This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3250495

Many-to-Many Task Offloading in Vehicular Fog Computing: A Multi-Agent Deep Reinforcement Learning Approach

Zhiwei Wei, Bing Li, *Member, IEEE,* Rongqing Zhang, *Member, IEEE,*
Xiang Cheng, *Fellow, IEEE,* and Liuqing Yang, *Fellow, IEEE*

Abstract—Vehicular fog computing (VFC) has emerged as a promising solution to mitigate vehicular network computation load. In the hierarchical VFC, vehicles are employed as mobile fog nodes at the edge to provide reliable and low-latency services. Particularly, since privately-owned vehicles are rational nodes, their intentions for both computation provision and service demand should be considered instead of overestimating their willingness. To remunerate the participation intentions of vehicles as well as improve vehicular fog resource utilization in the large-scale VFC, the trading-based mechanism is a potential solution. In this paper, we propose a many-to-many task offloading framework based on the vehicular trading paradigm. This framework enables computational resource trading across different VFC subsystems and decides the multi-tier task offloading results based on the trading consensus. The trading process is viewed as a partially observable Markov decision process (POMDP) and a Multi-Agent Gated actor Attention Critic (MA-GAC) approach is designed to reach an effective and stable offload-and-serve cooperation among vehicles. Theoretical analyses and experiments verify the feasibility and efficiency of the proposed framework, and simulation results demonstrate that the coordinated MA-GAC approach not only benefits vehicles with higher long-term rewards but also optimizes the system social welfare in a distributed manner.

Index Terms—POMDP, task offloading, multi-agent deep reinforcement learning, many-to-many, vehicular fog computing.

1 Introduction

With the rapidly emerging vehicular applications and technologies [1], [2], the spectral and computational resources of major service providers and mobile end users are limited to fulfill the large gap between the performance gains and the delay-intensive requirements [3]. To meet the exponentially increasing demands for high-complexity but low-latency computations, fog computing (also referred to as edge computing) has been developed as a new generation paradigm to improve vehicular network capability [4]. However, the conventional fog computing paradigm relies heavily on the deployed stationary servers at roadside units (RSUs) for computation offloading [5], and the limited resources of those fog servers as well as the lack of service applicability due to the vehicles' high mobility may hinder the improvement of vehicular applications to a great extent [6]–[8]. To further exploit network capability and provide available resources with a satisfactory quality-of-service (QoS), vehicular fog computing (VFC) [9] has been proposed in recent years and greatly mitigated the vehicular network computation load.

As an integration of vehicular network and fog computing, the hierarchical VFC paradigm enables moving vehicles as vehicular fog nodes to provide a more flexible share of services for the mobile end users at the edge [9]. Based on the exploitation of the mobile service providers, a growing number of works start to focus on the VFC architecture in topics like privacy preservation [10], caching optimization [11], radio resource allocation [12], UAV-assisted data dissemination [13], and so on. Furthermore, since the VFC sinks down a lot of mobile computational resources by employing vehicular fog nodes, the many-to-many task offloading problem, where multiple vehicles as service demanders and multiple vehicles as computation providers coexist, has become a hot-spot recently [14]. In [15], Lin *et al.* proposed a distributed task processing scheme based on the fog-based base station (BS) to provide a scalable network management protocol from a global view. To optimize the QoS of tasks in the fog layer, Zhu *et al.* in [16] formulated the task allocation process to fog nodes as a bi-objective minimization problem and proposed an event-triggered dynamic task allocation framework for latency and quality optimization. In [17], a distributed task offloading scheme was proposed to select nearby vehicles with idle computational resources to calculate the tasks under latency constraints. To tackle the complicated task offloading relationship between multiple fog nodes and multiple tasks, the authors in [18] formulated the offloading process as an optimization problem and solved the problem cooperatively from an overlapping perspective

- *Zhiwei Wei, Bing Li, and Rongqing Zhang are with the School of Software Engineering, Tongji University, Shanghai 200092, China.*
  E-mail: {2031563, 1710056, rongqingz}@tongji.edu.cn
- *Xiang Cheng is with the School of Electronics, Peking University, Beijing 100871, China, and also with the Department of Broadband Communication, Peng Cheng Laboratory, Shenzhen 518055, China.*
  E-mail: xiangcheng@pku.edu.cn
- *Liuqing Yang is with the Internet of Things Thrust & Intelligent Transportation Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511458, China, and also with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong SAR 999077, China.*
  E-mail: lqyang@ust.hk

(Corresponding author: Rongqing Zhang.)

to allow flexible resource scheduling.

However, privately-owned vehicles are viewed as rational nodes concerning their selfish natures so it is not applicable to offload tasks directly to them by overestimating their willingness in the traditional mathematical optimization manners [15]–[18]. Instead, to implement the enrollment of vehicles as vehicular fog nodes, incentive mechanisms are welcomed as external-enforced agreements for fog entities and thus stimulate the vehicles to share their computational resources. Zhou *et al.* in [19] adopted a two-stage task offloading framework from a contract-matching integration perspective to realize effective computational resources recruitment and reliable task offloading. In [20], Nazih *et al.* proposed a game theoretic-based incentive mechanism to stimulate vehicles as fog nodes in VFC. In [21], Shi *et al.* proposed a centralized priority-aware task offloading scheme with a pricing mechanism and designed a deep reinforcement learning (DRL) approach to maximize the expected reward of the vehicular fog nodes. Inspired by the economic features of computational resources in task offloading, the authors in [22] turned to the trading paradigm and proposed a multiattribute-based double auction mechanism in VFC. From the existing works, one can observe that the incentive mechanisms efficiently provide a platform (e.g., trading market) for the participating vehicles to exchange their resources with monetary currency and aims to remunerate the spontaneous offload-and-serve behaviors of the vehicles as well as improve vehicular utility concerning both economic benefits and their QoS.

Taking the mutual offload-and-serve intentions of vehicles into account, the trading-based paradigm becomes a potential method to provide an efficient solution to the complicated but practical many-to-many task offloading problem in VFC. The studies of the trading-based paradigms [22]–[26] have paved the way for an orderly trading market for mobile computations where monetary currency acts as the pricing mechanism and marketing rules sustain efficient offloading decisions. Nonetheless, trading resources in VFC can bring several challenges and incur undesirable performance degradation due to the high mobility of vehicles and the dynamic vehicular environment.

- First, *trading failure* happens when the vehicles cannot reach a trading consensus because of their negative utilities or fail to transmit the tasks because of the varying vehicular network topology for computation and communication.
- Second, the excessive decision-making latency resulted from inefficient trading strategies can inevitably reduce the available time for task offloading and calculating process, which harms the real-time QoS requirements of vehicles.
- Third, considering the changing supply and demand of computational services in different periods, the fluctuation in pricing may also experience a series of peaks and troughs given the supply-demand relationship, which generates the *trading unfairness* in the system. For instance, the buyer may be provided with the price of \$0.5/GHz for CPU resources in one trading, but \$1/GHz in another trading, which is unfair from the perspective of buyers to pay for the services at higher prices.

Besides, vehicles are only aware of the surrounding vehicles' information, making the instant computation resource trading issues intractable under the large-scale dynamic VFC scenario via traditional trading-based methods. Motivated by the above-mentioned challenges and vehicular offload-and-serve intentions, in this paper[1], we investigate the many-to-many task offloading problem in a dynamic and hierarchical vehicular fog computing scenario with the consideration of individual rationality. The hierarchical scenario facilitates the trading-based framework with resources from different regions via the RSUs as resource retailers and allows flexible multi-tier resource trading in the overall system. We first employ a vehicle-to-vehicle (V2V) computational resource trading paradigm and propose a distributed V2V trading-based task offloading framework. In the proposed framework, vehicles as service demanders and vehicles as computation providers are motivated to cooperate for higher individual utilities in a more flexible and efficient manner subject to their two-sided cooperation willingness, lowering the possibility of trading failures. Then, to fully exploit the local computing potentials and reach an effective and stable cooperation deal in the real-time dynamic offloading environment, the trading process is viewed as a partially observable Markov decision process (POMDP) and a Multi-Agent Gated actor Attention Critic (MA-GAC) approach is designed to obtain the strategic trading, leading to an efficient offloading optimization process in a distributed manner. Furthermore, the MA-GAC approach extracts the latent knowledge from the trading histories and the current states, which enables the prediction of the trading trends in the system to mitigate trading unfairness. Finally, extensive experiments are conducted to verify the feasibility and superiority of the proposed framework with other benchmarks. The main contributions of this paper are summarized as follows:

1) We focus on the many-to-many task offloading problem by taking the individual intentions of both vehicles as service demanders and vehicles as computation providers in VFC. To tackle this complicated optimization problem, we propose a novel distributed V2V trading-based task offloading framework across multiple layers in VFC. The proposed framework enables cooperative computational resource management in the hierarchical VFC by providing a feasible and efficient trading mechanism integrating the coalitional game and the mid-market-rate (MMR) pricing. Then the multi-tier task offloading is executed by the coalition heads and the RSUs based on the determined trading consensus.
2) The dynamic trading strategies in the proposed framework render the sequential decision-making process for both the resource buyers (service demanders) and the resource sellers (computation providers). To learn the optimal strategy in a long run with efficiency, we model this multi-agent trading scenario as a POMDP and design a soft actor-critic (SAC) based multi-agent deep reinforcement learning (MADRL) approach, termed as MA-GAC approach, to learn the V2V trading strategy

[1] Part of this work has been accepted as the conference paper [27] in GLOBECOM 2022.

cooperatively in a distributed manner. The MA-GAC approach in the proposed framework leads to an improved many-to-many task offloading optimization process and encourages the participation of the vehicles.

3) Extensive experiments are executed and simulation results demonstrate the superiority of the proposed framework and the designed MA-GAC approach. The proposed MA-GAC can motivate vehicles to offload and serve by offering a higher individual long-term reward which consists of QoS and monetary revenues. Meanwhile, the social welfare of the system is also optimized by the joint effort of the participating agents in the hierarchical VFC system.

The rest of this paper is organized as follows. Section 2 introduces the hierarchical VFC system overview and formulates the many-to-many task offloading problem as an optimization programming problem. Section 3 proposes a task offloading framework based on the V2V trading paradigm to tackle the multi-tier task offloading problem in a distributed manner. Section 4 proposes the MA-GAC approach and Section 5 verifies the performance with other benchmarks. Section 6 concludes this paper.

2 System Overview and Problem Formulation

In this section, we first introduce the task offloading paradigm in the hierarchical VFC architecture. Then, we describe some relevant models and formulate the task offloading process as an optimization problem for each vehicle.

2.1 Task Offloading in the Hierarchical VFC Scenario

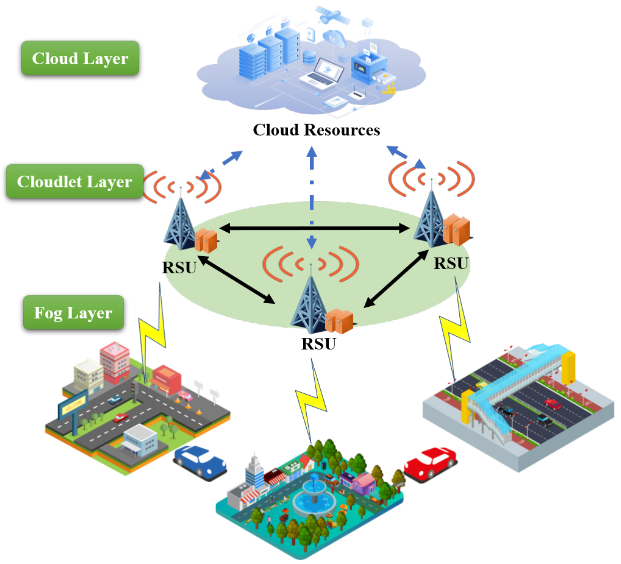


Fig. 1. The hierarchical VFC scenario consists of the cloud layer, cloudlet layer, and fog layer.

As shown in Fig. 1, the VFC scenario is a global hierarchical architecture composed of the cloud layer, cloudlet layer, and fog layer. The upper cloud layer aggregates abundant computational resources from cloud servers that are thousands of miles away from the end devices. The cloudlet layer consists of multiple service zones (i.e., VFC subsystem) that are manipulated by the RSUs (or BSs) and the computational resources of different service zones in the cloudlet layer are managed by the *coordinator servers*. The underlying fog layer consists of driving vehicles which can be categorized as service demanders and computation providers. The client vehicles (service demanders) broadcast their task profiles and the vehicular fog nodes (computation providers) share the computational resources on their willingness. By default, the computational resources are provided by the vehicular fog nodes or the RSUs in the fog layer for their closeness to the client vehicles, but it is also possible for the coordinator servers and the RSUs to allocate the resources from other service zones and the cloud layer with acceptable delays. Therefore, as for multiple service demanders and multiple computation providers, the service provision problem is regarded as a multi-tier many-to-many task offloading problem in the whole VFC system.

Such an offloading problem concerns not only the system dynamics (e.g., varying channel state information, vehicular network topology, computational capability distribution, etc.) but also the *incentives* and *individual intentions* of vehicles. As for the privately-owned vehicles, they are under no obligation to serve the client vehicles voluntarily. Therefore, an incentive mechanism is necessary based on which computational resources are treated as fee-paying services and the client vehicles ought to pay for the services. Then, relying on the incentive mechanism, the vehicles may show divergent preferences (i.e., individual intentions) while serving or offloading under different situations. For example, some client vehicles are likely to pay more for the urgent computational requirements while others may reject to offload tasks when the pricing of the resources is too high. Taking the individual intentions of all the vehicles into account, the self-interested task offloading decisions made by the vehicles may conflict with each other due to the mismatch between the limited resources and the excessive demands, which brings offloading failures and incurs the degradation in task QoS. What's more, this offloading decision-making process also causes high computation complexity in the real-time vehicular environment to find the optimal solution for those computation-limited client vehicles. Hence, based on the above-mentioned situations, there exists room for an offloading framework to address both the individual intentions and low-complexity decision-making of vehicles in the hierarchical VFC architecture.

2.2 System Model

We consider the set of RSUs as $\mathcal{R} = \{R_1, R_2, \cdots, R_M\}$, the set of vehicles as $\mathcal{V} = \{V_1, V_2, \cdots, V_N\}$, and the corresponding available computing capacity set as $\mathcal{F} = \{f_1^a, f_2^a, \cdots, f_N^a\}$. Then we introduce the mobility model, task model, communicating model, computing model, RSU model, cloud model, and trading model in the VFC system.

2.2.1 *Mobility Model*

We denote the velocity, direction, and location of vehicle V_i as v_i, dr_i, and loc_i, respectively. In this work, we focus on multiple vehicles within the communication range of the RSU, in which the traffic density and vehicular mobility jointly affect the offloading performance. Similar to [21], we refer to a free-flow traffic model and suppose that all the vehicles enter the service zone at an initial speed following the Gaussian distribution. The mean of this Gaussian distribution is determined by both the maximum permissible driving speed $v_{\max}$ and the vehicle density in the service zone, while the variance of the distribution is proportional to the mean value. The mean $\bar{v}$ and the variance σ_v follow:

$$\bar{v} = v_{\max}\left(1 - \frac{N}{N_{\max}}\right), \qquad \sigma_v = \alpha_v \bar{v} \tag{1}$$

where α_v is the scaling parameter, N is the number of vehicles in the service zone, and $N_{\max}$ is the maximum capacity of vehicles in the service zone. As a result, the initial velocity of vehicle V_i satisfies $v_i \sim \mathcal{N}(\bar{v}, \sigma_v)$, where $\mathcal{N}(\cdot,\cdot)$ is the Gaussian distribution.

After entering the service zone, each vehicle V_i enjoys a varying acceleration acc_i according to the distance from the vehicle V_j in front. If the vehicle V_i is too close to the front vehicle V_j, it chooses to brake with $-acc_i$; if the distance is very large, vehicle V_i speeds up with the acceleration acc_i. The velocity model of vehicle V_i is given by:

$$acc_i = \begin{cases} acc_0, & d_{i,j} > D_{d1} \\ 0, & d_{i,j} \in (D_{d2}, D_{d1}] \\ -acc_0, & d_{i,j} \le D_{d2} \end{cases} \tag{2}$$

where acc_0 is a fixed constant, $d_{i,j}$ is the distance between V_i and V_j, and D_{d1}, D_{d2} are the distance thresholds to denote whether the vehicle is speeding up or braking. Thereby, the location of vehicle V_i at time slot $t+1$ can be deduced by the states at the previous time slot t:

$$loc_{i,t} = loc_{i,t} + dr_{i,t}\left(v_{i,t}\Delta t + \frac{1}{2}\cdot acc_{i,t}\Delta t^2\right) \tag{3}$$

where Δt is the time slot length and the subscript t means the state at time slot t.

2.2.2 Task Model

In practice, the workload of the onboard applications always has sequential characteristics in a short period. For example, the drivers may use the onboard units (OBUs) on the vehicles for AR applications for a while, and therefore the *calculating conditions* (i.e., computing loads) of the OBUs during this period are different from the ones when the OBUs are idle. To build the task model properly, each vehicle V_i is assumed to be under a certain calculating condition ζ_i in each time slot related to its onboard applications, and the OBU generates the task based on the calculating condition. In this work, the generated task of V_i is divisible and can be featured by six attributes: tolerant service latency δ_i, required CPU frequency $cr_i \propto \zeta_i$, to-be-processed/returned data size up_i/dw_i, *delay-sensitive factor* ε_i, and *unit basic utility* u_i. Specifically, the unit basic utility is used to reflect the satisfaction brought by accomplishing the task (a similar design can be seen in [21]) and the delay-sensitive factor is a novel proposed attribute in this paper to reflect the divergent sensibility of tasks. For instance, the task with a higher delay-sensitive factor means that the driver shows a more pressing demand for computational services compared with the others.

Given the six attributes of tasks, we want to quantify the task QoS for mathematical optimization and performance evaluation. Driven by this motivation, we design a metric named *task utility* derived from the above six task attributes. In accordance with the practical scenario, the design of task utility should satisfy the following two rules: 1) Task utility is monotonically increasing to the shortened service latency. 2) There exists a marginal utility decrease when the latency is further shortened. Hence, a reasonable and suitable task utility function is given as:

$$U_i = \begin{cases} \dfrac{\delta_i}{T_i} u_i cr_i, & T_i \ge \delta_i \\ u_i cr_i + \varepsilon_i \log\left(1 + \delta_i - T_i\right), & T_i < \delta_i \end{cases} \tag{4}$$

where T_i is the service delay of the task. According to (4), when the task cannot be accomplished within the tolerant delay δ_i, the task utility is obtained at a discount; otherwise, the utility increment enjoys a logarithmic relation to the reduced latency[2]. A similar design can also be seen in [21].

To better understand the essence of the proposed task model, we summarize the characteristics below:

- *Partially Offloading:* The tasks in this paper is divisible and can be partially offloaded to multiple vehicular fog nodes with all the task data.
- *Calculating Condition:* The OBU of each vehicle is under a certain calculating condition during a period and the generated tasks at each time slot are positively related to the calculating condition.
- *Delay-Sensitive Factor:* The client vehicle's task QoS is influenced by its delay-sensitive factor, while the task with a higher delay-sensitive factor demands more computational resources compared with the others.
- *Task Utility:* Task utility is a vital concept to quantify the QoS. Large task utility brings great satisfaction to the client vehicle.

2.2.3 Communicating Model

The communicating links in the system consist of two ways to transmit data when the vehicles decide to offload the tasks, which are the V2V links and the vehicle-to-RSU (V2R) links. If vehicle V_i chooses V2R communications, the achievable information rate between vehicle V_i and the RSU:

$$r_{i,R} = B_{i,R}\log_2\left(1 + \frac{P^t_{i,R} d^{-\alpha}_{i,R} |h_{i,R}|^2}{N_0}\right) \tag{5}$$

where $B_{i,R}$ is the allocated V2R channel bandwidth between vehicle V_i and the RSU, $P^t_{i,R}$ is the transmission power, $d_{i,R}$ is the distance, α is the path loss exponent, N_0 is the power noise, and $h_{i,R}$ is the Rayleigh channel coefficient.

The V2V channel is assumed as the one-hop communicating channel in this paper, which indicates that the V2V links are effective only when the distance between two vehicles is within a given threshold. Similarly, the achievable information rate between vehicle V_i and V_j is calculated as:

$$r_{i,j} = B_{i,j}\log_2\left(1 + \frac{P^t_{i,j} d^{-\alpha}_{i,j} |h_{i.j}|^2}{N_0}\right). \tag{6}$$

The channel models are constructed without the co-channel interference from different vehicles by assuming that each vehicle is allocated an orthogonal spectrum resource block.

[2] The Logarithmic function is an optional choice in this paper to formulate the task utility satisfying the above-mentioned two rules.

Therefore, the roundtrip communicating delay from V_i to V_j and from V_i to the RSU for data transmission of the task is estimated as:

$$T^{tran}_{i,j} = \frac{x_{i,j}\left(dw_i + up_i\right)}{r_{i,j}}, \qquad T^{tran}_{i,R} = \frac{x_{i,R}\left(dw_i + up_i\right)}{r_{i,R}} \tag{7}$$

where $x_{i,j}, x_{i,R}$ are the offloading decision factors. $x_{i,j} = 1$ means the task is offloaded from vehicle V_i to V_j and $x_{i,j} = 0$ otherwise.

2.2.4 Computng Model

The computing model in this paper mainly consists of two parts, which are *local execution* and *offloading*. For each vehicle, the computational requirements can be executed by its OBU locally and the transmission latency is thus saved. The local execution latency of vehicle V_i is defined as:

$$T^{loc}_i = \frac{\left(1 - \sum_{V_j \in \mathcal{V}} \eta_{i,j}\right) cr_i}{f^a_i - \sum_{V_j \in \mathcal{V}} f_{i,j}} \tag{8}$$

where $\eta_{i,j} \in [0, 1]$ denotes the task partition offloaded from vehicle V_i to V_j and $f_{i,j} \in [0, f^a_i]$ is the devoted CPU resources of V_i towards V_j. Note that $\sum_{V_j \in \mathcal{V}} \eta_{i,j} \in [0, 1]$ and $\sum_{V_j \in \mathcal{V}} f_{i,j} \in [0, f^a_i]$. Apparently, the local execution latency depends on the computational requirements of the not-offloaded task partition and the remaining computational resources of V_i.

If the limited resources of the OBU cannot support the excessive requirements, the client vehicle V_i divides the task into sub-tasks and offloads the partitions to nearby vehicular fog nodes. When the sub-task with $\eta_{i,j}$ computational requirements is offloaded from vehicle V_i to V_j, the execution time depends on the required CPU cycles $\eta_{i,j} cr_i$ and the allocated computational capacity $f_{j,i}$ from vehicle V_j. The execution latency for vehicle V_j to calculate the task of V_i can be obtained by:

$$T^{off}_{i,j} = \frac{x_{i,j} \eta_{i,j} cr_i}{f_{j,i}}. \tag{9}$$

Therefore, the service latency T_i of the task is restricted by the maximum service latency of all the sub-tasks:

$$T_i = \max\left\{T^{loc}_i, \max_{V_j \in \mathcal{V}} \left\{T^{off}_{i,j} + T^{tran}_{i,j}\right\}\right\}. \tag{10}$$

2.2.5 RSU Model

In this paper, the RSUs are equipped with limited computational resources and are not capable of calculating the tasks. Instead of being edge computing servers, the RSU of each service zone collects information from all the local vehicles, including the computing conditions, locations, CSI, and so on. The RSUs exchange the service zone profiles in the cloudlet layer based on wired links and the information processing rate relies on the network infrastructures [29], [30]. Furthermore, the RSUs can act as the gateway of each service zone and help transmit the tasks from the fog layer to the upper layers. As such, when the client vehicle chooses to offload the task to the RSU, the RSU decides to determine whether to offload it intra-zone, inter-zone, or to the cloud. If there are other vehicular fog nodes in the service zone willing to provide reliable computational services without a direct one-hop link to the client vehicle, the RSU receives the task and transmits it to the vehicular fog node. Otherwise, the RSU inquires about the computational resources of other service zones and cloud servers. The transmission latency of the task from the vehicle V_i in the coverage of the RSU R_m to the vehicle V_j in the zone of R_m is defined as:

$$T^{tran}_{i,j} = T^{tran}_{i,R_m} + \frac{x_{R_m,R_n}\left(dw_i + up_i\right)}{r_{R_m,R_n}} + T^{tran}_{j,R_n} \tag{11}$$

where x_{R_m,R_n} indicates whether the task is offloaded from the RSU R_m to R_n and r_{R_m,R_n} is the information processing rate between the RSUs R_m and R_n.

2.2.6 Cloud Model

If the computational resources at the cloudlet layer are unable to support the task offloading, the RSUs transmit the tasks to the remote cloud servers for calculation. Similar to the most existing works [8], [23], [29], the resources at the cloud layer are supposed to be enough to tackle the computational tasks, and hereby the execution latency is ignored. Therefore, the service latency of V_i's task offloaded from the RSU R_m to the cloud only depends on the transmission latency, which is formulated as:

$$T_i = \frac{x_{R_m,cloud}\left(dw_i + up_i\right)}{r_{R_m,cloud}} + T^{tran}_{i,R_m} \tag{12}$$

where $x_{R_m,cloud}$ indicates whether V_i's task is offloaded from the RSU R_m to the cloud and $r_{R_m,cloud}$ is the information processing rate between the RSU R_m and the cloud.

2.2.7 Trading Model

To reflect the offloading relationship between vehicles with the consideration of individual intentions, we introduce the *trading model* for vehicles. Since the offloading of tasks shortens service latency but spends monetary payment for client vehicles and the sharing of computational resources benefits the monetary revenues but reduces local task utility for vehicular fog nodes, a trading model is proposed to evaluate the trade-off between the task utility and monetary currency. The monetary pricing and payment are termed as *external cost* and the external cost of vehicle V_i for resource trading with V_j is given by:

$$ext(V_i, V_j) = p^b_{i,j} f_{j,i} - p^s_{i,j} f_{i,j} \tag{13}$$

where $p^b_{i,j}$ and $p^s_{i,j}$ are the negotiated pricing or payment of unit computational resource between V_i and V_j as service providers and service requesters, respectively. Note that once $p^b_{i,j}$ and $p^s_{i,j}$ are negotiated by the mutual sides, V_i and V_j should trade at these pre-determined prices during a period for a certain quantity of resources until the next trading happens. The external cost of vehicle V_i with the cloud can be deduced in a similar way and the (13) is also used to include the cloud offloading situations in the following context without any ambiguity.

Finally, we combine the weighted task utility and external cost together as the trading objective for each vehicle V_i at time slot t:

$$obj_t(V_i) = U_{i,t} - \omega \sum_{V_j \in \mathcal{V}} ext_t(V_i, V_j) \tag{14}$$

where ω is the scaling factor to tradeoff the importance between the external cost and task utility, and the subscript t is used to denote the parameter at time slot t.

2.3 Problem Formulation

From the perspective of each vehicle V_i, the task offloading process to maximize the trading objective in the long run can be formulated as the following optimization problem:

$$\mathbf{P1}: \quad \max_{\mathbf{x}_t, \mathbf{f}_t, \eta_t} \quad \sum_{t=1}^{T} obj_t(V_i)$$

$$\begin{aligned}
\text{s.t.} \quad & C1: T_{i,j}^{off} + T_{i,j}^{tran} \leq \delta_{i,t}, \qquad \forall V_j \in \mathcal{V}, \forall t \in [1,T] \\
& C2: \sum_{V_j \in \mathcal{V}} f_{i,j,t} \leq f_{i,t}^{a}, \qquad \forall t \in [1,T] \\
& C3: \sum_{V_j \in \mathcal{V}} \eta_{i,j,t} \leq 1, \qquad \forall t \in [1,T] \\
& C4: f_{i,j,t} \in [0, f_{i,t}^{a}], \qquad \forall V_j \in \mathcal{V}, \forall t \in [1,T] \\
& C5: \eta_{i,j,t} \in [0,1], \qquad \forall V_j \in \mathcal{V}, \forall t \in [1,T] \\
& C6: x_{i,j,t}, x_{R_m,R_n,t}, x_{R_m,cloud,t} \in \{0,1\}, \\
& \qquad \qquad \forall V_j \in \mathcal{V}, R_n \in \mathcal{R}, \forall t \in [1,T].
\end{aligned}$$

The subscript t is used to denote the value of parameters at time slot t. In the optimization problem **P1**, $C1$ is the service latency constraint to guarantee that the offloaded task partitions should be finished within their deadlines; $C2$ is the computational resource constraint that requires the shared computational resources less than the available ones of V_i; $C3$ is the task partition constraint that requires the divided sub-tasks less than the original task; and $C4, C5, C6$ are the range constraints on the variable vectors $\mathbf{f}_t$, η_t, and $\mathbf{x}_t$.

However, the objective of each vehicle easily conflicts with the others restricted to the limited local resources and individual intentions. For example, the vehicles as service buyers always want to purchase the resources at low unit prices, which may harm the interests of the service sellers. Besides, the temporal effects of the trading results let the optimization problem even more complicated to solve, which poses challenges on the real-time decision-making scenario. Under this circumstance, solving the optimization problem **P1** for all the vehicles from a centralized resource-management view becomes intractable, and the conventional distributed trading manners which inquire the buyers and sellers to seek the trading consensus pair-by-pair are also not plausible due to its inefficiency and difficulty to converge. Therefore, we propose a distributed V2V trading-based task offloading framework with cooperation among vehicles to solve the problem **P1**.

3 A Distributed Many-to-Many Task Offloading Framework

In this section, we provide a multi-tier task offloading framework based on the V2V computational resource trading paradigm with a designed cooperative coalitional game and the MMR pricing mechanism to solve the many-to-many task offloading problem.

3.1 Framework Overview

As shown in Fig. 2, we propose a distributed many-to-many task offloading framework based on the V2V computational resource trading paradigm in the hierarchical VFC. As mentioned above, the cloudlet layer in VFC consists of multiple service zones and the RSUs in different service zones are connected with the neighboring ones through wired links. In such context, the RSU acts as the conduit and gateway for computational resource provision in each service zone, and thus can take the role of a third-party dependable resource retailer in the proposed V2V trading paradigm. The resource retailer needs to decide the unit sell/buy price of computational resources, and it is noteworthy that the unit sell/buy price determined by the RSU is relatively low/high for the sake of budget balance. Thereby vehicles are more motivated to trade with others directly for lower external costs.

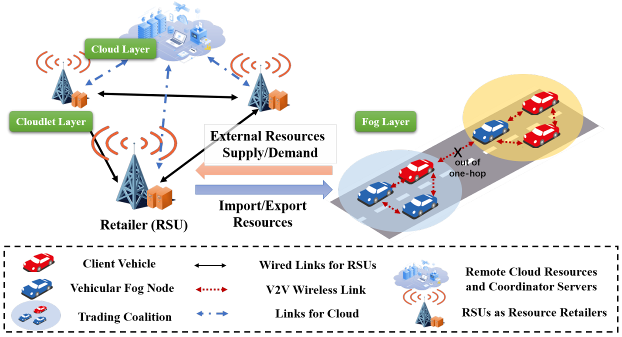


Fig. 2. Many-to-many task offloading based on V2V computational resource trading paradigm in the hierarchical VFC architecture.

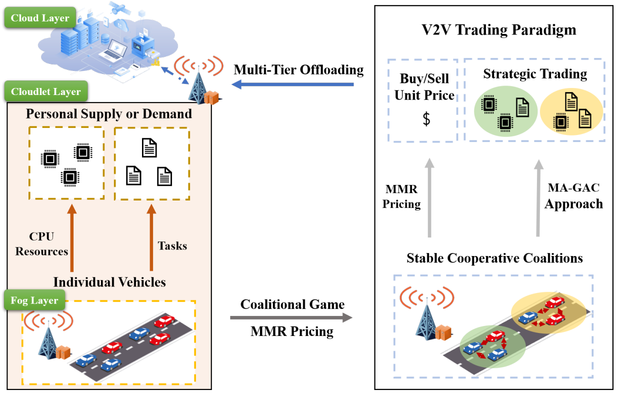


Fig. 3. The illustration of the modules in the distributed many-to-many task offloading framework.

During the V2V trading process, two main factors directly affect the framework feasibility and efficiency: the unstable V2V communication links and the trading unit price determination agreement. Vehicles without stable links cannot maintain dependable trading relationships with others and the unfair price determination leads to a degradation in trading intentions. To tackle these two issues, cooperation among vehicles is adopted as a feasible way to operate the resources with stability and efficiency [24]. In this paper, vehicles are motivated to form coalitions to operate the resources cooperatively and transmit data with reliability in the fog layer. Then, to achieve a stable coalitional structure and avoid the dynamics resulted from the changing topology of vehicles, a cooperative coalitional game is proposed and the MMR pricing mechanism is utilized as the revenue allocation strategy in the coalitional core. Based on the self-organized coalition structures and the obtained trading consensus, a multi-tier task offloading strategy is thereby taken by the RSU to execute cross-layer optimization in the whole system. The structure of the proposed framework is shown in Fig. 3.

3.2 Coalitional Game Formulation

Coalitional game theory is introduced to formulate effective coalitions in which vehicles can operate computational resources cooperatively.

3.2.1 Coalition Definition

A coalitional game $\mathcal{G} = \{\mathcal{V}, v\}$ in a resource trading framework is characterized by a set of vehicles $\mathcal{V}$ and a value function $v(\cdot)$ which projects coalition utility to a numerical scalar. To define a proper value function, we first discuss the external cost for coalition C_q as:

$$ext(C_q) = p_R^b \left[\sum_{V_i \in C_q} l_i \right]^+ + p_R^s \left[\sum_{V_i \in C_q} l_i \right]^- \tag{15}$$

where p_R^b and p_R^s are the buy and sell prices of computational resources decided by the RSU, and l_i is the declared trading intention of vehicle V_i. $l_i > 0$ means that the client vehicle V_i demands for l_i computational resources while $l_i < 0$ represents that the vehicular fog node V_i are willing to provide $-l_i$ computational resources in each time slot. (15) reflects the capability of a coalition to cooperatively deal with the offload-and-serve requirements of its members and a lower coalitional external cost shows less dependence on the RSU. Based on the coalitional external cost, we define the value function $v(C_q)$ as:

$$v(C_q) = \sum_{V_i \in C_q} ext\left(\{V_i\}\right) - ext(C_q). \tag{16}$$

The value function quantifies the saved external cost for vehicles by deriving the differences between trading with the RSU and trading with other local vehicles directly.

3.2.2 Coalition Head Selection

Next, we give the preconditions of a vehicular coalition. Similar to the definition of connectivity in [28], in this paper, we assume that vehicles form a coalition only when they are connected with at least one coalition member. To maintain the coalition, collect the member information, and further schedule the resources, a *head vehicle* is selected in each coalition for signaling and coordination. Since the coalition head is responsible for scheduling the intra-coalition resources, we select a head vehicle with the shortest total communication distance over all the coalition members, which is given by:

$$V_{C_q}^{head} = \arg\min_{V_i \in C_q} \left\{ \sum_{V_j \in C_q} d_{i,j} \right\}. \tag{17}$$

3.2.3 Coalition Stability

The coalition stability affects the complexity of the coalition formation process greatly and a coalition definition with stability assists the coalition structure to converge to the Nash-Equilibrium fast. To verify the coalition stability, we first prove that the proposed coalitional game is theoretically superadditive.

Theorem 1 *(Superadditive) The coalitional game with value function (16) is superadditive, i.e.,* $\forall C_q, C_w$, $v(C_q \cup C_w) \geq v(C_q) + v(C_w)$.

Proof According to the definition of $ext(\cdot)$ function in (15), we can conclude that:

$$\sum_{V_i \in C_q \cup C_w} ext\left(\{V_i\}\right) = \sum_{V_i \in C_q} ext\left(\{V_i\}\right) + \sum_{V_i \in C_w} ext\left(\{V_i\}\right)$$
$$ext(C_q \cup C_w) \leq ext(C_q) + ext(C_w). \tag{18}$$

Therefore, it is proved that $v(C_q \cup C_w) \geq v(C_q) + v(C_w)$. ■

Theorem 1 implies that the vehicles in the local service zone are always encouraged to form grand coalitions and save external costs. However, it is rigorous for all the vehicles to organize a grand coalition considering communication dynamics in a practical scenario. To be realistic, we assume that the trading relationship between any client vehicle and vehicular fog node lasts for a certain period T^{trade}. During the trading period T^{trade}, vehicles must form feasible coalitions in which the member vehicle V_i can maintain the wireless link with at least one other member vehicle V_j:

$$\left\| loc_{i,(t+T^{trade})} - loc_{j,(t+T^{trade})} \right\|_2 \leq D_0 \tag{19}$$

where D_0 is the distance threshold for V2V communicating.

Then we discuss the stability of the grand feasible coalitions. To maintain the stability of coalitions, the revenues (i.e., valuation calculated by (16)) of the coalition need to be allocated among all the members in such a way that no individual of the coalition tends to abandon the grand coalition for further benefits [24]. The set of allocation strategies is termed as the *core*. For instance, if the local trading price in the coalition is close to the buy price provided by the RSU, the vehicular fog nodes are satisfied while the buyers may have no motivation to stay in the coalition since the benefits brought by the coalition is too tiny; similarly, if the local trading price is close to the sell price of the RSU, the vehicular fog nodes are not encouraged to serve in the coalition. The existence of coalitional stability is addressed by the following theorem.

Theorem 2 *(Existence of Stability) Given buy price* p_R^b *and sell price* p_R^s *provided by the RSU, the coalition* C_q *has a non-empty core if and only if the buy price* p_i^b *and sell price* p_i^s *of vehicle* $V_i \in C_q$ *meet the following condition:*

$$p_R^s \leq p_i^s \leq p_i^b \leq p_R^b. \tag{20}$$

Proof A similar proof can be seen in Section V-A in [24]. ■

According to Theorem 2, there exists at least one revenue allocation strategy that can form stable and feasible grand coalitions. The revenue allocation strategy is specified as a price determination mechanism in the V2V trading paradigm and we propose to use the MMR pricing mechanism to remunerate the participation of vehicles [25].

3.3 Pricing Mechanism

In this paper, the MMR pricing mechanism is applied to remunerate the participation of vehicles for stable coalition structures. Denote the local demand and supply of coalition

C_q as $D_q = \sum_{V_i \in C_q} [l_i]^+$ and $S_q = -\sum_{V_i \in C_q} [l_i]^-$. The MMR mechanism set the local buy price p_i^b and sell price p_i^s of vehicle $V_i \in C_q$ under three specific cases[3]:

Case 1: If the local demand equals the local supply ($D_q = S_q$) in the coalition, the external requirements of the coalition are set to be zero. In this case, the local computational resources are sold to the client vehicles, and the local buy price and local sell price are set to be the middle price provided by the RSU:

$$p_i^b = p_i^s = p_R^{mid} = \frac{p_R^b + p_R^s}{2}, \forall V_i \in C_q \tag{21}$$

where p_R^{mid} is given as the middle price of p_R^b and p_R^s.

Case 2: If the local demand surpasses the local supply ($D_q > S_q$), in other words, the client vehicles need to buy the deficit computational resources from the BS. Since the buyers request more than the sellers can offer, the extra payment is proportionally shared by all the buyers while the sellers are paid at the middle price. Hence, the local buy price and the local sell price are defined as:

$$p_i^b = \frac{p_R^{mid} S_q + p_R^b (D_q - S_q)}{D_q}, \quad p_i^s = p_R^{mid}, \forall V_i \in C_q. \tag{22}$$

Case 3: If the local demand is less than the local supply in the coalition ($D_q < S_q$), the vehicular fog nodes need to provide the computational resources to the client vehicles first and supply the rest of the resources to the RSU. Similarly, the reduced revenue is shared by all the sellers, and thus the local unit price is defined as:

$$p_i^b = p_R^{mid}, \quad p_i^s = \frac{p_R^{mid} D_q + p_R^s (S_q - D_q)}{S_q}, \forall V_i \in C_q. \tag{23}$$

Theorem 3 *(MMR Stability) The MMR mechanism provides a non-empty core to the cooperative coalitional game.*

Proof The case in which the local demand surpasses the local supply ($D_q > S_q$) is illustrated for the coalition C_q. In such a case, the revenue allocation strategy of the MMR mechanism is to distribute the payment of the deficit computational resources proportionally to the buyers. Since the sellers in the coalition C_q are paid at the middle price, the sellers can endure participating in any subgroup C_w only when the demand is no less than the supply $D_w \geq S_w$. Meanwhile, it can be concluded from Theorem 1 that $v(C_w) + v(C_q \setminus C_w) \leq v(C_q)$. Therefore, the buyers inevitably spend more in any subgroup of coalition C_q since the payment of deficit $-v(C_w)$ is distributed in proportion to their personal requirements. The MMR mechanism is proven to provide coalitional stability. ■

The V2V trading-based task offloading framework is demonstrated in Algorithm 1. Given the V2V trading paradigm, the objective of each V_i is thus transformed from **P1** to the form with regard to l_i:

$$\max_{l_i} \sum_{t \in \mathcal{T}} \left[U_i - \omega \left(p_i^b [l_i]^+ + p_i^s [l_i]^- \right) \right]. \tag{24}$$

Algorithm 1 V2V Trading-Based Framework

```
for each time slot $t$ do
    if $t\%T^{trade} == 0$ then
        Each vehicle $V_i$ determines its trading intention $l_i$.
        Vehicles form feasible coalitions according to (19).
        Each coalition $C_q$ determines the local price according to (21)~(23).
    end if
    for each coalition $C_q$ in the service zone do
        Task offloading by the coalition head.
        Report computational requirements or computational resources to the RSU.
    end for
    Multi-tier task offloading by the RSU.
end for
```

3.4 Multi-Tier Task Offloading

Relying on the hierarchical VFC architecture and the V2V trading paradigm, the proposed framework allows a multi-tier task offloading process which can be categorized into four offloading modes: *intra-coalition mode*, *intra-RSU mode*, *inter-RSU mode*, and *cloud mode*. By selecting the intra-coalition mode, vehicles form coalitions to trade resources and offload the tasks within the coalitions during the trading time T^{trade}. For each coalition C_q in the coverage of the RSU R_m, the coalition head decides the offloading relationship between client vehicles and vehicular fog nodes. Since the target of the coalition head is to best utilize the resources of the member vehicles, the offloading optimization problem is formulated as follows:

$$\begin{aligned}
\textbf{P2}: \quad & \max_{\mathbf{x},\mathbf{f}} \sum_{V_i \in C_q} \left[\xi_i^{(1)} U_i - (1 - o_i^{(1)}) U^p \right] \\
\text{s.t.} \quad & C'1: \sum_{V_j \in C_q} f_{i,j} \leq [l_i]^+, \qquad \forall V_i \in C_q \\
& C'2: \sum_{V_j \in C_q} f_{j,i} \geq [l_i]^-, \qquad \forall V_i \in C_q \\
& C'3: f_{i,j} \in [0, [l_i]^+], \qquad \forall V_i, V_j \in C_q \\
& C'4: x_{i,j} \in \{0, 1\}, \qquad \forall V_i, V_j \in C_q \\
& C'5: \xi_i^{(1)} \in \{0, 1\}, \qquad \forall V_i \in C_q
\end{aligned}$$

where $\xi_i^{(1)}$ indicates whether the task of V_i is offloaded through the intra-coalition mode and U^p is a punishment item to denote the extra costs for selecting other modes. As for the client vehicles V_i, the partition of the tasks is proportional to the allocated CPU resources $f_{j,i}, \forall V_j$ to compute the task in parallel and shorten the computation latency of the task, which is the reason why we do not use $\eta_{i,j}$ as the variables in **P2**. After obtaining the solution of **P2**, the head vehicle offloads the tasks and reports the information about computational requirements (**Case 1** in MMR pricing) or the computational resources (**Case 3** in MMR pricing) to the RSU.

When the RSU receives information and tasks from the coalition head, it can offload the tasks to another coalition head in the same service zone via intra-RSU mode. If the local vehicular fog nodes cannot serve the client vehicles, the inter-RSU mode is activated, and the RSU transmits the tasks to the nearby RSUs servers via the coordinator

[3] We omit the subscript j of $p_{i,j}^s$ and $p_{i,j}^b$ since vehicle V_i trades resources at the same buy or sell price in the coalition based on the MMR mechanism.

Algorithm 2 Multi-Tier Task Offloading

```
for each time slot $t$ do
    Each client vehicle generates tasks.
    for each coalition $C_q$ do
        Coalition head determines the intra-coalition offloading results by solving P2.
        if $C_q$ satisfies Case 1 or Case 3 in MMR mechanism then
            Coalition head of $C_q$ sends information about the tasks and computational resources to the RSU.
        end if
    end for
    Each RSU $R_m$ determines the offloading results by solving P3.
    Each client vehicle offloads the task.
    Each vehicular fog node or the RSU returns the calculated results to the client vehicle.
end for
```

servers. Finally, if the resources at the edge are limited, the cloud mode is chosen. As for the RSU R_m, we denote the set of buyers towards the RSU as $\mathcal{B}_{R_m}$ and the set of sellers towards the RSU as $\mathcal{S}_{R_m}$. Then the multi-tier offloading optimization problem of the RSU R_m is formulated as **P3**:

$$\begin{aligned}\textbf{P3}: \quad & \max_{\mathbf{x},\mathbf{f}} \sum_{V_i \in \mathcal{B}_{R_m}} U_i \\ \text{s.t.} \quad & C'1 \sim C'4 \\ & C''5: \xi_i^{(2)} + \xi_i^{(3)} + \xi_i^{(4)} = 1, \qquad \forall V_i \in \mathcal{B}_{R_m} \\ & C''6: \xi_i^{(k)} \in \{0,1\}, \qquad \forall V_i \in \mathcal{V}, k = 2,3,4\end{aligned}$$

where $\xi_i^{(2)}, \xi_i^{(3)}, \xi_i^{(4)}$ indicate whether vehicle V_i selects intra-RSU mode, inter-RSU mode, and cloud mode, respectively. All the relevant information is collected by the coalition head and summarized by the local RSU in each service zone. Algorithm 2 shows the details of the offloading process in the hierarchical VFC architecture, and the optimization problems **P2**, **P3** are solved by the particle swarm optimization (PSO)-based heuristic method similar to the method in [23] for its simplicity and feasibility.

4 Multi-Agent Deep Reinforcement Learning-Based Trading Strategy

In this section, we model the multi-agent trading scenario as a POMDP and design a SAC-based MADRL approach with the attention mechanism and gated recurrent unit (GRU), termed as MA-GAC approach, to learn the strategy of trading intention l_i for each vehicle V_i cooperatively.

4.1 POMDP Formulation

The coordinated many-to-many computational resource trading problem in the fog layer renders a dynamic decision-making process involving multiple agents. Nonetheless, confronted with the dynamics in the V2V trading paradigm and computing conditions, it is difficult to optimize the objective of each vehicle in conventional distributed trading manners. Moreover, the agents in this scenario need to take into account and interact with not only the dynamic environment but also other agents. To fully exploit the local computing potentials, in this paper, we model this multi-agent decision-making process as a POMDP and execute the MADRL learning approach to learn the coordinated computational resources management in a V2V trading way. The main elements associated with the POMDP formulation are listed as follows:

4.1.1 Observation

The observation $o_{i,t}$ for each vehicle V_i at each time slot t contains the following features: the time step identifier t, the local sell price history $p^s_{i,t-\iota:t}$, the local buy price history $p^b_{i,t-\iota:t}$, the local amount of supply resources $n^s_{i,j,\iota:t}$, the local demanding resources $n^b_{i,j,\iota:t}$, the current computing frequency $\{f_{j,t}\}$ of all the surrounding vehicles (vehicles within one-hop distance D_0), the current calculating condition $\{\zeta_{j,t}\}$ of all the surrounding vehicles, current location loc_i, current velocity v_i, current unit basic utility u_i, and delay-sensitive factor ε_i. To foster more efficient computational resource management decisions from the trading paradigm, the observation is incorporated with a length of ι history trading information, and the GRU network is employed to learn the history information and extracts the features containing the future trends. Let $o_{i,t}$ denote the observation of each vehicle V_i at time slot t:

$$\begin{aligned}o_{i,t} = [t, p^s_{i,t-\iota:t}, p^b_{i,t-\iota:t}, n^s_{i,j,t-\iota:t}, \\ n^b_{i,j,t-\iota:t}, \{f_{j,t}\}, \{\zeta_{j,t}\}, loc_i, v_i, u_i, \varepsilon_i].\end{aligned} \quad (25)$$

Taking local observations of all the agents, the global state space is deduced by $O_t = \{o_{1,t}, o_{2,t}, \cdots, o_{N,t}\}$.

4.1.2 Action

The action $a_{i,t}$ of each vehicle V_i at time slot t consists of its discrete deciding action $x_{i,t} \in \{0,1\}$ to decide whether to trade and a continuous managing action $l_{i,t}$ to present the amount of computational resources to sell or buy as trading intention. Note that the sellers can not offer computational resources beyond the limit. For vehicles V_i with computational resources or computational requests, the action $a_{i,t}$ is expressed as:

$$a_{i,t} = [x_{i,t}, l_{i,t}] \quad (26)$$

and the global action space is $A_t = \{a_{1,t}, a_{2,t}, \cdots, a_{N,t}\}$.

4.1.3 Reward

As mentioned before, each vehicle V_i is in pursuit of not only the QoS of tasks but also the minimized external costs. Therefore, the reward design should concern the gain or loss in not only the task utility but also the external cost:

$$Re^{QoS}_{i,t} = U_{i,t} - u_{i,t} cr_{i,t} \quad (27)$$

$$Re^{cost}_{i,t} = -\omega \left(p^b_{i,t} \left[l_{i,t} \right]^+ + p^s_{i,t} \left[l_{i,t} \right]^- \right). \quad (28)$$

Since the offloading process happens in each time slot during the trading time, the task utility at time slot t is also affected by the previous decisions. To avoid the randomness brought by the generating task utility and properly account for the advantages brought by previous actions, the QoS reward Re^{QoS} is calculated as the gained task utility minus the utility $u_{i,t} cr_{i,t}$. The final reward function is expressed as:

$$Re_{i,t} = Re^{QoS}_{i,t} + Re^{cost}_{i,t} \quad (29)$$

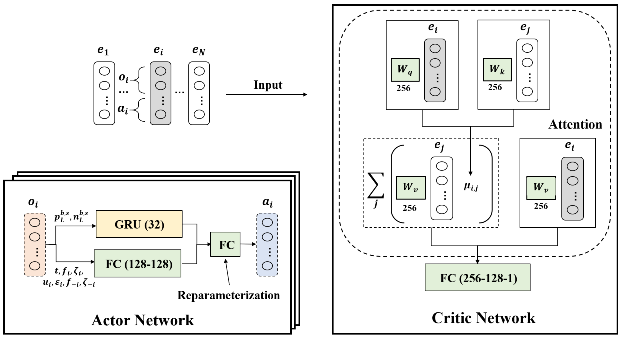


Fig. 4. Actor network and critic network in the proposed MA-GAC approach.

and the global reward at time slot t is thereby denoted by the vector $Re_t = \{Re_{1,t}, Re_{2,t}, \cdots, Re_{N,t}\}$.

4.2 MA-GAC Approach

The proposed distributed task offloading framework based on the V2V trading paradigm inherently renders a dynamic decision-making process among multiple learning agents. To better incorporate the learning process of each other and to fully exploit the local computational potentials, a tailored MADRL approach, namely the multi-agent gated actor attention critic (MA-GAC) method based on the centralized training with decentralized execution (CTDE) framework is adopted. Different from the traditional MADRL approach such as MADDPG which directly incorporates the observation-action pair across multiple agents to train their critics, the MA-GAC learns a joint critic by sharing the parameters among agents with the help of the attention mechanism [31]. The attention mechanism enables the critic network to pay attention to the relevant information and can also avoid useless information from the inactivated agents in the dynamic VFC system by adding a mask tensor. Furthermore, the time-variant characteristics of the local computational resources trading history reflect the implicit intention of the agents in the same coalition. To this effect, the GRU network is employed in the MA-GAC to extract the features containing future trends. In this way, the proposed MA-GAC turns out to be more robust and efficient to tackle the challenges of the dynamic task offloading problem in the stochastic VFC system compared with other state-of-the-art MADRL methods.

4.2.1 MA-GAC Structure

In this part, the MA-GAC algorithm framework is introduced based on SAC design [32]. The SAC is an extension of the actor-critic DRL approaches which contain the critic network and actor network in the algorithm structure. The parameterized critic network $Q^{\theta}(O_t, A_t)$ takes state-action pairs as input and outputs a scalar evaluating the Q-value function; meanwhile, the parameterized actor network $\pi^{\psi}(a_{n,t}|o_{n,t})$ takes the local observation as inputs and outputs the action probability distribution at $o_{n,t}$[4].

The optimization objective in SAC is not only to maximize the long-term reward but also to maximize the entropy of policies at the same time. As such, the SAC methods tend to select optional actions with a similar estimated Q-value faced with the same states. Therefore, the SAC method is more generalized and robust than the traditional deterministic DRL approach and avoids premature converging to non-optimal policies.

To begin with, we discuss the update rule of the critic network Q^{θ}. The critic network evaluates the policies produced by the actor network by estimating the Q-value. The accuracy of the estimation is optimized by minimizing the mean squared temporal difference (TD) error as given:

$$L(\theta) = \mathop{\mathbb{E}}_{(O_t, A_t, Re_t, \tilde{O}_t)\sim\mathcal{D}} \left[(Q^{\theta}(O, A) - y)^2 \right] \tag{30}$$

where $\mathcal{D}$ is the replay buffer storing the experiences, $\tilde{O}_t$ is the changed state at time slot t from O_t, and y is the next step estimated Q-value. Combined with an entropy term, the next step estimated value y is given by:

$$y = Re_t + \gamma \mathop{\mathbb{E}}_{\tilde{A}_t \sim \pi^{\bar{\psi}}} \left[Q^{\bar{\theta}}(\tilde{O}_t, \tilde{A}_t) - \alpha \log(\pi^{\bar{\psi}}(\tilde{A}_t|\tilde{O}_t)) \right] \tag{31}$$

where α is the temperature parameter weighing the importance of policy entropy, $\tilde{A}_t$ is the action taken by $\pi^{\bar{\psi}}$ given state $\tilde{O}_t$, $Q^{\bar{\theta}}$ is the target critic network, and $\pi^{\bar{\psi}}$ is the target policy network. In SAC, the critic and actor networks are generated as the current network to explore the environment and target network to avoid overestimating the Q-value. The target network is soft updated by the current network during training.

The policy parameters ψ are updated by maximizing the performance function evaluated by the critic network. Taking the performance gradient $\nabla_{\psi} J(\pi^{\psi})$ as:

$$\begin{aligned} J(\pi^{\psi}) = \mathop{\mathbb{E}}_{(O_t, A_t)\sim\mathcal{D}} & \left[\alpha \log(\pi^{\psi}(A'_t|O)) \right. \\ & \left. - Q^{\theta}(O_t, A'_t) + Q^{\theta}(O_t, A_t) \right] \end{aligned} \tag{32}$$

$$\nabla_{\psi} J(\pi^{\psi}) = \nabla_{\psi} \alpha \log(\pi^{\psi}(A'_t|O)) - \nabla_{A'_t} Q^{\theta}(O_t, A'_t) \tag{33}$$

where A_t is the action taken in the replay buffer and A'_t is the estimated action with $a_{i,t}$ taken by π^{ψ} while the actions of other agents $a_{-i,t}$ being viewed as fixed. Note that the item $-Q^{\theta}(O_t, A'_t) + Q^{\theta}(O_t, A_t)$ is utilized to signify the multi-agent advantages, where a similar design can be seen in [26]. This item compares the value of a specific action $a_{i,t}$ and treats all other agents' actions as fixed. It therefore indicates whether the current action of agent V_i can cause an increase in the expected return.

4.2.2 Attention Mechanism in Critic Network

The dynamics in the stochastic VFC environment pose severe challenges to the system convergence with multiple learning agents. To promote the robustness of the learning process among an uncertain number of agents, the attention mechanism is employed to the critic network Q^{θ}. As shown in Fig. 4, the attention mechanism takes the action and observation of all the agents as embeddings and trains three matrices (value matrix W_v, query matrix W_q, and key matrix W_k) as the MLP layers. The objective of the attention mechanism is to obtain the weighted attention contribution $att_{i,j}$ of agent V_i from other agents as:

$$att_i = \sum_{j \neq i} \mu_{i,j} h(W_v e_j) \tag{34}$$

[4] The variables θ and ψ are used to denote the network parameters of Q^{θ} and π^{ψ}, respectively.

Algorithm 3 Training of the MA-GAC Approach

```
1:  Initialize the environment with $N$ agents.
2:  for each training episode do
3:      Reset environment.
4:      for time step $t = 0 : T$ do
5:          Get observation $O_t$ from the environment.
6:          Select an action from $a_{i,t} \sim \pi^{\theta}(o_{i,t})$.
7:          Each agent execute the action and get the changed observation $\tilde{o}_{i,t}$ as well as the reward $Re_{i,t}$.
8:          Store the experiences $(O_t, A_t, R_t, \tilde{O}_t)$ into $\mathcal{D}$.
9:          if start to train then
10:             Retreive a minibatch of experiences from $\mathcal{D}$.
11:             Evaluate each agent values with the networks $Q^{\theta_{1,2}}(O_t, \pi^{\psi}(O_t))$, $Q^{\theta_{1,2}}(O_t, A_t)$, $Q_i^{\bar{\theta}_{1,2}}(\tilde{O}_t, \tilde{A}_t)$, $\tilde{a}_{i,t} \sim \pi_i^{\bar{\psi}}(\tilde{o}_{i,t})$.
12:             Update current critic network with $\nabla_{\theta} L(\theta)$ in (30).
13:             Update current actor network with $\nabla_{\psi} J(\psi)$ in (32).
14:             Soft update target network with $\bar{\theta} \leftarrow \tau\bar{\theta} + (1-\tau)\theta$, $\bar{\psi} \leftarrow \tau\bar{\psi} + (1-\tau)\psi$.
15:         end if
16:     end for
17: end for
```

where $\mu_{i,j}$ is the attention weights given by:

$$\mu_{i,j} = \frac{\exp^{(W_k e_j)^T W_q e_i}}{\sum_j \exp^{(W_k e_j)^T W_q e_i}} \tag{35}$$

and $h(\cdot)$ is a non-linear activation function in the network. The derivation of attention value for each agent is illustrated in three steps. Firstly, the value, query, and key of all the agents are extracted as tensors by taking the embeddings as input for the three MLP layers W_v, W_q, and W_k, respectively. After that, each agent V_i requests for the key tensor from other agents and multiplies the transposed key tensor by its personal query tensor. The attention weights $\mu_{i,j}$ are thereby calculated. Finally, each agent uses the attention weights to achieve the attention contribution via (34). The trainable parameterized matrices W_v, W_q, and W_k are shared among all the agents and embedded in the critic network. Therefore, those three networks can be updated by minimizing the $L(\theta)$ in (30) via backward propagation.

4.2.3 *GRU in Actor Network*

The implicit intention of surrounding vehicles can be inferred from the local history information including the unit buy price, unit sell price, local demanding resources, and local supply resources. GRU is thereby merged into the actor network in the MA-GAC method as the hidden layer to help make decisions more wisely. GRU is chosen in our approach because it performs well in sequence analysis and enjoys a cheap computational cost.

The training process of the proposed MA-GAC approach is shown in Algorithm 3. Specifically, to avoid the overestimated Q-value during training, we apply a double critic network in the proposed MA-GAC approach as shown in Line 11. Q^{θ_1} and Q^{θ_2} are trained separately and use the minimum of those network outputs as the certain Q-value.

TABLE 1
Simulation Parameters

RSU cell diameter	500 m
RSU number $\lvert\mathcal{R}\rvert$	2
RSU processing rate r_{R_m,R_n}	5 MB/s
Cloud processing rate $r_{R,cloud}$	3 MB/s
Agent number $\lvert\mathcal{V}\rvert$	100
Maximum service zone capacity $N_{\max}$	100
Maximum vehicular velocity $v_{\max}$	20 m/s
Vehicular one-hop distance D_0	100 m
Vehicular accelerating distance D_{d1}	100 m
Vehicular braking distance D_{d2}	50 m
Vehicular acceleration acc_0	1.5 m/s^{-2}
Vehicular CPU frequency f	$[2, 6]$ GHz
Vehicular calculating condition ζ	$[3, 5]$ GHz
V2V transmission power $P_{i,j}^t$	23 dBm
Static transmission power $P_{i,R}^t$	26 dBm
Noise power N_0	-114 dBm
Delay-sensitive factor ε	$[0, 0.1]$
Unit basic utility u_i	$[0.5, 1]$ $(\text{GHz} \cdot \text{s})^{-1}$
Task deadline δ_i	$[0.1, 0.4]$ s
Task download data size dw_i	$[0.02, 0.2]$ MB
Task upload data size up_i	$[0.05, 0.1]dw_i$ MB
Task computational requirement cr_i	$[0.75, 1.25]\zeta_i\delta_i$ G-cycles
Retailer buy unit price p_R^b	$[0.029, 0.053]$ GHz^{-1}
Retailer sell unit price p_R^s	$[0.008, 0.023]$ GHz^{-1}
Scaling factor ω	0.5
Time slot length Δt	0.5 s
Trading time T^{trade}	1.5 s
Trading intention l_i	$[0.5, 3]$ GHz
Total bandwidth $B_{i,\cdot}$	10 MHz
Total simulation time T	1.0×10^3 s

5 Simulation and Results

In this section, various experiments are executed to verify the superiority of the proposed distributed many-to-many task offloading framework. Firstly, simulation setup and implementation are introduced as preliminary. Then, numerous evaluation metrics are applied to evaluate the system's performance. Finally, we analyze the features of the proposed V2V trading-based offloading framework.

5.1 Simulation Setup and Implementation

We construct the system environment via PYTHON and assume a bi-direction two-lane road in each service zone with the RSU located at $(500, 0)$. The coverage of each RSU is with a cell diameter of 500 m and we adopt 2 service zones in the hierarchical VFC environment, discussing the multi-tier offloading strategy in the whole system. In each service zone, vehicles drive in with an initial velocity. The entry of vehicles follows the Poisson distribution with the Poisson parameter equal to 1 with $\alpha_v = 3$ in (1). As such, vehicles run on the road and the tasks are generated at each time slot. To execute the simulations in specific conditions with a fixed number of vehicles, the maximum capacity of each service zone is fixed as different numbers (e.g., 30, 50, 100). If the number of vehicles in the service zone exceeds the predefined maximum capability, the following vehicles can not enter the service zone.

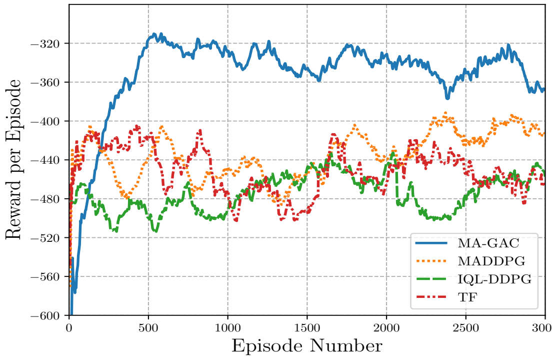


Fig. 5. The overall reward in each episode of all the approaches.

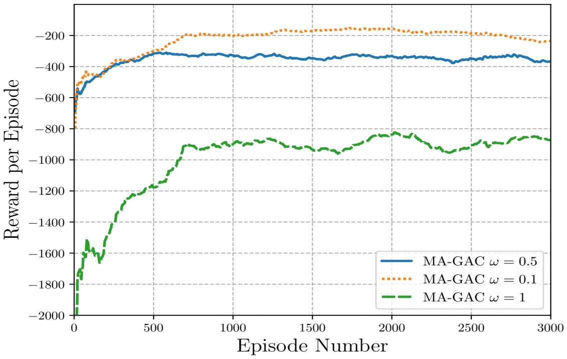


Fig. 6. The overall reward in each episode with different ω.

Next, we discuss the parameters of the trading mechanism. The observable pricing history length of each vehicle is set as $\iota = 10$ and a bounded amount of trading intention $l_i \in [0.5, 3]$ GHz is adopted to standardize the trading behaviors of each vehicle. Moreover, the scaling factor in (24) is set to $\omega = 0.5$ as default. As for the determined retailer sell/buy unit price, a pricing design of the RSU is given referring to the one in [33] to set the sell/buy price properly and the prices are directly influenced by the network demand and supply:

$$p_R^s = \lambda_1 (E^s)^2 + \lambda_2 E^s, \qquad p_R^b = \lambda_1 (E^b)^2 + \lambda_2 E^b$$

where E^s and E^b are the quantified desires towards available resources and tasks in the global network, and λ_1 and λ_2 are the pricing parameter. For simplicity, a simple sinusoidal is utilized to simulate the pricing fluctuation within the range of $E_t^b \in [0.8, 1.2], E_t^s \in [0.3, 0.7]$, and the pricing parameters are set as $\lambda_1 = \lambda_2 = 0.2$.

For analysis, four alternative methods are selected as benchmarks for comparison.

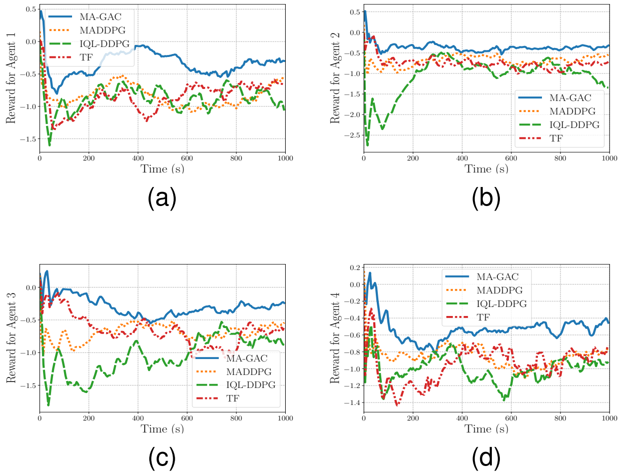
(1) *MADDPG*: The multi-agent deep deterministic policy gradient method [34] is a typical MADRL approach with deterministic policy. To be fair, we apply this approach with the same learning parameters and settings.

(2) *Individual Q-Learning*: The individual Q-learning (IQL) tends to treat the learning task as a single-agent process and views the interactions of other agents as part of the environment. We apply the deep deterministic policy gradient (DDPG) method as the IQL learning model, namely IQL-DDPG for comparison.

(3) *Trading Forward*: The trading forward (TF) method denotes the circumstance in which the agents trade resources instantly regardless of the price. Each agent only concerns with its computational requirements or the surplus resources at the current time slot.

(4) *Centralized Offloading*: A PSO-based heuristic approach is proposed to solve the centralized offloading, termed Cen-PSO. In the Cen-PSO method, the RSU offloads tasks regardless of individual intentions of vehicles to maximize the sum of task utility at each time slot.

The implementation of the MA-GAC network is listed as follows. We set the episode length as 50, training episode number as 3000, soft updating parameter $\tau = 0.01$, learning rate for actors as $lr_a = 1 \times 10^{-4}$, learning rate for the critic as $lr_c = 5 \times 10^{-4}$, discount factor $\gamma = 0.97$, batch size = 256, and memory size = 1×10^5. We use leakyReLUs as activation units and AdamW as the optimizer. The detailed network structure has been demonstrated in Fig. 4 and Table 1 shows other relevant parameters.



(a)

(b)

(c)

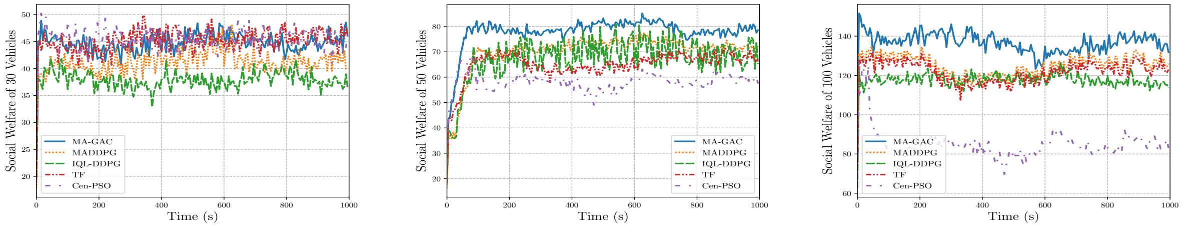
(d)

Fig. 7. Personal reward gained by arbitrary four agents during 1.0×10^3 seconds. (a)~(d) Agent 1~Agent 4.

5.2 Performance Evaluation

Fig. 5 shows the convergence of the overall reward compared with four trading-based methods. It is observed that the deterministic DRL approaches MADDPG and IQL-DDPG are not suitable in the stochastic V2V task offloading environment and thereby converge slowly. Specifically, the IQL-DDPG method deems the other agents as part of the environment, which causes instability in learning the environment and performs worse than the multi-agent approaches. In detail, the proposed MA-GAC method converges after 500 training episodes and achieves about −320 overall reward. Note that according to (29), the unit basic utility u is deducted from the gained utility and external cost, which makes the reward function a negative value. For comparison, the MADDPG method converges in about 2300 training episodes with an overall reward of −400 and the IQL-DDPG method falls into the premature optimal point with a −440 reward. Since the Cen-PSO method does not concern the individual intention of vehicles, the objective function is not

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3250495



(a) (b) (c)

Fig. 8. Social welfare performance comparison during 1.0×10^3 seconds simulation given different numbers of vehicles. (a) 30 vehicles. (b) 50 vehicles. (c) 100 vehicles.

suitable to evaluate the system performance of the Cen-PSO method, so we only compare the performance of the MA-GAC, MADDPG, IQL-DDPG, and TF methods in Fig. 5.

Then, we compare the performance of the proposed MA-GAC under different scaling factors in Fig. 6. The factor ω is set to 0.1, 0.5, and 1 respectively to put different weights on the vehicles' intention between task utility and external costs. When ω is set to be small, the vehicles are uninterested in the gain of monetary revenues and thereby tend to possess computational resources instead of sharing them for the sake of local task utility. Meanwhile, when ω is large, the vehicles despise their QoS of tasks and decide to sell the computational resources for more revenue. Statistics in Fig. 6 display that the proposed MA-GAC enables one to learn suitable policy under different conditions with the changing ω.

Furthermore, the personally gained reward comparison among the MA-GAC, MADDPG, IQL-DDPG, and TF methods are presented in Fig. 7 with the same random seeds during simulation. Four agents are chosen randomly from the vehicle set and we compare the long-term average reward trends lasting for 1.0×10^3 seconds. Among the four methods, the proposed MA-GAC performs the best in terms of reward and the IQL-DDPG performs the worst. The TF method and the MADDPG method have similar performance. As a result, since the agents are prompted to choose the trading-based approach with higher revenues, they are motivated to use the coordinated MA-GAC method compared with other methods. Therefore, the MA-GAC method is verified to benefit the best for not only the overall system but also any single agent in terms of computational resource trading.

After that, in order to evaluate the performance properly, we introduce an important metric to reflect the overall system performance as *social welfare*. The social welfare is given by:

$$\begin{aligned} SW &= \sum_{V_i \in \mathcal{V}} obj_i \\ &= \sum_{R_m \in \mathcal{R}} \sum_{V_i \in \mathcal{V}} U_i + ext(V_i, R_m). \end{aligned} \tag{36}$$

Since the external costs of vehicles are offset by summing the trading objectives up, the system social welfare mainly reflects the computing capability of the local service zone and the QoS of the tasks. In Fig. 8, we compare the social welfare performance among the five methods after the continuous and dynamic traffic environment runs for 1.0×10^3 seconds. To verify the robustness and scalability of the distributed offloading framework, 30, 50, and 100 vehicles are tested in the environment as shown in Fig. 8a, Fig. 8b, and Fig. 8c, respectively. When the number of vehicles is larger than 50 in each service zone, the distributed trading-based methods exhibit superiority in the metric of social welfare compared with the centralized Cen-PSO method. It is mainly because that the centralized heuristic method cannot tackle the large-scale NP-hard assignment problem and sinks into the premature local optimum early. Nonetheless, when the number of vehicles is 30 in the service zone, the PSO-based centralized offloading method achieves ap-

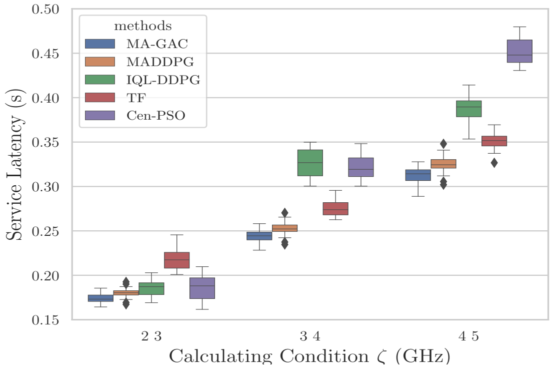


Fig. 9. The box plot of the average service latency under different calculating conditions.

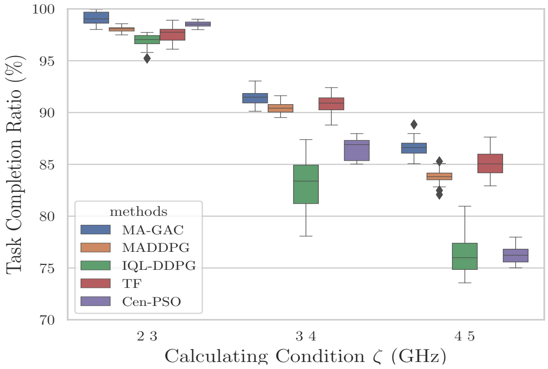


Fig. 10. The box plot of the task completion ratio under different calculating conditions.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2023.3250495

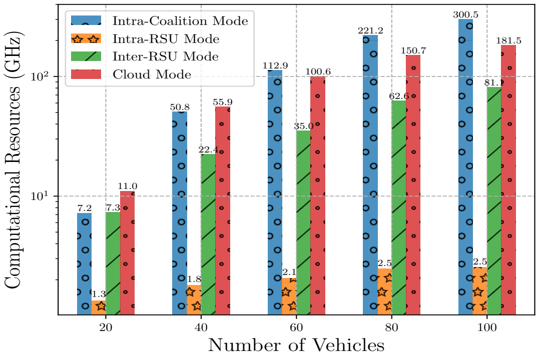


Fig. 11. The exchanged computational resources of the four offloading modes versus different numbers of vehicles in the fog layer.

proximate social welfare compared with the MA-GAC and TF methods. The TF method always exhibits medium social welfare with small fluctuation due to its stability as the static strategy. As for the learning-based methods (i.e., MADDPG, IQL-DDPG, and MA-GAC), the proposed MA-GAC method shows better scalability than the MADDPG and IQL-DDPG methods versus various numbers of vehicles thanks to the attention mechanism. Moreover, the proposed MA-GAC method performs almost the best in social welfare with a varying number of vehicles.

Finally, to evaluate the performance thoroughly, general metrics including average service latency and task completion ratio are also considered. Task completion ratio is defined as the percent of the tasks completed within their deadlines. Box plots are utilized to reflect the stability of the performance explicitly. As shown in Fig. 9 and Fig. 10, we modify the range of calculating conditions to simulate the cases where vehicles are running on the lower load, the medium load, and the higher load. It can be inferred from Fig. 9 that the proposed MA-GAC method achieves the shortest service latency with rather dependable stability in variance. The MADDPG method wins the second-best performance in service latency followed by the TF method. The IQL-DDPG method is of great variance in performance due to its learning strategy for individual agents and the Cen-PSO method also fails to address the complicated offloading relationship when the calculating condition increases. A similar conclusion can be deduced from Fig. 10, in which the MA-GAC method accomplishes the tasks at a prominent 99% completion ratio with the calculating burden $\zeta \in [2, 3]$ and 87% completion ratio when the calculating condition falls in the range of $\zeta \in [4, 5]$, which is distinctly higher than other methods. What's more, by evaluating the performance comparison of the MADDPG method and TF method in Fig. 8c, Fig. 9, Fig. 10, one can find that the TF method succeeds in optimizing the service latency and task completion ratio, but fails in gaining social welfare, which induces the result that the learning-based method better utilizes the resources in the local service zones.

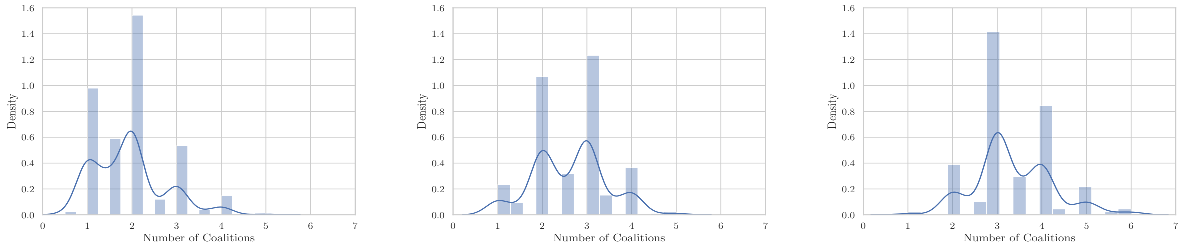
5.3 Analyses on the V2V Trading-Based Framework

In this part, we pose analyses on the multi-tier offloading modes and the coalition formation process of the proposed V2V trading-based framework. We analyze the multi-tier offloading process of the proposed framework. Fig. 11 illustrates the exchanged computational resources of the four offloading modes (i.e., intra-coalition, intra-RSU, inter-RSU, and cloud) given a varying number of vehicles at each time slot. It can be seen that when the number of vehicles is small (e.g., 20 and 40), the computational resources in the cloud layer are more frequently used by the vehicles than trading resources with each other because the resources at the edge are not sufficient. With the increasing number of vehicles, intra-coalition resources are more efficiently utilized than the other three modes, indicating that V2V offloading happens more often in the fog layer and computational resources in the service zone are further exploited.

Additionally, we compare the number of coalitions in each service zone given different numbers of vehicles at each time slot. As shown in Fig. 12, histograms are depicted and the Gaussian kernel density estimation method is applied to find the probability density function of the coalition number given 30, 50, and 100 vehicles in each service zone. The coalition number is most likely to be between 1 and 2 given 30 vehicles in Fig. 12a, between 2 and 3 given 50 vehicles in Fig. 12b, and between 3 and 4 given 100 vehicles in Fig. 12c. The coalition number distribution restricted by the feasibility constraints in (19) verifies Theorem 1 and indicates that in most cases the vehicles tend to form a grand coalition for cooperation.

6 Conclusion

In this paper, we formulated the many-to-many task offloading problem among multiple vehicles in the hierarchical VFC system with the consideration of each vehicle's intention. We proposed a distributed multi-tier task offloading framework based on the V2V computational resource trading paradigm. In the proposed framework, the coalitional



(a) (b) (c)

Fig. 12. Distributions of the coalition numbers given different numbers of vehicles. (a) 30 vehicles. (b) 50 vehicles. (c) 100 vehicles.

game and the MMR pricing mechanism were applied to allow vehicles to trade the computational resources cooperatively and the multi-tier task offloading was executed based on the trading consensus. Then, to learn the strategic trading for each participant efficiently, we designed a multi-agent SAC approach with the attention mechanism and GRU, termed MA-GAC to learn the environment in a coordinated manner. Numerical results verified that the proposed MA-GAC approach outperforms other benchmarks in a dynamic environment and further exploits the system computing capability.

7 Acknowledgments

This work is supported in part by the National Key R&D Program of China (Grant No. 2021ZD0112700), in part by the National Natural Science Foundation of China under Grant 62271351, 62125101, and 62201390, in part by the Natural Science Foundation of Shanghai under Grant 20ZR1462400 and 22ZR1463600, in part by Chenguang Program supported by Shanghai Education Development Foundation and Shanghai Municipal Education Commission, in part by the Shanghai Municipal Science and Technology Major Project (2021SHZDZX0100) and the Fundamental Research Funds for the Central Universities.

References

[1] X. Cheng, R. Zhang, and L. Yang, “Wireless Toward the Era of Intelligent Vehicles,” *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 188-202, Feb. 2019.

[2] K. -C. Chen, T. Zhang, R. D. Gitlin, and G. Fettweis, “Ultra-Low Latency Mobile Networking,” *IEEE Network*, vol. 33, no. 2, pp. 181-187, Mar./Apr. 2019.

[3] X. Liu, H. Zhang, K. Long, A. Nallanathan, and V. C. M. Leung, “Energy Efficient User Association, Resource Allocation and Caching Deployment in Fog Radio Access Networks,” *IEEE Transactions on Vehicular Technology*, vol. 71, no. 2, pp. 1846-1856, Feb. 2022.

[4] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, “Mobile Edge Computing: A Survey,” *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450-465, Feb. 2018.

[5] K. Wang, J. Li, Y. Yang, W. Chen, and L. Hanzo, “Content-Centric Heterogeneous Fog Networks Relying on Energy Efficiency Optimization,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 13579-13592, Nov. 2020.

[6] H. Sami, A. Mourad and W. El-Hajj, “Vehicular-OBUs-As-On-Demand-Fogs: Resource and Context Aware Deployment of Containerized Micro-Services,” in *Proc. IEEE/ACM Transactions on Networking*, vol. 28, no. 2, pp. 778-790, Apr. 2020.

[7] A. Waheed *et al.*, “A Comprehensive Review of Computing Paradigms, Enabling Computation Offloading and Task Execution in Vehicular Networks,” *IEEE Access*, vol. 10, pp. 3580-3600, 2022.

[8] H. Wang *et al.*, “Architectural Design Alternatives Based on Cloud/Edge/Fog Computing for Connected Vehicles,” *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2349-2377, Fourthquarter 2020.

[9] J. Zhang, H. Guo, J. Liu, and Y. Zhang, “Task Offloading in Vehicular Edge Computing Networks: A Load-Balancing Solution,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 2092-2104, Feb. 2020.

[10] J. Wei, X. Wang, N. Li, G. Yang, and Y. Mu, “A Privacy-Preserving Fog Computing Framework for Vehicular Crowdsensing Networks,” *IEEE Access*, vol. 6, pp. 43776-43784, 2018.

[11] K. Wang, J. Li, Y. Yang, W. Chen, and L. Hanzo, “Content-Centric Heterogeneous Fog Networks Relying on Energy Efficiency Optimization,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 13579-13592, Nov. 2020.

[12] K. Zhang, M. Peng, and Y. Sun, “Delay-Optimized Resource Allocation in Fog-Based Vehicular Networks,” *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1347-1357, 1 Feb.1, 2021.

[13] R. Zhang, R. Lu, X. Cheng, N. Wang, and L. Yang, “A UAV-Enabled Data Dissemination Protocol with Proactive Caching and File Sharing in V2X Networks,” *IEEE Transactions on Communications*, vol. 69, no. 6, pp. 3930-3942, Jun. 2021.

[14] K. Wang, Y. Tan, Z. Shao, S. Ci, and Y. Yang, “Learning-Based Task Offloading for Delay-Sensitive Applications in Dynamic Fog Networks,” *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11399-11403, Nov. 2019.

[15] C. Lin, G. Han, X. Qi, M. Guizani, and L. Shu, “A Distributed Mobile Fog Computing Scheme for Mobile Delay-Sensitive Applications in SDN-Enabled Vehicular Networks,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5481-5493, May. 2020.

[16] C. Zhu *et al.*, “Folo: Latency and Quality Optimized Task Allocation in Vehicular Fog Computing,” *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4150-4161, Jun. 2019.

[17] M. S. Bute, P. Fan, L. Zhang, and F. Abbas, “An Efficient Distributed Task Offloading Scheme for Vehicular Edge Computing Networks,” *IEEE Transactions on Vehicular Technology*, vol. 70, no. 12, pp. 13149-13161, Dec. 2021.

[18] Z. Wei, B. Li, R. Zhang, X. Cheng, and L. Yang, “OCVC: An Overlapping-Enabled Cooperative Vehicular Fog Computing Protocol,” *IEEE Transactions on Mobile Computing*, 2022.

[19] Z. Zhou, H. Liao, X. Zhao, B. Ai, and M. Guizani, “Reliable Task Offloading for Vehicular Fog Computing Under Information Asymmetry and Information Uncertainty,” *IEEE Transactions on Vehicular Technology*, vol. 68, no. 9, pp. 8322-8335, Sep. 2019.

[20] O. Nazih, N. Benamar, and A. Addaim, “An Incentive Mechanism for Computing Resource Allocation in Vehicular Fog Computing Environment,” in *Proc. International Conference on Innovation and Intelligence for Informatics, Computing and Technologies (3ICT)*, Sakheer, Bahrain, 2020, pp. 1-5.

[21] J. Shi, J. Du, J. Wang, J. Wang, and J. Yuan, “Priority-Aware Task Offloading in Vehicular Fog Computing Based on Deep Reinforcement Learning,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 16067-16081, Dec. 2020.

[22] X. Peng, K. Ota, and M. Dong, “Multiattribute-Based Double Auction Toward Resource Allocation in Vehicular Fog Computing,” *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3094-3103, Apr. 2020.

[23] Z. Wei, B. Li, R. Zhang, and X. Cheng, “Contract-Based Charging Protocol for Electric Vehicles with Vehicular Fog Computing: An Integrated Charging and Computing Perspective,” *IEEE Internet of Things Journal*, 2022.

[24] W. Tushar *et al.*, “Peer-to-Peer Energy Trading With Sustainable User Participation: A Game Theoretic Approach,” *IEEE Access*, vol. 6, pp. 62932-62943, 2018.

[25] J. Li, Y. Ye, D. Papadaskalopoulos, and G. Strbac, “Computationally Efficient Pricing and Benefit Distribution Mechanisms for Incentivizing Stable Peer-to-Peer Energy Trading,” *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 734-749, 15 Jan.15, 2021.

[26] Y. Ye, Y. Tang, H. Wang, X. -P. Zhang, and G. Strbac, “A Scalable Privacy-Preserving Multi-Agent Deep Reinforcement Learning Approach for Large-Scale Peer-to-Peer Transactive Energy Trading,” *IEEE Transactions on Smart Grid*, vol. 12, no. 6, pp. 5185-5200, Nov. 2021.

[27] Z. Wei, B. Li, R. Zhang, X. Cheng, and L. Yang, “Dynamic Many-to-Many Task Offloading in Vehicular Fog Computing: A Multi-Agent DRL Approach,” in *Proc. GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, Rio de Janeiro, Brazil, 2022, pp. 6301-6306.

[28] X. Hou, Y. Li, D. Jin, D. O. Wu, and S. Chen, “Modeling the Impact of Mobility on the Connectivity of Vehicular Networks in Large-Scale Urban Environments,” *IEEE Transactions on Vehicular Technology*, vol. 65, no. 4, pp. 2753-2758, Apr. 2016.

[29] W. Zhang, Z. Zhang, and H. Chao, “Cooperative Fog Computing for Dealing with Big Data in the Internet of Vehicles: Architecture and Hierarchical Resource Management,” *IEEE Communications Magazine*, vol. 55, no. 12, pp. 60-67, Dec. 2017.

[30] J. Li *et al.*, “Service Migration in Fog Computing Enabled Cellular Networks to Support Real-Time Vehicular Communications,” *IEEE Access*, vol. 7, pp. 13704-13714, 2019.

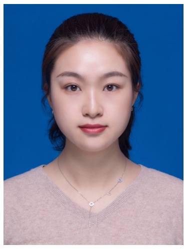
[31] J. Wu, X. Cheng, X. Ma, W. Li, and Y. Zhou, “A Time-Efficient and Attention-Aware Deployment Strategy for UAV Networks Driven by Deep Reinforcement Learning,” in *Proc. IEEE 94th Vehicular Technology Conference (VTC2021-Fall)*, 2021, pp. 01-05.

[32] T. Haarnoja *et al.*, “Soft Actor-Critic Algorithms and Applications,” 2018, arXiv:1812.05905.

[33] C. P. Mediwaththe and D. B. Smith, “Game-Theoretic Electric Vehicle Charging Management Resilient to Non-Ideal User Behavior,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 11, pp. 3486-3495, Nov. 2018.

[34] J. Tian, Q. Liu, H. Zhang, and D. Wu, “Multiagent Deep-Reinforcement-Learning-Based Resource Allocation for Heterogeneous QoS Guarantees for Vehicular Networks,” *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 1683-1695, 1 Feb.1, 2022.

Zhiwei Wei (Student Member, IEEE) received the B.E. degree from University of Shanghai for Science and Technology in 2020. He is currently pursuing the master’s degree in software engineering with the Tongji University. His current research interests include vehicular fog computing and computational resource allocation.



Bing Li (Member, IEEE) received the Ph.D. degree from Tongji University, Shanghai, China, in 2021. She is currently an Assistant Professor with Tongji University, Shanghai, China. Her current research interests include UAV communications, wireless resource allocation, and relay communications.

Rongqing Zhang (Member, IEEE) received the B.S. and Ph.D. degrees (with honors) from Peking University, Beijing, China, in 2009 and 2014, respectively. From 2014 to 2018, he worked as a postdoctoral research fellow at Colorado State University, CO, USA. Since 2019, he has been an Associate Professor at Tongji University, Shanghai, China. He has authored and co-authored two books, two book chapters, and over 100 papers in refereed journals and conference proceedings. His current research interests include vehicular communications and networking, UAV communications, and autonomous driving.

Dr. Zhang was the recipient of the Academic Award for Excellent Doctoral Students, Ministry of Education of China, the co-recipient of the First-Class Natural Science Award, Ministry of Education of China, and received the Best Paper Awards at IEEE ITST’12, ICC’16, GLOBECOM’18, and ICC’19. He was also awarded as International Presidential Fellow of Colorado State University in 2017. Currently, he is serving as an Associate Editor of *IEEE Transactions on Vehicular Technology* and *IET Communications*.

Xiang Cheng (Fellow, IEEE) received the Ph.D. degree jointly from Heriot-Watt University and the University of Edinburgh, Edinburgh, U.K., in 2009. He is currently a Boya Distinguished Professor of Peking University. His general research interests are in areas of channel modeling, wireless communications, and data analytics, subject on which he has published more than 280 journal and conference papers, 9 books, and holds 17 patents. Prof. Cheng is a Distinguished Young Investigator of China Frontiers of Engineering, a recipient of the IEEE Asia Pacific Outstanding Young Researcher Award in 2015, a Distinguished Lecturer of *IEEE Vehicular Technology Society*, and a Highly Cited Chinese Researcher in 2020. He was a co-recipient of the 2016 IEEE JSAC Best Paper Award: Leonard G. Abraham Prize, and IET Communications Best Paper Award: Premium Award. He has also received the Best Paper Awards at IEEE ITST’12, ICCC’13, ITSC’14, ICC’16, ICNC’17, GLOBECOM’18, ICCS’18, and ICC’19. He has served as the symposium lead chair, co-chair, and member of the Technical Program Committee for several international conferences. He is currently a Subject Editor of *IET Communications* and an Associate Editor of the *IEEE Transactions on Wireless Communications*, *IEEE Transactions on Intelligent Transportation Systems*, *IEEE Wireless Communications Letters*, and the *Journal of Communications and Information Networks*. In 2021, he was selected into two world scientist lists, including World’s Top 2% Scientists released by Stanford University and Top Computer Science Scientists released by Guide2Research.

Liuqing Yang (Fellow, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Minnesota, Minneapolis, MN, USA, in 2004. She is currently a Professor with the Hong Kong University of Science and Technology (Guangzhou). Before joining Hong Kong University of Science and Technology (Guangzhou), she has been a Faculty Member with the Department of Electrical and Computer Engineering, University of Florida (2004-2010), Colorado State University (2010-2020), and University of Minnesota (2020- 2021). Her research interests include communications and networking subjects on which she has published more than 370 journals and conference papers, four book chapters, and five books. She was a recipient of the ONR Young Investigator Program (YIP) Award in 2007, the NSF Faculty Early Career Development (CAREER) Award in 2009, and the Best Paper Award at IEEE ICUWB 2006, ICCC 2013, ITSC 2014, GLOBECOM 2014, ICC 2016, WCSP 2016, GLOBECOM 2018, ICCS 2018, and ICC 2019. She is the Editor-in-Chief of *IET Communications*, a Executive Editorial Committee (EEC) Member of the *IEEE Transactions on Wireless Communications*, and a Senior Editor of the *IEEE Transactions on Signal Processing*. She has also served as an Editor for the *IEEE Transactions on Communications*, the *IEEE Transactions on Intelligent Transportation Systems*, *IEEE Intelligent Systems*, and PHYCOM: *Physical Communication*, and as a program chair, a track/symposium, or a TPC chair for many conferences.