 2, \dots, V_{sk}\}$, where k_{sn} denotes the packet size of the *n*th task received by the service unit, and ω_{sn} denotes the number of CPU revolutions required by the current service unit to process the *n*th computing task.

Transmission Model

The data forwarding delay between the task generating node V_{si} and the serving node V_{ej} is defined as $D_{V,V}^{C}$, as shown in equation 1.

$$D_{V_{a},V_{d}}^{c} = \frac{k_{V_{a}}}{r_{V_{a},V_{d}}}$$
(1)

Parameters	Define
V_{s}	Pool of vehicles generating tasks
V_{e}	Calculate the vehicle pool
r	roadside unit ^M sn
ω _{sn}	$Task^{M_{sn}}$ CPU RPMs required
k _{sn}	Size of data volume for task M_{sn}
D _{ij}	Communication distance for $link(i,j)$
r _{ij}	Transmission rate for $link^{(i,j)}$
B_{ij}	Communication bandwidth for $link^{(i,j)}$
P _{ij}	Transmission power of the $link(i,j)$
d_{ij}^t	Transmission time for task M_{sn}
R	Maximum communication radius of vehicle nodes
C_{ij}^{t}	Maximum connection time for $link^{(i,j)}$
L_{ij}^t	Stability of the link (i,j)
T_d^t	Direct trust assessment of target nodes
T_p^t	Past trust assessment of target nodes
T^{i}	Global trust assessment of target nodes

In a time-varying vehicular network, the distance between vehicle node V_{si} and vehicle node V_{ej} varies in real time due to the mobility of the vehicle, which affects the transmission rate $r_{V_s V_{ej}}$; the transmission rate should decrease as the node pair V_{si}, V_{ej} distance becomes larger. Thus, transmission rate $r_{V_s V_{ej}}$ can be expressed as shown in equation 2.

$$r_{V_{u},V_{ej}}(t) = B_{V_{u},V_{ej}} \times \log\left(1 + \frac{p_{V_{u},V_{ej}} \times d_{V_{u},V_{ej}}^{-1}(t)}{\sigma^{2}}\right)$$
(2)

where σ^2 is additive Gaussian white noise (AGWN). In the scheme of this paper, it is assumed that the channel adopts the orthogonal frequency division multiple access (OFDMA) communication method, which enables different communication links to occupy orthogonal spectrum resources to avoid co-channel interference. $p_{V_{\alpha}V_{\alpha}}$ represents the channel transmission power. $B_{V_{\alpha}V_{\alpha}}$ represents the channel transmission bandwidth. Therefore, the transmission delay between vehicles should be satisfied as shown in equation 3.

$$\int_{0}^{P_{v_{a}v_{q}}} B_{V_{a}v_{q}} \times \log\left(1 + \frac{p_{V_{a}v_{q}} \times d_{V_{a}v_{q}}^{-1}(t)}{\sigma^{2}}\right) dt = k_{V_{a}}$$
(3)

Assume that V_{si} maintains uniform motion, as shown in Figure 2.

The change in the horizontal and vertical components of the node during the moment t can be calculated as shown in equation 4.

Figure 2. The change law of vehicle user motion



$$\begin{cases} x_{V_{ai}} = x_{V_{ai}}^{t} + \Delta x_{V_{ai}} \times T \\ y_{V_{ai}} = y_{V_{ai}}^{t} + \Delta y_{V_{ai}} \times T \end{cases}$$

$$\tag{4}$$

where x_{V_a}, y_{V_a} denotes the running coordinates of the center node at moment *t* in the current direction of motion after time *T*, Δx_{V_a} denotes the rate of change of the horizontal component of the centre node, and Δy_{V_a} denotes the rate of change of the vertical component of the centre node, as shown in equation 5.

$$\begin{cases} \Delta x_{v_a} = x_{v_a}^t - x_{v_a}^{t-1} \\ \Delta y_{v_a} = y_{v_a}^t - y_{v_a}^{t-1} \end{cases}$$
(5)

Similarly, the change rule of node V_{ej} can be obtained, assuming that V_{ej} maintains uniform motion. The distance between the node pair and $(V_{si}, V_{ej}) d_{V_{a}, V_{ej}}(t)$ can be expressed as shown in equation 6.

$$d_{V_{ui}V_{uj}}(t) = \sqrt{\left(x_{V_{ui}}^{t} + \Delta x_{V_{ui}}t - x_{V_{uj}}^{t} - \Delta x_{V_{uj}}t\right)^{2} + \left(y_{V_{ui}}^{t} + \Delta y_{V_{ui}}t - y_{V_{uj}}^{t} - \Delta y_{V_{uj}}t\right)^{2}}$$
(6)

The data forwarding delay between node V_{si} and curb unit *r* is defined as $D_{V_{si},r}^{C}$ as shown in equation 7.

$$D_{v_{s},r}^{c} = \frac{k_{v_{s}}}{r_{v_{s},r}}$$
(7)

In contrast to vehicles, roadside units are stationary, so the variation in $r_{V_{q,r}}$ is completely determined by the mobility of V_{si} . Therefore, the transmission delay between the additive point and the roadside unit can be calculated as shown in equation 8.

International Journal of Digital Crime and Forensics

$$\int_{0}^{P_{v,r}^{c}} B_{V_{u,v}r} \times \log\left(1 + \frac{p_{V_{u}r} \times d^{-1}(t)}{\sigma^{2}}\right) dt = k_{V_{u}}$$

$$\tag{8}$$

The end-to-end link with the lowest transmission delay is generally selected as the optimal serving node. However, in an in-vehicle network environment, the relative distance between individual nodes varies due to different speeds, and the optimal service node of a vehicle may change between different time gaps. A change in service nodes may lead to the failure of task data transmission, which affects task migration efficiency. Therefore, link stability is also an important influencing factor in the selection process of service nodes. We use the vehicle connection time to measure the stability of the data link, and the maximum connection time of two nodes under the time slot *t* needs to meet the data forwarding delay requirement. The deviation of node connection time under multiple time slots is also compared to measuring the similarity of vehicle node behaviour to further ensure the stability of the data link.

The roadside unit *r* is stationary, and the maximum connection time $D_{V_{x,r}}^t$ of the two nodes at moment *t* can be calculated based on the law of coordinate motion. When node V_{si} intersects with *r* in the communication coverage curve along the current trajectory, it is considered that the two nodes are about to lose their connection, as shown in equation 9.

$$\left(x_{r}^{t} - x_{v_{u}}^{t} - \Delta x_{v_{u}} \times C_{v_{u,v}}^{t}\right)^{2} + \left(y_{r}^{t} - y_{v_{u}}^{t} - \Delta y_{v_{u}} \times C_{v_{u,v}}^{t}\right)^{2} = R^{2}$$
(9)

The maximum connection time $C_{V,V}^t$ of two mobile vehicle nodes at moment *t* can be calculated via the same approach. As shown in Figure 3, two nodes are considered about to lose connections when node V_{ej} intersects the communication coverage curve along its current trajectory, as shown in equation 10.

$$\begin{cases} \text{Skipping malformed tag: msup + Skipping malformed tag: msup } = R^2 \\ x_{V_{eq}} = x_{V_{eq}}^t + \Delta x_{V_{eq}} \times C_{V_{a}V_{eq}}^t \\ y_{V_{eq}} = y_{V_{eq}}^t + \Delta y_{V_{eq}} \times C_{V_{a}V_{eq}}^t \end{cases}$$
(10)

At this point, the end-to-end maximum connection time can be obtained at the moment *t*; when the maximum connection time meets the data forwarding delay, the current data link is considered stable, as shown in equation 11 and 12.

$$D^{C}_{V_{a}V_{ij}} \leq C^{t}_{V_{a}V_{ij}} \tag{11}$$

$$D_{V_{u}r}^{C} \le C_{V_{u}r}^{t} \tag{12}$$

Considering the deviation of the connection times of the node pairs V_{si} , V_{ej} and V_{si} , r at past moments, which is defined as the stability of links L_{V_s,V_j}^t and $L_{V_s,v}^t$, the larger the deviation of the connection time, the greater the risk of packet loss. The stability of the link at moment t is expressed as the sum of the connection time deviations of the previous moments, as shown in equations 13 and 14.

$$L_{V_{x},V_{ij}}^{t} = \Delta C_{V_{x},V_{s}}^{t} + \gamma \Delta C_{V_{x},V_{s}}^{t-1} + \dots + \gamma^{n-1} C_{V_{x},V_{s}}^{t-n+1}$$
(13)



Figure 3. Trends in point-to-point connectivity for vehicle users

$$L_{V_{,,r}}^{t} = \Delta C_{V_{,,r}}^{t} + \gamma \Delta C_{V_{,,r}}^{t-1} + \dots + \gamma^{n-1} C_{V_{,,r}}^{t-n+1}$$
(14)

where $\Delta C_{V_{a_i}V_{a_j}} = |C_{V_{a_i}V_{a_j}}^t - C_{V_{a_i}V_{a_j}}^{t-1}|$ is the connection time change of two consecutive moments, and the connection time change of the most recent moment is more informative than that of relatively old connections. Therefore, this paper introduces the decay factor $\gamma \in [0, 1]$ to reflect the effect of time.

Trust Models

The evaluation T_d^t obtained by node V_{si} and node V_{ej} through direct interaction within time interval *t* can be expressed as shown in equation 15.

$$T_{d(i,j)}^{(t)} = \alpha T_{c(i,j)}^{(t)} + (1 - \alpha) T_{p(i,j)}^{(t)}$$
(15)

In equation 15, T_p^r is a measure of the historical behaviour of a node V_{ej} . Past trust changes over time, ensuring that its impact is reduced before new trust values are aggregated. Therefore, in this paper, we assume that the trust value decays over time with a decay factor of μ . Past trust decays over time regardless of the presence of new interactions within the time interval *t*, as shown in equation 16.

$$T_{p(i,j)}^{(t)} = \mu T_{d(i,j)}^{(t-1)}$$
(16)

Current trust is the trust value calculated by the direct interaction between node V_{si} and node V_{ej} at the current moment and is only calculated if node V_{si} has direct communication with node V_{ej} at moment t. In this paper, $T_{c(i,j)}^{(t)}$ is calculated by collecting the number of positive and negative interactions between node V_{si} and node V_{ej} , as shown in equation 17.

$$T_{c(i,j)}^{(t)} = \frac{P}{P+R}$$
 (17)

where P represents the number of forward interactions between node V_{si} and node V_{ej} , and R represents the number of reverse interactions between V_{si} and node V_{ej} .

Indirect trust represents the trust value of node V_{ej} calculated by node V_{si} using the recommendations of all neighbor nodes V_{ek} within its communication range. The recommendation given by each neighbor node depends on the direct observation of the target node V_{ej} by that neighbor node V_{ek} . Indirect trust predicts the trust value of the target node and helps to overcome the cold start problem that exists when there is no direct trust between the source node V_{si} and the target node V_{ej} . Source node V_{si} broadcasts an indirect trust request to all neighbor nodes V_{ek} within the communication range and sends it as a recommendation message to source node V_{si} when V_{ek} possesses the recommended trust $T_{d(k,j)}^{(i)}$. When evaluating nodes using indirect trust values, malicious nodes may send recommendation messages.

We use confidence $Conf_k$ with trust bias Dev_k to filter neighbor nodes that behave abnormally or make false recommendations in the neighbor recommendation list, and when $Conf_k - Dev_k < 0$, the current neighbor node is considered a malicious neighbor node. The confidence $Conf_k$ measures whether source node V_{si} can trust recommended neighbor nodes V_{ek} , and node V_{si} judges the recommendation of each neighbor node V_{ek} based on the current trust value $T_{c(i,k)}^{(i)}$. False recommendations from malicious nodes $T_{r(k,j)}^{(i)}$ are ignored.

The trust bias $De v_k$ is used to determine whether there is any improper false recommendation by further judging the compatibility of the recommendation of each node in the list with the subjective judgement value of the source node V_{si} after filtering out the nodes with abnormal behaviour in the list. Node V_{si} compares the received recommendations with the current trust value $T_{c(i,j)}^{(i)}$ and accepts only those recommendations that have a smaller deviation from $T_{c(i,j)}^{(i)}$.

When a node lacks historical interaction information with the target node, the subjective judgement value cannot be used as the base value for deviation comparison. To overcome this problem, the confidence value of the target node is temporarily used as the base value for comparison with that of the current neighbor nodes. In contrast, when the source node has sufficient subjective judgement on the target node, it can compare the deviation between the neighbor's recommended value and its own subjective judgement value, as shown in equation 18.

$$De v_{k} = \begin{cases} \left| T_{c(i,j)}^{(i)} - T_{c(k,j)}^{(i)} \right| if T_{p(i,j)}^{(i)} \ge 0.5 \\ \left| Conf_{k} - Conf_{j} \right| otherwise \end{cases}$$
(18)

By calculating $Conf_k$ and Dev_k , it is possible to exclude nodes that make false recommendations. The behaviour between nodes is similar to that of social networks, and we use the closeness $Int_{(k,j)}$ and similarity $Sim_{(i,j)}$ in social relationships to further measure the relationships between nodes.

The historical interaction degree between neighbor node V_{ek} and target node V_{ej} is measured by $Int_{(k,j)}$. Under the condition of good behaviour of neighbor nodes, when the neighbor node has a long time content interaction with the target node, it can be considered that neighbor node V_{ek} is more sufficiently aware of the current target node V_{ej} , and the value of its recommendation is high.

Figure 4. Travelling pinching angle for vehicle users at time gap



 $Sim_{(i,j)}$ represents the behavioural similarity between source node V_{si} and neighbor node V_{ek} . A node always wants to find a possible long-term neighbor within a dynamic communication range. In this way, trusted long-term neighbors can continuously feed the central vehicle with a high value of benign recommendations.

As shown in Figure 4, the position information of the two nodes is extracted from the beacon messages of the source node and the neighbor nodes under the two time slots. Based on the coordinates of the source node under moment $t - 1(x_i^{(t-1)}, y_i^{(t-1)})$ and the coordinates of the source node under moment $t(x_i^{(t)}, y_i^{(t)})$, the vector \vec{L}_i of V_{si} under time slot t is constructed. Based on the coordinates of the neighbor node at time $t - 1(x_k^{(t-1)}, y_k^{(t-1)})$ and the coordinates of the neighbor node t at time $(x_k^{(t)}, y_k^{(t)})$, we construct the vector \vec{L}_k of V_{sk} under the time slott.

Based on the two vectors, the travelling angle within the current time slot of the two nodes can be determined, as shown in equation 19.

$$\cos\theta = \frac{\vec{L}_i \times \vec{L}_k}{\left|\vec{L}_i\right| \times \left|\vec{L}_k\right|} \tag{19}$$

The clip angle can be used to simply determine whether the neighbor can continue to be a trusted neighbor to provide recommendations by the next task migration, and if the current neighbor node meets the judgement conditions, it is said to be an old neighbor within the communication range. Finding additional neighbor nodes and assigning higher weights to the recommendations they provide can improve the trustworthiness of the whole indirect trust model. If the current neighbor node does not meet the judgement conditions, then such a short-term passer-by within the communication range is given a lower similarity weight.

When the angle of vehicle travel is from 0 to 45, it is considered that the neighboring vehicle and the current center vehicle are still driving on the same road, and the deviation of the vehicle's running direction indicates that the vehicle may change lanes or overtake, and such a deviation of the driving behavior is considered temporary, which means that the neighboring vehicle has a high degree of behavioral similarity with the current center vehicle. When the angle of vehicle travel is from 45 to 90, the neighboring vehicle is considered at a fork in the road with the current center vehicle, and the deviation of the vehicle's running direction indicates that the vehicle may change the road to the left or to the right; such a deviation of driving behaviour accelerates the reduction of the remaining connecting time between the vehicles, and this time, the neighboring vehicle has a proper behavioral similarity with the current center vehicle. When the vehicle driving angle is between 90 and 180, the neighboring vehicle is considered changing the road in the opposite direction or driving on the opposite road from the current center vehicle at the beginning of the connection, and such a driving behavior deviation results in the connection between the vehicles being only temporary, and the neighboring vehicle has a lower behavioral similarity to the current center vehicle *Sim_(i,j)*. This can be expressed as shown in equation 20.

$$\begin{split} Sim_{(ij)} &= \frac{\cos\theta + 1}{2} \\ \begin{cases} 0^{\circ} \leq \theta < 45^{\circ} & Sim_{(ij)} \in [1, \frac{1}{2} + \frac{\sqrt{2}}{4}) \\ 45^{\circ} \leq \theta < 90^{\circ} & Sim_{(ij)} \in [\frac{1}{2} + \frac{\sqrt{2}}{4}, \frac{1}{2}) \\ 90^{\circ} \leq \theta \leq 180^{\circ} & Sim_{(ij)} \in [\frac{1}{2}, 0] \end{split}$$
(20)

where β represents the assigned weights of the current trust and indirect trust. At moment *t*, the global trust value $T_{(i,i)}^{(i)}$ computed by node V_{si} on node V_{ei} can be computed as shown in equation 21.

$$T_{(i,j)}^{(i)} = \frac{1}{n} \sum_{k \in \mathbb{R}} (1 - \beta) \times T_{d(i,j)}^{(i)} + \beta \times T_{d(k,j)}^{(i)}$$

$$\beta = \begin{cases} 0.5 & if \frac{Int_{(k,j)} + Sim_{(i,k)}}{2} \ge 0.5 \\ \frac{Int_{(k,j)} + Sim_{(i,k)}}{2} & otherwise \end{cases}$$
(21)

Service Models

Assuming that the computational powers of r and the service vehicle V_{ej} are f_r and $f_{V_{ej}}$, respectively, the computational latency required to process task ω_{si} is denoted as shown in equations 22 and 23.

$$D^{M}_{V_{si}r} = \frac{\omega_{si}}{f_{r}}$$
(22)

$$D^M_{V_a,V_{aj}} = \frac{\omega_{si}}{f_{V_{aj}}}$$
(23)

Thus, the task generates the total processing delay required for vehicle V_{si} unloading to the roadside unit versus the service vehicle, as shown in equations 24 and 25.

$$D_{V_{a},r} = D_{V_{a},r}^{C} + D_{V_{a},r}^{M}$$
(24)

$$D_{V_{a},V_{a}} = D_{V_{a}V_{a}}^{C} + D_{V_{a},V_{a}}^{M}$$
(25)

To avoid the inefficiency of service processing caused by many real-time tasks, this paper designs service processing windows C_r and C_j for the roadside units and service vehicles, respectively, and the service nodes refuse to perform the newly arrived computational tasks when the service window is congested.

The goal of this paper is to select the global best service node in the scenario for each taskgenerating vehicle to minimize the task processing delay and maximize the communication link stability with the trust metric of the target service node. Therefore, in this paper, we denote γ as the sum of the weights of the above parameters, and the joint optimization problem of task offloading can be formulated as shown in equation 26.

$$P1: \min_{x_{ij} \forall j_{ir}} \sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij} \gamma_{ij} + y_{ir} \gamma_{ir}$$
s.t. $C_{1}: D_{V_{xi} V_{ij}} \leq C'_{v_{xi}}$
 $C_{2}: D_{V_{xi} r} \leq C'_{v_{xi}}$
 $C_{3}: L'_{v_{xi} V_{ij}}, L'_{V_{xi} r} \leq \Gamma_{L}$
 $C_{4}: 0 \leq p_{j}, p_{r} \leq p_{\max}$
 $C_{5}: \sum_{j=1}^{M} x_{ij} + y_{ir} = 1, i \in N, j \in M$
 $C_{6}: x_{ij}, y_{ir} \in \{0, 1\}, i \in N, j \in M$
 $C_{7}: \sum_{i=1}^{N} \omega_{si} x_{ij} \leq C_{j}$
 $C_{8}: \sum_{i=1}^{N} \omega_{si} y_{ir} \leq C_{r}$

$$(26)$$

where C_1 and C_2 indicate that the task processing delay must not exceed the maximum connection time of the end-to-end link. C_3 indicates that the stability of the end-to-end link cannot be lower than the minimum stability requirement, C_4 indicates that the transmit power of each service vehicle and roadside unit cannot exceed the power maximum limit, and C_5 and C_6 indicate that each task-generating vehicle can only select a unique service node in the scenario for task offloading. C_7 and C_8 indicate that the service node cannot perform more computational tasks than the maximum capacity of the service window.

PROBLEM ANALYSIS AND ALGORITHM DESIGN

Since problem P1 contains binary decision variables x_{ij} and y_{ir} , the presence of discrete variables causes P1 to be a nonconvex problem, making the solution of this problem more complex. At the same time, all binary decision variables are coupled. Referring to the objective function form of Du et al. (2023) we can transform the P1 problem into a SAT problem to prove that the P1 problem is an NP question. At this point, it is almost impossible to find the optimal solution to the P1 problem in polynomial time.

Parallel ADMM Optimization Framework

For problem P1, the number of variables and constraints reaches MN + N and 2MN + 6N + 3M, respectively, and the computational burden of this problem increases as the number of vehicles and

service units generated by the task increases. The ADMM is a heuristic algorithm for solving large-scale optimization problems (Boyd et al., 2021). In this paper, the original problem is divided independently through the ADMM algorithm, and the convex difference algorithm is used to solve the independent subproblems. Specifically, this paper decomposes P1 into M + 1 subproblems, and the problems are independent of each other. Each subproblem corresponds to a service vehicle or a roadside unit. This paper utilizes the same centralized controller to parallelize these subproblems, and each subproblem is run on a different computational unit to ensure solution efficiency. Thus, the solving efficiency is guaranteed.

First, P1 is deconstructed into several independent subproblems, but due to the existence of the coupling constraint C_5 , the original problem cannot be divided simply. Therefore, in this paper, we consider copying the global variable x_{ij}, y_{ir} to obtain its corresponding local variable $\overline{x_{ij}}, \overline{y_{ir}}$, and at the same time, according to the constraint C_5 , we can obtain the coupling constraints of the local variables, as shown in equation 27.

$$\begin{cases} \overline{x_{ij}} = x_{ij} \ i \in N, j \in M \\ \overline{y_{ir}} = y_{ir} \ i \in N \end{cases}$$

$$(27)$$

Based on equation 27, the coupling constraint C_5 of problem P1 can be converted to equation 28.

$$\sum_{j=1}^{M} \overline{x_{ij}} + \overline{y_{ir}}, i \in N, j \in M$$
(28)

Using equations 27 and 28 to replace C_5 , the problem P1 can be equivalently converted to equation 29.

P2:
$$\min_{V_{u} \in V_{x}} \sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij} \gamma_{ij} + y_{ir} \gamma_{ir}$$

s.t. $C_{1}, C_{2}, C_{3}, C_{4}, C_{5}, C_{6}, C_{7}, C_{8}, (27), (28)$ (29)

Thus, the augmented Lagrangian function of problem P2 can be expressed as shown in equation 30.

$$L_{p}(x_{ij},\overline{x_{ij}},y_{ir},\overline{y_{ir}},\lambda,\mu) = \sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij}\gamma_{ij} + y_{ir}\gamma_{ir} + \sum_{i=1}^{N} \sum_{j=1}^{M} \lambda_{ij}(x_{ij} - \overline{x_{ij}}) + \sum_{i=1}^{N} \mu_{i}(y_{ir} - \overline{y_{ir}}) + \frac{\rho}{2} \sum_{i=1}^{N} \sum_{j=1}^{M} \text{Skipping malformed tag: msup} + \frac{\rho}{2} \sum_{i=1}^{N} \text{Skipping malformed tag: msup}$$
(30)

where $\lambda = \{\lambda_{ij}\}_{i \in N, j \in M}$, $\mu = \{\mu_i\}_{i \in N}$ are the Lagrange multipliers on the coupling constraints of the global and local variables in equation 28, respectively. $\rho > 0$ is the coefficient of the quadratic penalty term, and this coefficient determines the convergence speed of the ADMNM. In this paper, the problem P1 is solved step by step by treating the Lagrangian function by augmenting and generalizing the global variables, the local variables, and the update of the dyadic variables.

For global variable updating, at the t + 1 iteration, the optimization framework can obtain the updated values of local and dyadic variables after the *t* iteration, so the global variable $\{x_{ij}, y_{ir}\}$ will be further updated at the t + 1 iteration by solving the problem P3, as shown in equation 31.

P3:
$$\min_{v_{s} \in V_{s}} L_{p}(x_{ij}, \overline{x_{ij}(t)}, y_{ir}, \overline{y_{ir}(t)}, \lambda(t), \mu(t))$$

s.t. $C_{1}, C_{2}, C_{3}, C_{4}, C_{6}, C_{7}, C_{8}$ (31)

For each service vehicle V_{ej} , let $x_j = \{x_{ij}, i \in N\}$ denote the vector of all local offloading decisions associated with the service vehicle V_{ej} , and for the curb unit r, let $y = \{y_i, i \in N\}$ denote the vector of all local offloading decisions associated with the curb unit r. Thus, problem P3 can be rewritten as shown in equation 32.

$$L_p(x_{ij},\overline{x_{ij}(t)},y_{ir},\overline{y_{ir}(t)},\lambda(t),\mu(t)) = \sum_{j=1}^M f_j(x_j) + g_r(y)$$
(32)

where the functions $f_i(x_i)$, $g_r(y)$ are denoted as shown in equations 33 and 34.

$$f_{j}(x_{j}) = \sum_{i=1}^{N} x_{ij} \gamma_{ij} + \lambda_{ij}(t) (x_{ij} - \overline{x_{ij}(t)}) + \frac{\rho}{2} (x_{ij} - \overline{x_{ij}(t)})^{2}$$
(33)

$$g_{r}(y) = \sum_{i=1}^{N} y_{ir} \gamma_{ir} + \mu_{i}(t) (y_{ir} - \overline{y_{ir}(t)}) + \frac{\rho}{2} (y_{ir} - \overline{y_{ir}(t)})^{2}$$
(34)

Therefore, the objective function of problem P3 is fully separable with the division of the set of decision vectors of service vehicles and roadside units. Moreover, the objective function can be further deconstructed into two independent functions $f_j(x_j)$, $g_r(y)$. In this paper, problem P3 is further deconstructed into two subproblems, P4 and P5, where each computational unit $j,j \in [1, M]$ can independently handle problem P4, computational unit M + 1 will handle problems P5, P4 and P5 associated with roadside unit r, and the computational unit will handle problems and be associated with the roadside unit, as shown in equations 35 and 36.

P4:
$$min_{x_{j}}f_{j}(x_{j})$$

s.t. $C_{1}, C_{3}, C_{4}, C_{7}, x_{ij} \in \{0, 1\}, i \in N$ (35)

$$P5:min_{y}g_{r}(y)$$
s.t. $C_{2}, C_{3}, C_{4}, C_{8}, y_{i} \in \{0, 1\}, i \in N$
(36)

The subproblem P4 is nonconvex due to the presence of the binary variable $x_{ij} \in \{0, 1\}$. We equivalently convert the binary variable constraints in P4 to the interaction of equation 37 and further convert P4 to P6, as shown in equations 37 and 38.

$$\sum_{i \in N} (x_{ij} - x_{ij}^2) \le 0$$
(37)

 $s.t. \ x_{ii} \in [0,1], i \in N$

P6:
$$\min_{x_j} f_j(x_j)$$

s.t. $C_1, C_3, C_4, C_7, (37)$ (38)

Table 2. Subproblem solving based on convex difference algorithm

Algorithm 1: Convex Difference Algorithm for Solving the Subproblem p^7 **Inputs:** maximum number of iterations n_{\max} , convergence value δ **Output:** convergence result for x_j **Initialization:** Select an initial feasible solution x_j^0 to the subproblem p^7 . n = 01. for $n \le n_{\max}$ do 2. Solve the convex problem P8: $\min_{x_j} a_j(x_j^n) - b_j(x_j^n) - \nabla x_j b_j(x_j^n)(x_j - x_j^n)$ 3. Obtain the optimal solution x_j^{n+1} 4. n = n + 15. end for

After the above equivalent conversion, the objective function of problem P6 is linear, but the existence of equation 37 causes problem P6 to remain nonconvex. To address equation 37, this paper converts problem P6 to problem P7, as shown in equation 39.

P7:
$$min_{j}f_{j}(x_{j}) + \zeta \sum_{i \in \mathbb{N}} (x_{ij} - x_{ij}^{2})$$

s.t. $C_{1}, C_{3}, C_{4}, C_{7}, (37)$ (39)

where ζ is a penalty factor, $\zeta > 0$. Problem P7 is equivalent to problem P6 when there exists a sufficiently large ζ (Che et al., 2014).

We use a convex difference algorithm to solve the nonconvex problem P7, which splits the problem P7, into the difference of two standard convex functions, i.e., $a_j(x_j) - b_j(x_j)$, where $a_j(x_j) = f_j(x_j) + \zeta \sum_{i \in \mathbb{N}} x_{ij}$ and $b_j(x_j) = \zeta \sum_{i \in \mathbb{N}} x_{ij}^2$. According to the algorithm in Table 2, in this paper, we use a sequential convex approximation method to obtain a locally optimal solution in the current iteration round.

 x_j^n is the solution obtained by the algorithm at the nth iteration, and ∇x_j is a subgradient operation on the feasible set of service vehicle decision variables x_j that yields the problem P8 as a convex function. In this paper, we use the interior point method (Wright, 2023) to solve P8 and thus obtain the near-optimal solution x_j of P7. In Algorithm 1, x_j^n obtained in each iteration is always better than x_j^{n-1} obtained in the previous iteration. The iterative solution set $\{x_j^n\}$ converges. Thus, the iterative process can converge in a finite number of rounds, and the convergence results are consistent.

Consistent with P4, P5 is also linear and has the binary variable $y_i \in \{0, 1\}$. Therefore, P5 can be solved using the same solution method as its counterpart, the serial-convex approximation.

For the update of local variables, at iteration t + 1, the optimization framework can obtain the updated values of the global variables after iteration t + 1 and the updated values of the dyadic variables after iteration t, so the local variable $\{\overline{x_{ij}}, \overline{y_{ir}}\}$ will be further updated at iteration t + 1 by solving problem P9, as shown in equation 40.

$$P9:\min_{V_{i} \in V_{i}} L_{p}(x_{ij}(t+1), \overline{x_{ij}}, y_{ir}(t+1), \overline{y_{ir}}, \lambda(t), \mu(t))$$

$$s.t. \quad (28), \overline{x_{ij}}, \overline{y_{ir}} \in \{0, 1\}, i \in N, j \in M$$

$$(40)$$

After eliminating the constant term in the objective function P9, the objective function is equivalent to equation 41.

$$\min_{\overline{x_{ij}, \overline{y_{ir}}}} \sum_{i=1}^{N} L_i(\overline{x_{ij}}, \overline{y_{ir}})$$

$$s.t. \quad (28), \overline{x_{ij}}, \overline{y_{ir}} \in \{0, 1\}, i \in N, j \in M$$

$$(41)$$

where $L_i(\overline{x}, \overline{y_i})$ is the local variable update function, which is expressed as shown in equation 42.

$$L_{i}(\overline{x_{ij}}, \overline{y_{ii}}) = \sum_{i}^{M} \left[\frac{\rho}{2} \overline{x_{ij}^{2}} - \lambda_{ij}(t) \overline{x_{ij}} - \rho \overline{x}_{i_{ij}} x_{ij}(t+1) \right] + \left[\frac{\rho}{2} \overline{y_{ii}^{2}} - \mu(t) \overline{y_{ii}} - \rho \overline{y_{i}} y_{ij}(t+1) \right]$$
(42)

According to the coupling constraints of local variables, for each requesting vehicle *i*, only one of the local variables $(\overline{x_{ij}}, \overline{y_{ir}})$ can be assigned a value of 1. Therefore, for each requesting vehicle *i*, there are M + 1 possible solutions for the service node selection decision. By calculating the values of the local variable update function corresponding to the M + 1 decisions, the minimum is selected as the solution of the local variable. Thus, the optimal solution of problem P9 is formalized as shown in equation 43.

$$\begin{cases} \overline{x_{ij}} = 1, \overline{y_{ir}} = 0 \ if \ c_{i,j} = C_{i,\min}, i \in N, j \in M \\ \overline{x_{ij}} = 0, \overline{y_{ir}} = 1 \ if \ c_{i,r} = C_{i,\min}, i \in N, j \in M \end{cases}$$
(43)

 $c_{i,j} = 1/2 \ \rho - \lambda_{ij}(t) - \rho x_{ij}(t+1) \text{ and } c_{i,r} = 1/2 \ \rho - \mu_i(t) - \rho y_{ir}(t+1). \ c_{i,min} \text{ denotes the optimal value among all feasible solutions for the requested vehicle } i, which is denoted as <math>c_{i,min} = \min\{c_{i,j \in M}, c_{i,r}\}$.

To update the dyadic variables, this paper utilizes $\{x_{ij}(t+1), y_{ir}(t+1), \overline{y_{ij}(t+1)}, \overline{y_{ir}(t+1)}\}$ obtained from the above optimization of the global and local variables and further updates the dyadic variables $\lambda_{ij}(t+1)$ and $\mu_{ij}(t+1)$ in the first t+1 iteration. As shown in Table 3, Algorithm 2 completes the solution of the original problem P1 through a parallel optimization framework shown in equations 44 and 45.

$$\lambda_{ij}(t+1) = \lambda_{ij}(t) + \rho \left(x_{ij}(t+1) - \overline{x_{ij}(t+1)} \right), i \in N, j \in M$$
(44)

$$\mu_i(t+1) = \mu_i(t) + \rho(y_{ir}(t+1) - \overline{y_{ir}(t+1)}), i \in N$$
(45)

Performance Analysis of the Proposed Algorithm

In this paper, problem P3 is deconstructed into two subproblems, P4 and P5, by updating the global variables. The two subproblems use the convex difference algorithm to solve the problem with the presence of binary variable constraints. First, in subproblem P4, there are N optimization variables and 2N + 2 convex constraints for each service vehicle node V_{e_i} . Therefore, the computational complexity of solving subproblem P4 is $O(\max_{j \in M} |N|^3) (2|N| + 2)) = O(\max_{j \in M} |N|^4)$. In subproblem P5, the common curb unit r has N optimization variables and 2N + 2 convex constraints, so the computational complexity of solving subproblem P5 is also $O(\max_r |N|^4)$. Second, in the local variable updating problem P9, let c_1 denote the computational steps required to solve equation 43, so the computational complexity of solving P9 is c_1N . Finally, in the update problem for the dyadic variables, let c_2 denote the computational steps required to solve equational complexity is finally measured as c_2N . Thus, the computational complexity of each iteration of the ADMM optimization framework is $O(N^4 + (c_1 + c_2)N) = O(N^4)$.

Let *t* denote the maximum number of iterations required for the algorithm to converge; thus, the overall computational complexity of the proposed algorithm is obtained as $O(N^4)t$. The proposed algorithm significantly improves the solution efficiency compared to the branch-and-bound method for solving the problem at the exponential scale.

Table 3. ADMM algorithm for trusted computation offloading of service nodes

Algorithm 2: ADMM-based service node selection algorithm
Inputs: maximum number of iterations t_{max} , convergence error <i>emp</i>
Output: Task completion time
Initialization: $t = 1, \varepsilon = 0.01, \rho = 0.05, \lambda_{ij}^{1} = 0, i \in N, j \in M, \mu_{i}^{1} = 0, i \in N$
1. While $ \{x,y\} - \{\bar{x},\bar{y}\} \ge \varepsilon$ and $t \le t_{\max}$
2. Update global variables: M computation unit parallel computation subproblem p^7 , $M + 1$ computation unit
computation subproblem p^5 , update global variables $\{x, y\}^{r+1}$
3. Update local variables: update local variables by equation 43
4. Update the dyadic variables: update the dyadic variables by equations 44 and 45
5. t = t + 1
6. Calculate the value of the generalized Lagrangian function by bringing $\{x, y\}$ into equation 31
7. End while

Parameters	Define	(be) Worth
р	Calculate task packet size	1M
B _{i,r}	Roadside unit channel bandwidth	20MHz
B _{i,j}	Vehicle node channel bandwidth	15MHz
Р	Channel transmission power	200mW
R	Vehicle node communication radius	1000m
ω_i	Number of CPU cycles required for task computation	[3,10]Gigacycles
f_r	Roadside unit computing power	5 Gigacycles/s
C _r	Calculated Load for Curbside Units	30 Gigacycles
f_{f}	Helper Vehicle Computing Capability	1 Gigacycles/s
C_j	Helper Vehicle Calculation Load	20 Gigacycles

Table 4. Simulation parameters

Experimental Analysis

In this section, the paper first evaluates the convergence of the proposed algorithm and then verifies the performance of the proposed scheme against other baseline schemes. The main dataset used in this paper is derived from a publicly available dataset (Wang et al., 2019). The required experimental parameter settings for this section are shown in Table 4.

Algorithm Performance Analysis

Figure 5 demonstrates the convergence process of the proposed scheme in this paper for N=5, N=10, N=20, and N=40, respectively, and compares the performance of the proposed algorithm with that of the branch-and-bound method. The branch-and-bound (BNB) method is a classic method for solving discrete combinatorial optimization problems that is able to achieve the optimal performance of $1 - \varepsilon$ for a given small parameter $\varepsilon \ge 0$. In this experiment, $\varepsilon = 0.001$ is taken to obtain the approximate optimal solutions obtained by the branch-and-bound method in four cases. With increasing system size, the number of iterations required by the proposed algorithm does not increase explosively, the total number of iterations in the four cases is relatively stable, and all of them can converge quickly within 35 iterations. At the same time, the final results under the four system sizes are very close to those of the branch-and-bound method, and it can be seen that the proposed algorithm can obtain an approximate optimal solution.



Figure 5. Convergence process of the proposed algorithm

Figure 6 compares the running time of the proposed algorithm with that of the branching delimitation method for different numbers of requested vehicles. Due to the time-consuming nature of the branching delimitation method, its running time grows approximately exponentially with the increase in the number of requested vehicles. In contrast, the average running time of the proposed algorithm increases slowly with the increase in the number of requested vehicles, which shows that the proposed algorithm can significantly reduce the running time while guaranteeing good performance.

Comparative Analysis

To validate the performance of the proposed scheme in system task computations, two additional baseline schemes are introduced in this paper: All-Local and All-Edge. In the All-Local scheme, all tasks generated by the requesting vehicles are handed over to the local terminal for computation. In the All-Edge scheme, all tasks are fully offloaded to be computed at the roadside unit.

Figure 7 reflects the overall task duration required with different schemes for different average computation volumes of requesting vehicles.

As the average computation amount of the requesting vehicles increases, the computation latency required by the service nodes and the local terminals increases with the fixed computation capacity of the nodes, and the overall task lengths of the three strategies increase significantly. The overall task duration required by the algorithm proposed in this paper is lower than that of the All-Edge scheme. This is because, in this paper, the two metrics of D2D link stability and security, which

Figure 6. Comparison of running times with different numbers of requested vehicles



Figure 7. Comparison of overall task duration for different average computations of requested vehicles



affect the quality of task service, are introduced in the selection of optimal service nodes. Although there is a significant difference in computational power between helper vehicles and roadside units, the computational latency required by both roadside units and helper vehicles is within the latency tolerance of the requesting vehicle when the given computational volume is small. The nodes will favor the two parameters of D2D link stability and security when selecting the optimal service node. Under the comprehensive consideration of delay, link stability, and trust assessment, the overall task duration obtained by the algorithm proposed in this paper is significantly smaller than that of the All-Edge scheme, shortening the task duration by approximately 13%. As the average computation of the requesting vehicle increases, the roadside unit cannot satisfy the full computation from the vehicle. At this point, the requesting vehicle hands over the tasks that cannot be offloaded to local computation, resulting in a rapid increase in the overall task duration of the All-Edge scheme. When

the average computation volume of the requesting vehicles is set to 10, the scheme proposed in this paper shortens the task duration by approximately 48% compared to the All-Edge scheme. Compared with the other two baseline schemes, the scheme proposed in this paper can significantly reduce the system task duration while ensuring service stability and security.

As shown in Figure 8, the experiment examines the computational resource utilization of the proposed scheme in this paper at the roadside unit (RSU) and the helper vehicle for different average computations of the requesting vehicles.

According to Figure 8, the computational resource utilization of both the roadside unit and the helper vehicle increases as the average computational volume of the requesting vehicle continues to increase. When the average number of computations of the requesting vehicle is low, the computational resource utilization of the roadside unit fails to reach 100%. This is because, with a smaller computational delay gap, helper vehicles with higher link stability and trust assessment can provide better computational services for some of the requesting vehicles. In addition, as the average computation volume of the requesting vehicles continues to increase, the difference in computation delay between the roadside unit and the helper vehicles significantly increases when more vehicle users choose the roadside unit with lower computation delay as the target service node. At this time, the helper vehicles with higher link stability and trust assessment in the system start to share the computational pressure for the roadside unit, and the utilization rate of the computational resources of the helper vehicles is continuously improved. In the All-Edge scheme, the computational resource satisfaction rate of vehicle users decreases significantly with the increase in the average computational volume of requesting vehicles because the size of the average computational volume of requesting vehicles directly affects the computational load of the roadside unit. The scheme proposed in this paper can fully satisfy the computational resource demand of the requesting vehicles and effectively alleviate the computational load of each service node in the system.

Figure 9 compares the overall task length required with different schemes for different helper vehicle computing powers.



Figure 8. Service node utilization and computational resource satisfaction rate for different average computations of the requested vehicles

average computational volume of the requesting vehicle



Figure 9. Overall task duration with different helper vehicle computing power

The overall task duration required by both the All-Local and All-Edge schemes remains unchanged with increasing computational power of helper vehicles because neither baseline scheme considers roadside travelling vehicles as service nodes in the system, and thus the task duration in the system is not affected. In contrast, the overall task duration required by the scheme proposed in this paper decreases as the computational power of the helper vehicle continues to increase. When the computational power of helper vehicles is the lowest, it saves approximately 13% of the task duration compared to the All-Edge scheme, and when the computational power of helper vehicles is equal to that of roadside units, it saves approximately 30% of the task duration compared to the All-Edge scheme.

Figure 10 compares the overall task duration required with different schemes for different roadside unit computing powers.

The overall task duration required by the All-Local scheme remains unchanged because all the tasks generated by the requesting vehicles are handed over to the local terminals for computation, and the task duration is related only to the local terminal's computing power. Both the All-Edge and the scheme proposed in this paper decrease slowly as the roadside unit computational power increases, this is because for a fixed average computation amount of the requesting vehicle, as the computation power of the roadside unit increases significantly, the change in the computation delay required by the roadside unit decreases. For the All-Edge scheme, doubling the computational power of the roadside unit saves approximately 4% of the overall task duration. For the scheme proposed in this paper, doubling the computational power of the proposed scheme in this paper changes more gently. It can be seen that the scheme proposed in this paper can help the system to perform better when the computational power of the roadside unit is insufficient.

CONCLUSION

This paper investigates the problem of trusted computing offloading for multivehicle users in scenarios lacking roadside infrastructure. To this end, this paper acts as a service node to provide computing services to vehicular users from vehicles with free computing resources that are travelling or parked on the roadside and simultaneously proposes a trust evaluation model for mobile vehicular



Figure 10. Overall task duration for different curb unit computing powers

networks. In this paper, the trusted computing offloading problem for service nodes based on trust metrics is modelled as an integer planning problem, with a goal of minimizing the total delay of tasks performed by multivehicle users under the requirements of service node security and D2D link stability. To solve this problem, this paper proposes a parallel ADMM solution method that uses a convex difference algorithm to solve the subproblem. On this basis, this paper analyses the convergence and complexity of the algorithm and demonstrates the performance of the proposed method through simulation experiments. In the future, we aim to introduce additional effective strategies such as energy harvesting, service caching, and other technologies into the existing solution to further expand the applicability of the solution. In addition, we plan to introduce reinforcement learning methods into vehicular networks to better design offloading strategies.

CONFLICTS OF INTEREST

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

FUNDING STATEMENT

No funding was received for this work.

PROCESS DATES

Received: May 20, 2024, Revision: May 24, 2024, Accepted: May 26, 2024

CORRESPONDING AUTHOR

Correspondence should be addressed to Xiao Han (China, han_xiao@hrbeu.edu.cn)

REFERENCES

Boyd, N. S. P., Parikh, N., Chu, E. B., Peleato, B., & Eckstein, J. (2021). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, *3*(1), 1–122. 10.1561/2200000016

Chao, Z., Jin, T., Giancarlo, P., & Yu, X. (2019). Folo: Latency and quality optimized task allocation in vehicular fog computing. *IEEE Internet of Things Journal*, 6(3), 4150–4161. 10.1109/JIOT.2018.2875520

Che, O. E., Tuan, H., & Nguyen, H. H. (2014). Joint optimization of cooperative beamforming and relay assignment in multi-user wireless relay networks. *IEEE Transactions on Wireless Communications*, *13*(10), 5481–5495. 10.1109/TWC.2014.2324588

Chen, Y. X., Guo, H., & Liu, J. (2022, Dec. 4-8). *Efficient and trusted task offloading in vehicular edge computing networks* [Conference session]. IEEE Global Communications Conference, GLOBECOM, Rio de Janeiro, Brazil.

Dai, M. Y., Xu, D., Maharjan, S., & Zhang, Y. (2019). Joint load balancing and offloading in vehicular edge computing and networks. *IEEE Internet of Things Journal*, 6(3), 4377–4387. 10.1109/JIOT.2018.2876298

Du, L., Huo, R., Sun, C., Wang, S., & Huang, T. (2023). Adaptive joint placement of edge intelligence services in mobile edge computing. *Wireless Networks*, *3*(2), 799–817. 10.1007/s11276-023-03520-4

Fan, L. W., Liu, J., Hua, M., Wu, F., & Liu, Y. (2022). Joint task offloading and resource allocation for multiaccess edge computing assisted by parked and moving vehicles. *IEEE Transactions on Vehicular Technology*, 71(5), 5314–5330. 10.1109/TVT.2022.3149937

Fan, W., Su, Y., Liu, J., Li, S., Huang, W., Wu, F., & Liu, Y. (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. 10.1109/TITS.2022.3230430

Fan, Z. W., Hua, M., Zhang, Y., Su, Y., Li, X., Tang, B., Wu, F., & Liu, Y. (2023). Game-based task offloading and resource allocation for vehicular edge computing with edge-edge cooperation. *IEEE Transactions on Vehicular Technology*, 72(6), 857–7870. 10.1109/TVT.2023.3241286

Gao, X. J., Kuang, Z., Gao, J., & Zhao, L. (2023). Joint offloading scheduling and resource allocation in vehicular edge computing: A two layer solution. *IEEE Transactions on Vehicular Technology*, 72(3), 3999–4009. 10.1109/ TVT.2022.3220571

Hou, X., Ren, Z., Wang, J., Cheng, W., Ren, Y., Chen, K.-C., & Zhang, H. (2020). Reliable computation offloading for edge-computing-enabled software-defined IoV. *IEEE Internet of Things Journal*, 7(8), 7097–7111. 10.1109/JIOT.2020.2982292

Lin, Y. Y., Huang, J., Fan, C., & Chen, W. (2018). Local authentication and access control scheme in M2M communications with computation offloading. *IEEE Internet of Things Journal*, *5*(4), 3209–3219. 10.1109/JIOT.2018.2837163

Ma, Z. C., Zhu, J., Liu, M., Zhao, H., Liu, N., & Zou, X. (2021). Parking edge computing: Parked-vehicleassisted task offloading for urban VANETs. *IEEE Internet of Things Journal*, 8(11), 9344–9358. 10.1109/ JIOT.2021.3056396

Ren, J., Mahfujul, K. M., Lyu, F., Yue, S., & Zhang, Y. (2020). Joint channel allocation and resource management for stochastic computation offloading in MEC. *IEEE Transactions on Vehicular Technology*, 69(8), 8900–8913. 10.1109/TVT.2020.2997685

Shi, K. J., Du, J., Wang, J., & Yuan, J. (2020, May 25-28). *Distributed V2V computation offloading based on dynamic pricing using deep reinforcement learning* [Conference session]. IEEE Wireless Communications and Networking Conference, Seoul, South Korea.

Wang, S., Guo, Y., Zhang, N., Yang, P., Zhou, A., & Shen, X. (2019). Delay-aware microservice coordination in mobile edge computing: A reinforcement learning approach. *IEEE Transactions on Mobile Computing*, 20(3), 939–951. 10.1109/TMC.2019.2957804

Wright, P. S. E. (2023). Direct solution of the normal equations in interior point methods for convex transportation problems. *Operations Research Letters*, *51*(5), 469–472. 10.1016/j.orl.2023.07.001

Xu, Z. W., Dai, X., Zhao, C., Tang, Y., Cheng, J., Qu, Z., Zhang, T., & Lv, Y. (2023). Parallel testing for centralized traffic control systems of intelligent railways. *IEEE Transactions on Intelligent Vehicles*, 8(9), 4249–4262. 10.1109/TIV.2023.3305543

Xue, J., Wang, Q., Zhang, H., An, N., & An, C. (2023). Idle-parked vehicles assisted collaborative resource allocation in VEC based on Stackelberg game. *Ad Hoc Networks*, *142*, 103069. 10.1016/j.adhoc.2022.103069

Zhou, Y. Z., Chen, X., Li, E., Zeng, L., Luo, K., & Zhang, J. (2019). Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE*, *107*(8), 1738–1762. 10.1109/JPROC.2019.2918951

Zhu, X., Shu, J., & Jiang, H. (2023). Perception task offloading with collaborative computation for autonomous driving. *IEEE Journal on Selected Areas in Communications*, 41(2), 457–473. 10.1109/JSAC.2022.3227027

Xiao Han received her B.E. degree and M.S. from Jiangnan University, Wuxi, China, in 2011and 2014, respectively. She is currently working toward the Ph.D. degree in computer science and technology with the School of Harbin Engineering University, Harbin, China. Her research interests involve Machine learning, edge computing and crowdsensing.

Huiqiang Wang received his received M.E. and Ph.D. degrees from HEU in 1985 and 2005, respectively. From 2001 to 2002, he was at Queens University, Ontario, Canada, as a senior visiting scholar. Now, he is engaged in teaching and researching as a professor and a doctoral advisor at HEU. Up to now, he holds ten Chinese patents. His research interests involve network security, cognitive networks, and autonomic computing.

Guoliang Yang received his B.E. degree from HEU in 2020. Now, he is a postgraduate student in Harbin Engineering University in China and majoring in computer science and technology. His research interests involve edge computing and optimization.

Chengbo Wang received his B.E. degree from Shandong University of Finance and Economics in 2017. Now, he is a postgraduate student in Harbin Engineering University in China and majoring in computer science and technology. His research interests involve computation migration and optimization.