Multi-hop Task Routing in Vehicle-assisted Collaborative Edge Computing

Yiqin Deng, Member, IEEE, Haixia Zhang, Senior Member, IEEE, Xianhao Chen, Member, IEEE, and Yuguang Fang, Fellow, IEEE

Abstract—Collaborative edge computing has emerged as a novel paradigm that allows edge servers (ESs) to share data and computing resources, effectively mitigating network congestion in traditional multi-access edge computing (MEC) scenarios. However, existing research in collaborative edge computing often limits offloading to only one hop, which may lead to suboptimal computing resource sharing due to challenges such as poor channel conditions or high computing workload at ESs located just one hop away. To address this limitation and enable more efficient computing resource utilization, we propose a multi-hop MEC approach that leverages omnipresent vehicles in urban areas to create a data transportation network for task delivery. Here, we propose a general multi-hop task offloading framework for vehicle-assisted collaborative edge computing where tasks from users can be offloaded to powerful ESs via potentially multi-hop transmissions. Under the proposed framework, we formulate an aggregated service throughput maximization problem by designing the task routing path subject to end-to-end latency requirements, spectrum, and computing resources. To efficiently address the curse of dimensionality problem due to vehicular mobility and channel variability, we develop a deep reinforcement learning, i.e., multi-agent deep deterministic policy gradient, based multi-hop task routing approach. Numerical results demonstrate that the proposed algorithm outperforms existing benchmark schemes.

Index Terms—Collaborative edge computing, Vehicular networks, Computation offloading, Multi-hop routing, Deep reinforcement learning (DRL).

I. INTRODUCTION

Multi-access edge computing (MEC) has been identified as a promising architecture for computing services that aims to provide real-time or low latency services to end-users located in close proximity [1], [2]. One of the primary techniques utilized in MEC is computation offloading, which enables computing tasks to be processed locally or offloaded to an edge server (ES) based on the availability of local computing resources and transmission conditions [3]. This approach proves particularly beneficial for resource-constrained mobile devices (MDs) with limited computing capability, storage, and battery power [4], [5]. Moreover, through effective computation offloading in MEC, end-to-end (e2e) latency for emerging capability-demanding or latency-sensitive applications can be drastically reduced, ultimately providing high quality-of-service to end users [6], [7].

Extensive research efforts have been devoted to optimizing resource utilization and efficiency in computation offloading for MEC [8]–[13]. Previous research efforts have primarily focused on computation offloading and resource optimization involving direct associations between users and ESs within a user’s communication range (i.e., one-hop away ESs), considering computing resources and/or communication resource optimization at a single ES [8]–[10]. However, this approach may fall short in practice due to a lack of coordination among ESs, hindering effective load balancing [14]. Despite research efforts attempting to enable resource-constrained ESs to collaborate in processing computation-intensive tasks for workload balancing, the current literature is primarily concerned with one-hop offloading between MDs and ESs [11]–[13]. Such an approach incurs an implicit assumption, whereby communication resources available at the ESs one-hop away are sufficient for uploading complex computing tasks. This suppositional approach may not always be effective in reality. Specifically, the single-hop offloading approaches may not work well under resource-constrained scenarios. For instance, in scenarios like MEC-enabled surveillance video analytics for public safety applications in smart cities [15], where a large volume of high-resolution videos must be transported from street cameras to distributed MEC servers for processing, task uploading and/or computing may fail when the spectrum/computing resources are insufficient to meet service demands at the spot. To enhance resource utilization, a data transportation network is needed for task delivery from end users to appropriate edge servers with available computing and spectrum resources, potentially involving multi-hop delivery [16], [17]. Therefore, future research efforts in MEC systems ought to explore better coordination and collaboration between MDs and ESs to address computation offloading issues, leading to improved throughput performance, better workload balancing, and ultimately, enhanced resource utilization [18], [19].
Aiming at small computing latency while avoiding network congestion, Dai et al. [20] recently proposed a cooperative offloading framework in device-to-device (D2D)-assisted MEC networks, where both ESSs and idle MDs enable offloading services for computing-intensive industrial tasks. Here, each user delivers offloading service for at most one neighbor MD to avoid queuing latency as the communication coverage by D2D links is small. Chukhno et al. [21] emphasized the importance of multi-hop relaying for achieving reliability in public safety services, a key technology expected to enhance system performance in future 5G+ systems. For instance, multi-hop relaying allows establishing direct connections between devices outside the coverage area, ensuring first responders have the necessary connectivity, especially in hazardous situations. Notably, the Third Generation Partnership Project (3GPP) has identified new study and work items for New Radio (NR) Vehicle-to-Everything (V2X) side-link (SL) communication within Release 17, proposing the concept of MD relaying to extend coverage range [22]. Different from utilizing a single relay, which is referred to in 3GPP as a single-hop NR SL-based relay, forward compatibility for multi-hop relay support in a future release will be taken into account [22]. The ongoing standardization progress underscores the importance and feasibility of relaying in vehicular networks, which can be leveraged for computing task offloading.

Motivated by the performance enhancement brought by such multi-hop D2D transmissions, in our prior work [23], we have explored vehicle-assisted multi-hop transmissions to balance the computing workload at ESSs under a simple scenario with one MD and multiple ESSs. In this paper, we propose to employ vehicles ubiquitously available in a city to form a data transportation network, which could facilitate multi-hop task offloading between a user and the associated ESS. Due to the omnipresence of vehicles, this approach is economically sound because no additional fixed relays are needed [24]. Moreover, thanks to the short device-vehicle and inter-vehicle distances, MDs and vehicles can employ short-range multi-hop transmissions with low transmit power, thereby causing less interference and improving network-wide spectrum reuse [25]. Expanding to a more general case introduces new challenges. One critical issue in multi-hop routing is striking a balance between communication overhead and computing capability to meet quality-of-service (QoS) requirements [26], [27], increasing the complexity of the task offloading problem. This complexity is further compounded by the dynamic nature of network topology due to vehicular mobility [28], [29].

To address these challenges, we first propose a general multi-hop task routing framework for a vehicle-assisted collaborative edge computing system. This framework allows for the simultaneous establishment of multiple end-to-end (e2e) paths between users and remote ESSs via multi-hop transmissions involving different groups of relay vehicles and the destination ESS. This introduces a novel task routing design problem, optimizing the selection of relay vehicles, the target ESS, and routing paths for different users to balance communication and computing workloads and maximize the number of successfully processed task sizes across the whole system. When addressing the aforementioned issues, we encounter four main challenges. First, the vehicular network environment is highly complicated and dynamic, which can hardly be captured by an accurate and mathematically solvable model. Thus, traditional task offloading methods are not suitable in this scenario. Second, dynamic task routing decisions are jointly made with task-server assignments, which is more challenging than traditional routing with predetermined source and destination nodes. Third, the network-wide tradeoff between the communication and computing workloads further complicates the task routing problem. At last, typical solutions based on queueing theory may not work well for the situation involving multi-hop routing and e2e QoS guarantees because a few strong assumptions (e.g., task arrivals at every source node and intermediate node follow a Poisson process) underpinning the analytical results may not hold and many problems in multi-point to multi-point queuing networks still remain open [30]. Although the system capacity can be enhanced by coordinating the network-wide resources, it is hard to guarantee service reliability with the uncertainty of vehicular trajectories. To efficiently address these issues, we resort to the multi-agent deep deterministic policy gradient (MADDPG) method, a powerful deep reinforcement learning (DRL), which is capable of addressing issues with high dimensional states and huge action spaces [31]. To this end, we present a novel and highly effective MADDPG-based multi-hop task routing approach by learning network dynamics. Note that our approach is not restricted to vehicle-aided MEC and it can be easily extended to other multi-hop MEC systems facing similar challenges.

Our main contributions can be summarized as follows.

- We present the first framework for multi-hop task routing in the context of vehicle-assisted collaborative edge computing. This innovative framework effectively coordinates resources in a multi-edge multi-user MEC system, enabling the establishment of multiple e2e paths through multi-hop transmissions involving relay vehicles and destination ESSs.
- We formulate a throughput maximization problem that addresses the tradeoff between communication and computing resource constraints, as well as end-to-end latency requirements. This optimization problem aims to balance the workloads across the system and maximize the number of successfully processed task sizes by optimizing the task routing path.
- We adopt a model-free DRL method, specifically MADDPG approach, to efficiently address the challenges posed by vehicular mobility and channel variability. Through interactions with the vehicle-assisted MEC environment, this novel solution effectively learns the undetermined model and provides an optimal task-routing strategy. Extensive simulations demonstrate that the proposed MADDPG-based approach significantly enhances system performance.

The remainder of this paper is organized as follows. In Section II, we present the related works. Section III describes the system model and problem formulation. In Sections IV and V, we present the preliminaries for MADDPG and the MADDPG-based task offloading scheme, respectively. Section
VI presents simulation results, and Section VI concludes this paper.

II. RELATED WORK

Most existing works on computation offloading in MEC focus on single-hop offloading from MDs to ESs. They can be roughly divided into two categories according to whether the cooperation between ESs is involved: resource optimization for an MEC with a single ES [8]–[10] or cooperative MEC over multiple ESs [11], [12]. In this section, we will first review the research status according to the above two categories, and then survey the related works on multi-hop task offloading from MDs to ESs.

A. Resource optimization for an MEC with a single ES

Cao et al. [8] proposed computation partitioning, dispatching, and scheduling algorithms for 5G-based edge computing systems, under the assumption that there is plenty of spectrum bandwidth, to support the data transmissions between MDs and an ES, which could parallelize computing tasks and fully utilize the computing resources at both the ES and MDs. Based on the observations that a considerable amount of data should be pre-stored and asymmetric spectrum bandwidth is required for uplink and downlink transmissions to support many emerging services (e.g., Augmented Reality (AR) services) at ESs, Poularakis et al. [9] studied the joint optimization of service placement and computation offloading for MEC networks with storage, computation, and communication constraints. While the optimization problem here is probably the most general one to minimize the computing workload offloaded to the centralized cloud under the above system consideration, they did not consider queueing at ESs, which is commonly encountered in practical systems. In [10], Deng et al. proposed a scheme to maximize the task completion ratio (throughput) in MEC under e2e latency constraints by using a tandem queue model to characterize the joint resource allocation of communications and computing. They also considered the stochasticity of involved processes, e.g., task arrivals, random channels, and varying computing power. However, this paper only focuses on a single MD and single ES scenario.

It is also observed that all these works only study the scenarios that the computing tasks can directly be transmitted to the destination ES within the MD’s communication range (i.e., one-hop) at one single ES without considering the cooperation among ESs.

B. Cooperative MEC over multiple ESs

By exploiting cooperation among ESs, tasks that arrive at one ES can be either processed locally or partially/fully offloaded to powerful ESs via backbone or backhaul links to enhance the quality of experience (QoE). In [11], Li et al. proposed an online cooperative offloading mechanism to optimize the decision of task admission and scheduling among ESs with the objective to minimize the long-term system cost by considering full offloading (i.e., binary offloading). In [12], Li et al. extended the cooperative computing framework in MEC to vehicular networks by considering challenges in computing result delivery due to the uncertainty of vehicular mobility. To address the complexity resulting from the dynamic network topologies in MEC-enabled vehicular networks, they proposed a location-aware offloading and computing strategy to coordinate ESs with partial offloading (i.e., computing at multiple ESs in parallel). However, they still assume that backbone/backhaul links have plenty of bandwidth, and hence will not pose any constraints on communications between ESs.

It is also observed that all these works enable resource-constrained ESs to help each other in processing computation-intensive tasks, thereby enhancing computing workload balancing and resource utilization in MEC systems.

C. Multi-hop task offloading between MDs and ESs

The aforementioned research works generally make an implicit assumption that MDs can only offload tasks to ES one-hop away, which significantly restricts the solution space and limits resource sharing. For example, when a nearby server is overwhelmed with its computing, it is natural to offload a MD’s task to other servers potentially unreachable by one-hop communications, or when too many MDs at one-hop away ESs are excessive, there is no spectrum used to offload data to one-hop away ESs, while there may exist multi-hop path connecting to multi-hop away ESs. In either case, multi-hop offloading may be leveraged to increase resource sharing and load balancing.

As far as we know, [18] and [23] are probably the most related works tackling vehicle-assisted multi-hop task offloading as done in this paper. In [18], Hui et al. designed a request relay mechanism for MEC-enabled vehicular networks to reduce the cost of the relay service by taking the dynamic traffic conditions and the reputation of vehicles into consideration. However, they merely considered the limited transmission ranges of vehicles and ESs while ignoring resource constraints and QoS requirements. In [23], Deng et al. proposed a load-balanced relay mechanism for MEC in which the relay vehicle and destination ES are jointly determined according to the queuing status at an MD and traffic status, significantly enhancing the system performance. Nevertheless, their work just considered a simple case with a single MD where the complicated task routing between multiple MDs and destinations is not involved. Different from these works, this paper intends to employ vehicles as relays for computing task delivery by taking advantage of the mobility and spectrum opportunities in vehicular environments. To deal with the “curse of dimensionality” arising from large-scale vehicular networks, we use DRL to find the multi-hop task routing paths.

III. SYSTEM MODEL AND PROBLEM FORMULATION

In this section, we first provide a comprehensive overview of the multi-hop task routing framework proposed in this paper. Then, we detail the system model and problem formulation. For convenience, the main notations used are summarized in Table I.
TABLE I:
Main Notations

<table>
<thead>
<tr>
<th>Notation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\mathcal{I})</td>
<td>Set of mobile devices (MDs)</td>
</tr>
<tr>
<td>(\mathcal{N})</td>
<td>Set of vehicles</td>
</tr>
<tr>
<td>(\mathcal{J})</td>
<td>Set of edge servers (ESs)</td>
</tr>
<tr>
<td>(\mathcal{P})</td>
<td>Set of feasible routes for MDs</td>
</tr>
<tr>
<td>(\mathcal{T})</td>
<td>Duration for the considered time-slotted system</td>
</tr>
<tr>
<td>(p_i)</td>
<td>Route from MD (i) to a destination ES</td>
</tr>
<tr>
<td>(a_{i,p_i}(t))</td>
<td>Routing decision for MD (i) in time slot (t)</td>
</tr>
<tr>
<td>(W_i(t))</td>
<td>Size of data generated by MD (i) in time slot (t)</td>
</tr>
<tr>
<td>(L_{i,p_i}(t))</td>
<td>E2e service latency for MD (i)</td>
</tr>
<tr>
<td>(D_i)</td>
<td>E2e service latency requirement for MD (i)</td>
</tr>
</tbody>
</table>

A. Multi-hop task routing framework for vehicle-assisted collaborative edge computing

As shown in Fig. 1, our multi-hop task routing framework comprises multiple mobile devices (MDs) and edge servers (ESs), with \(N\) distributed vehicles acting as communication relays between the MDs and remote ESs. Without loss of generality, we consider two types of MDs: pedestrians located on the roadside and passengers in vehicles. The tasks generated by MDs can be offloaded either to the ESs within their communication range or to remote ESs through multi-hop transmission paths facilitated by the vehicles. Notably, our proposed approach is independent of the specific mobility model for vehicle movement. We operate in a time-slotted system denoted by \(t \in \mathcal{T} = \{0, 1, 2, \ldots, T\}\). At any given time, each MD is associated with at most one ES since the task is indivisible while respecting the resource constraint that each ES can serve at most \(K\) MDs through proper user scheduling [32]. As such, our focus lies in designing a routing path for tasks generated by a set of MDs within a single time slot. For convenience, we denote the sets of MDs, vehicles, and ESs as \(\mathcal{I} = \{1, 2, \ldots, I\}\), \(\mathcal{N} = \{1, 2, \ldots, N\}\), and \(\mathcal{J} = \{1, 2, \ldots, J\}\), respectively.

MD \(i\) may fail to access the service of an MEC server within its deadline \(D_i\) because i) the computing workload at the surrounding ESs is excessive or ii) the ES is out of its communication range or iii) the channel condition between MD \(i\) and the ES is poor or transmission channel between MD \(i\) and its surrounding ESs are excessively busy. Thus, it is possible that MD \(i\) seeks help from vehicles on the road to relay its data to an appropriate ES multi-hop away, improving the system capacity. We assume that a global controller has global knowledge of the network dynamics and makes offloading decisions for all users in a centralized manner. For example, we could take software-defined networking design approach to implement our proposed MEC systems. To conclude, a multi-hop MEC service session for a task includes the following four steps while we ignore the procedure of result returning since the size of results in many practical applications (e.g., object detection results) is relatively small.

1) Offloading: When a computing task is generated at MD, it selects a relay vehicle within its communication range and offloads the computing data of the task to the vehicle. In this paper, we employ a binary offloading scheme, wherein the service task cannot be divided but is rather offloaded in its entirety or not offloaded at all.

2) Relaying: After a vehicle receives the computing data from the MD, it initiates data transfer across vehicles on the road by selecting an appropriate route to the destination ES. We assume that the vehicles cannot perform data reception and data transmission simultaneously, which means a vehicle can either receive data from one source (MD) or send data to another vehicle or the ES, but not both simultaneously [33].

3) Uploading: When the relay vehicle arrives within the communication range of the destination ES, it proceeds to upload the data it carries to the ES.

4) Computing: After the computing data is fully offloaded, the destination ES can begin processing the computing task. Once the computing process is completed, the ES sends back the result to the MD.

B. Task routing model

The route from MD \(i\) to a destination ES is represented by \(p_i \in \mathcal{P}\), where \(\mathcal{P}\) is the set of feasible routes and it includes options for either direct one-hop transmission from MD \(i\) to the ES or multi-hop routing involving multiple vehicles and one ES. We denote the routing decision for MD \(i \in \mathcal{I}\) at time slot \(t\) as \(a_{i,p_i}(t) = \{0, 1\}\). Based on the analysis presented earlier, we can formulate the following constraint:

\[
\sum_{p_i \in \mathcal{P}} a_{i,p_i}(t) \leq 1, \forall i, t.
\]  

According to the propagation model in 3GPP standards [12], the path loss between a transmitter and a receiver with distance \(d\) (km) can be computed as:

\[
\Psi(d) = 40 \left(1 - 4 \times 10^{-3}H\right) \log_{10}d - 18 \log_{10}H + 21 \log_{10}f + 80(dB),
\]  

where \(H\) and \(f\) are the antenna height in meter and the carrier frequency in MHz, respectively. The distance between node \(a\) and \(b\) is denoted as \(D_{a,b}\). Thus, from the Shannon capacity
theorem, the data rate between node $a$ and $b$ can be expressed as:
\[ R_{a,b}(t) = B \log_2 \left( 1 + \frac{P \cdot 10^{-\frac{\psi(D_{a,b})}{10}}}{\sigma^2} \right), \]

where $\sigma^2$ denotes the power of the Gaussian noise in the channel (e.g., the user-to-vehicle channel, the vehicle-to-vehicle (V2V) channel, or the vehicle-to-infrastructure channel), $P$ represents the node’s transmit power, and $B$ represents the spectrum bandwidth used by the MD.

\[ L_{\text{queue}}(t) = \max \left\{ \sum_{i' \in I_{t,i}} L_{\text{comp}}(t-1) - \epsilon, 0 \right\}, \]

where $\epsilon$ is the duration of a time slot.

Given the transmission latency, computing latency, and queueing latency, the e2e service latency for MD $i$ can be formulated as follows:
\[ L_{i,p_i}(t) = L_{\text{trans}}(t) + L_{\text{comp}}(t) + L_{\text{queue}}(t). \]

D. Problem formulation

As mentioned earlier, in vehicle-assisted collaborative edge computing, a fundamental challenge is to devise a multi-hop routing policy, denoted as $\alpha$, that optimizes the service throughput by effectively managing the communication overhead and computing capability while satisfying quality-of-service requirements. Based on the aforementioned analysis, we now formulate the following problem with the aim of maximizing the aggregated throughput for the vehicle-assisted multi-hop edge computing system while satisfying the e2e latency requirements from MDs, i.e.,
\[ \max_{\alpha} \sum_{t \in T} \sum_{i \in I} \sum_{p_i \in P} \alpha_{i,p_i}(t) W_i(t) \cdot \mathbb{1}_{\{L_{i,p_i}(t) \leq D_i\}} \] where $\mathbb{1}_{\{L_{i,p_i}(t) \leq D_i\}}$ is the indicator function whose value takes 1 when the e2e service latency requirement of MD $i$ is satisfied, or 0 otherwise. Note that $L_{i,p_i}(t)$ is calculated after the task generated by user $i$ at time slot $t$ is accomplished in the current or the future time slot. Moreover, the objective function in (11) represents the total size of the tasks completed with latency requirements during the considered time duration $T$. With the optimization objective in (11), we have to take into account communication and computing resource constraints.

To address problem (11), we are confronted with three major challenges that require thoughtful consideration and innovative solutions. First, expressing the e2e latency in a closed form is difficult. Traditional queueing theory, commonly used for latency analysis, may not be suitable in this context since the assumption of task arrivals at every node does not hold for multi-point to multi-point scenarios, making it challenging to derive closed-form solutions. Second, directly solving problem (11) using traditional optimization methods is infeasible as it involves a mixed-integer non-linear optimization problem. Such problems are notoriously difficult to solve efficiently.

Third, when we resort to learning-based solutions, the curse of dimensionality presents a significant challenge in terms of both state space and action space. For instance, considering the action variable, $\alpha_{i,p_i}(t)$, there exists $I \times N \times J$ decisions to make in each time slot, leading to a large number of possible combinations that quickly become impractical to explore.

IV. PRELIMINARIES FOR DEEP DETERMINISTIC POLICY GRADIENT

In this section, we reformulate problem (11) as a Markov decision process (MDP) to enable an effective solution using the deep deterministic policy gradient (DDPG) method.

A. MDP-based Task Routing Model

To solve problem (11) efficiently, we first model it as a Markov decision process (MDP) $(S, A, P, R)$, where $S$ and $A$ are the sets of system states and actions, respectively, and $P$ and $R$ are the functions of state transition and reward, respectively. The specific definitions are given below.

*State space: The design of state space is to reflect the status of the considered system completely and informatively. Therefore, we build the state space $S$ consisting of vehicle status, server status, and system workload. Vehicle status
provides the information of the feasible relay vehicles and the channel states among vehicles. Server status includes the computing capability of ESs and the available bandwidth. System workload provides the information of the amount of input data from MDs and the number of queuing tasks at ESs.

Action space: Based on the observed state, the actions can be chosen from the feasible action space \( \mathcal{A} \) in each time slot whose element represents the routing path for each MD.

Transition probability: Transition probability in MDP represents the probability that the system state moves from the current state \( s \) to the next state \( s' \) when action \( a \) is taken, i.e., \( P_{ss'} = \mathbb{P}\{s'|(s,a)\} \).

Reward: In an MDP, the reward is related to both state and action. When an action, e.g., a task scheduling policy, is selected under the current state, the corresponding reward will be received from the system, i.e., \( R_a^s = \mathbb{E}\{R|(s,a)\} \). In the considered problem, the reward function can be set according to the objective function (11).

For the MDP, \( \pi(s,a) : \mathcal{S} \times \mathcal{A} \rightarrow [0,1] \) is set to a policy that gives the probability of taking action \( a \) when in the state \( s \). To obtain the expected long-term discounted reward, the value function \( Q \) of state \( s \) by taking policy \( \pi \) is

\[
Q(s, \pi) = \mathbb{E} \left[ \sum_{t \in T} \gamma^t R_a^s(t) \right],
\]

where \( \gamma \in [0,1) \) is a discounting factor. By maximizing the value function across different states, we can obtain the optimal task scheduling policy \( \pi^* \):

\[
\pi^*(s,a) = \arg \max_s \sum_{s'} \mathbb{P}(s'|(s,a))[R(s,a) + \gamma Q(s', \pi^*)].
\]

B. Deep Deterministic Policy Gradient

For the optimization in (13), the traditional dynamic programming is not applicable as we lack knowledge about the transition probability \( \mathbb{P} \) in the considered system. To address this issue, we adopt the DDPG method as our approach, which offers several advantages. DDPG is a model-free method, meaning it can learn the underlying model through interactions between agents and the environment. This characteristic is particularly beneficial for our vehicle-assisted collaborative edge computing system, where accurately modeling the end-to-end (e2e) latency is challenging due to the dynamic and complex nature of the vehicular network environment. Moreover, one of DDPG’s key strengths is its ability to handle problems with high-dimensional state spaces and large action spaces. By utilizing deep neural networks, DDPG can efficiently approximate complex mappings between states and actions, making it well-suited for our dynamic and complex vehicular network environment. Besides, DDPG has demonstrated its effectiveness in achieving stability and convergence in complex tasks, which is crucial for addressing the challenges posed by our scenario [34].

In DDPG, there are a total of four networks: the Actor, the Critic, and the corresponding target networks for the Actor and Critic, respectively. The target networks can be regarded as time-delayed copies of their original networks that slowly track the learned networks, which will significantly enhance the stability of learning. The specific functions of these four neural networks are as follows.

1) Actor network: The Actor network is in charge of the iterative update of policy network parameters and the direct maps from the current state to the current action. In this way, it interacts with the vehicle-assisted multi-hop MEC environment to generate the next state and reward.

2) Actor target network: The Actor target network outputs the next optimal action according to the next state sampled in the experience replay. The network parameters in the Actor target network are periodically copied from the Actor network.

3) Critic network: The Critic network is responsible for the iterative update of the parameters in the value network and calculating the current Q value.

4) Critic target network: The Critic target network calculates \( Q' \) value according to the next state-action. The network parameters in the Critic target network are periodically copied from the Critic network.

The above two target networks have “soft”-updates based on main networks, i.e., the target networks only update a small part based on the current network, to improve the stability of learning. That is,

\[
\theta' \leftarrow \tau \theta + (1 - \tau)\theta',
\]

\[
w' \leftarrow \tau w + (1 - \tau)w',
\]

where \( 0<\tau \ll 1 \) is the update frequency for the parameters in actor target network (\( \theta \)) and critic target network (\( w \)).

To improve the exploration capability and thus avoid getting stuck in a local optimum, DDPG typically adds noise (\( \mathcal{N}_t \)) to the action (\( \pi_{\theta}(s) \)) produced by the actor network to get a new action, i.e.,

\[
a = \pi_{\theta}(s) + \mathcal{N}_t.
\]

The loss functions for the critic network and the actor network are respectively defined as

\[
L(w) = \frac{1}{m} \sum_{z=1}^{m} (y^z - Q(\phi(S^z), A^z, w))^2,
\]

and

\[
L(\theta) = -\frac{1}{m} \sum_{z=1}^{m} Q(s, a, \theta), \quad z = 1, 2, \cdots, m,
\]

where \( m \) is the number of samples (including eigenvector of state \( \phi(S^z) \), and action \( A^z \)) from Replay Buffer \( D \), \( y^z \) is the target value of \( Q \).

V. MADDPG-BASED MULTI-HOP TASK ROUTING IN VEHICLE-ASSISTED COLLABORATIVE EDGE COMPUTING

In this section, we present our approach for leveraging the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) method to solve the task routing problem in our vehicle-assisted collaborative edge computing system.
Although DDPG can adapt to the environment of multi-dimensional actions, it is difficult for a single super-agent to learn large-scale decentralized policies whose action space grows exponentially with the number of participants [35]. MADDPG is an intuitive extension to the DDPG algorithm under a multi-agent system by decomposing a single monolithic agent into multiple simpler agents to reduce the dimensionality of the state and action spaces and thus overcome the scalability issue. In MADDPG, each agent makes the most suitable decision for itself, and multiple agents can achieve the common goal through cooperation. In this paper, we take advantage of MADDPG to train multiple agents for the optimization of multi-hop task routing in vehicle-assisted collaborative edge computing.

### A. State Space, Action Space and Reward Function

1) **State Space:** The state observed by MD $i$ at time slot $t$ is defined as

$$ s_i(t) = \{s_i^w(t), s_i^a(t), s_i^n(t), s_i^l(t)\}, \quad (19) $$

where $s_i^w(t)$ represents the number of tasks of MD $i$ arriving in time slot $t$, $s_i^a(t)$ denotes the indexes of the selected ESs for all MDs at time slot $t$, $s_i^n(t)$ denotes the number of MDs which select the same ES in time slot $t$, and $s_i^l(t)$ denotes the remaining task size in the buffer at each ES in time slot $t$.

2) **Action Space:** In the system, every MD has to decide the serving ES. Thus, the action of user $i$ at time slot $t$ is expressed as

$$ a_i(t) = \{a_i^s(t)\}, \quad (20) $$

where $a_i^s(t)$ is the index of the ES selected by MD $i$. Besides, let $A_i^s(t)$ denote the set of the selection actions by the feasible destinations. Therefore, action $a_i(t)$ is valid if $a_i^s(t) \in A_i^s(t)$. Note that the routing path from MD $i$ to its destination ES will be uniquely determined if the ES is selected in MADDPG. For example, in this paper, we use the shortest path in terms of the travel distance between an MD and the associated ES.

3) **Reward Function:** Since each MD intends to maximize its completed tasks while meeting the required e2e latency, the immediate reward is represented as

$$ r_i(t) = c_i(t), \quad (21) $$

where $c_i(t)$ is defined as the total size of accomplished tasks in time slot $t$ within the deadline, including tasks generated in the current time slot and those queued in the buffer. Note that the choice of the reward function is to approximately maximize the objective function defined in (11), i.e., the number of tasks accomplished with latency requirements in the long run.

The gained reward depends on the action of an MD, i.e., the MD gets an immediate reward $r_i(t)$ given observed state $s_i(t)$ and action $a_i(t)$ in time slot $t$. Each MD aims at learning the optimal policy which maximizes the long-term reward, which is given by

$$ R_i(t) = \max\ E \left[ \sum_{k=0}^{T-1} \gamma^k r_i(k + t) \right], \quad (22) $$

where $T$ is the number of consecutive time slots for calculating the long-term reward and $0 < \gamma < 1$ is the discounting factor for determining the importance of the immediate reward and future rewards, where a smaller $\gamma$ means that more importance is given to the immediate reward.

### B. The Training and Execution of MADDPG

Fig. 2 illustrates the framework of MADDPG with two main procedures: i) using the global information to train the critic network, which is different from the traditional DDPG algorithm; and ii) using the local information to execute the actor network. Suppose that there are $I$ agents (corresponding to the MDs in our system) in the vehicle-assisted MEC environment, in which we have two assumptions: i) the policy of each MD depends only on its own observed state, ii) the environment is unknown, and thus the reward for each agent and the next state after taking an action is unpredictable, which can only be acquired through the feedback from the environment.

**Global training for the Critic network:** In the training of MADDPG, the actor network selects an action according to the current state, and then the critic network can calculate a Q value according to the state-action pair as feedback to the action. The critic network is trained based on the estimated Q value and the actual Q value, and the actor network updates the policy based on the feedback from the critic network. To speed up the learning process of an MD, the input to the critic network for training includes both its own observation and the observations (e.g., the states and actions) of other agents in the environment. The parameters in the critic network are updated by minimizing the loss function based on Eq. (17).
Algorithm 1: Multi-Agent Deep Deterministic Policy Gradient for Task Routing in Vehicle-assisted MEC

**Input:** the number of arriving tasks for each MD, the locations of MDs, vehicles, and ESs, e2e latency requirements for MDs, computing capabilities of ESs, spectrum resources

**Output:** task routing policy

<table>
<thead>
<tr>
<th>for each episode do</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize a Gaussian noise ( N ) for action exploration</td>
</tr>
<tr>
<td>Receive initial state ( s )</td>
</tr>
<tr>
<td>for each time slot ( t = 1, 2, \cdots, T ) do</td>
</tr>
<tr>
<td>For each agent ( i ), select action ( a_i = \pi_{\theta_i}(s_i) + \mathcal{N}_i ) w.r.t the current policy and exploration</td>
</tr>
<tr>
<td>Execute actions ( a = (a_1, \cdots, a_I) ) and obtain rewards ( r ) and new state ( s' ) from the environment</td>
</tr>
<tr>
<td>Store ((s, a, r, s')) in Replay Buffer ( D )</td>
</tr>
<tr>
<td>for each agent ( i \in I ) do</td>
</tr>
<tr>
<td>Sample a random minibatch of ( m ) samples ((s^z, a^z, r^z, s'^z), z = 1, 2, \cdots, m), from ( D )</td>
</tr>
<tr>
<td>Set ( y_i^z = r_i^z + \gamma Q_i^z(s'^z, a'^z) )</td>
</tr>
<tr>
<td>UpdateDDPG</td>
</tr>
</tbody>
</table>

Procedure: UpdateDDPG

15 Update critic network by minimizing the loss function for \( w_i \)

\[
L(w_i) = \frac{1}{m} \sum_{z=1}^{m} (y_i^z - Q(\phi(s^z), A^z, w_i))^2
\]

16 Update actor network by minimizing the loss function for \( \theta_i \)

\[
L(\theta_i) = -\frac{1}{m} \sum_{z=1}^{m} Q(s_i^z, a_i^z, \theta_i)
\]

17 Update target network parameters for each MD \( i \)

\[
w'_i \leftarrow \tau w_i + (1 - \tau) w_i' \\
\theta'_i \leftarrow \tau \theta_i + (1 - \tau) \theta'_i
\]

Local execution for the actor network: When each MD is fully trained, each actor network outputs appropriate actions according to its own state without the observed information from other MDs. The parameters in the actor network are updated using gradient descent according to the loss function based on Eq. (18).

The training algorithm is summarized in Algorithm 1. We omit the introduction to Algorithm 1 due to the page limit. Please refer to [36] for more information.

VI. PERFORMANCE EVALUATION

In this section, we conduct extensive evaluations and comparisons to assess the performance of the proposed multi-agent deep deterministic policy gradient (MADDPG)-based task routing in vehicle-assisted collaborative edge computing, comparing it with other benchmark algorithms. The simulation experiments carry out on a ThinkPad X1 Carbon with a 4.7 GHz 12-Core Intel Core i7-1260P processor. We use two performance metrics, namely average throughput and success rate, to evaluate the algorithms.

A. Simulation Settings

1) Simulation Parameters: As shown in Fig. 3, we conduct performance evaluations in a simulated road network to assess the proposed MADDPG-based task routing approach in a vehicle-assisted edge computing scenario. The road network consists of 4 stationary edge servers (ESs) (depicted as black circles) deployed as indicated in the figure, and 10 moving vehicles (represented by blue rectangles) following traffic rules, such as speed limits, safe distance, and traffic lights. The length of each considered vehicle is set to 4 meters, and the safe distance between vehicles was required to be no less than 4 meters. Besides, the speed limit within the simulated road network is set to 60 km/h.

The positions of mobile devices (MDs) in the network are randomly initialized at the beginning of the simulation and remained fixed throughout the evaluation. For each MD, computing tasks are generated following a Poisson process, and the task size is randomly distributed in the range of \( [2, 5] \times 10^5 \) Kbits. In the simulation, we use the parameter \( \beta \) to represent the probability of generating tasks for each MD in each time slot. Moreover, a fair spectrum allocation rule among links is applied, proportionally allocating the total bandwidth according to the size of transmitted tasks [37]. Other simulation parameter settings are given in Table II. The simulation is conducted for a duration of 100 seconds.

2) MADDPG Hyperparameters: In the MADDPG approach, we utilize specific hyperparameters to train the actor and critic networks, enabling efficient task-routing decisions. The actor
network is designed as a four-layer neural network with two fully connected hidden layers, each consisting of 256 units and activated by sigmoid functions. The number of units in the input and output layers of the actor network is set to match the dimension of states and actions, respectively. For the critic network, it receives inputs from both the actor network’s actions and its states. To process these inputs, the critic network employs two hidden layers for the states and one hidden layer for the actions. These two inputs are concatenated before being passed through two additional fully connected hidden layers, each containing 256 units and activated by ReLu functions. Finally, the critic network produces an output layer responsible for calculating the Q value for a given state-action pair, without using any activation function. Other hyperparameters used for training MADDPG can be found in Table III.

### TABLE III: Simulation Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coverage Radius of ESs</td>
<td>200 m</td>
</tr>
<tr>
<td>Height of antenna</td>
<td>1.5 m</td>
</tr>
<tr>
<td>Carrier frequency</td>
<td>2800 MHz</td>
</tr>
<tr>
<td>Computation complexity</td>
<td>1200 CPU cycles/bits</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>5 MHz</td>
</tr>
<tr>
<td>Computing rate at ESs</td>
<td>$[1, 2, 3, 4] \times 10^7$ cycles/s</td>
</tr>
<tr>
<td>Transmit power of MD</td>
<td>1 W</td>
</tr>
<tr>
<td>Power of the Gaussian noise</td>
<td>$5 \times 10^{-13}$ W</td>
</tr>
<tr>
<td>Discounting factor</td>
<td>0.99</td>
</tr>
</tbody>
</table>

**B. Compared Methods**

To evaluate and compare the performance of the proposed MADDPG-based task routing method (MADDPG) in the context of vehicle-assisted collaborative edge computing, we present two baseline algorithms for comparison:

1) **Single-hop**: This algorithm involves a direct task offloading approach from an MD to its nearest ES through one-hop transmission, taking into account both communication and computing resources. However, when the workload exceeds the capacity of nearby servers, tasks are rejected and not admitted into the system. An example of this approach can be found in [10] where a game theoretic task offloading algorithm is proposed.

2) **Multi-hop+Greedy**: This algorithm represents a naive solution for multi-hop task routing. When the workload exceeds the capabilities of local ESs, the remaining tasks are delivered to ESs within the shortest distance via multi-hop transmission from the MD to the target ES. It is important to note that this algorithm has not been thoroughly investigated yet, and it is introduced here to serve as a baseline for performance comparison.

To conduct the comparisons, we employ two essential metrics: the average service throughput (average throughput) and the average success rate (success rate). Average throughput provides insight into the average size of completed tasks from MDs during T time slots. Success rate measures the average ratio of completed tasks (including newly arrived tasks and those buffered at ESs) to the total generated tasks in each time slot.

**C. Evaluation of the average throughput**

In the evaluation of average throughput, we conduct a comparison among Single-hop, Multi-hop+Greedy, and MADDPG. The results are depicted in Fig. 4.

Fig. 4a shows that average throughput of MADDPG increases with the task arrival rate. This improvement can be attributed to the fact that as more tasks arrive from MDs or the number of ESs in the system increases, MADDPG can efficiently handle and complete a larger number of tasks. This adaptive behavior allows MADDPG to take advantage of the available resources and provide better task completion rates. Fig. 4b illustrates average throughput of MADDPG in relation to the number of MDs within the range of [10, 20]. We observe that average throughput increases with the number of MDs. However, it reaches a maximum value when the number of MDs reaches 16. This can be attributed to the limited availability of system resources, where the capacity of the ESs might become saturated, leading to a saturation point in average throughput. In Fig. 4c, it is demonstrated that MADDPG achieves a relatively stable average throughput as the number of vehicles varies from 2 to 10. The adaptive multi-hop task routing decisions made by MADDPG, through interaction with the environment, allow for efficient utilization of resources in the dynamic vehicular network. This adaptability leads to a consistent average throughput across different numbers of vehicles. Fig. 4d indicates that average throughput of MADDPG increases with the number of ESs in the system. As more ESs are available, MADDPG can effectively distribute tasks among them, leveraging the additional computing resources. Consequently, this leads to higher overall system throughput.

Moreover, we note that MADDPG consistently outperforms Single-hop and Multi-hop+Greedy in terms of average throughput. The superior performance of MADDPG can be attributed to its capability of load balancing through multi-hop transmissions between MDs and ESs, providing an advantage over Single-hop, which can only utilize ESs one-hop away, and Multi-hop+Greedy, which may lead to computing overload and network congestion due to unoptimized ES selection. The results of this evaluation demonstrate that the proposed MADDPG-based task routing method is highly effective in improving the average service throughput.

**D. Evaluation of the success rate**

In the evaluation of success rate, Fig. 5 presents a comparison of the success rates of MADDPG with Single-hop and Multi-hop+Greedy.

Fig. 5a illustrates success rate of the three task offloading mechanisms against the task arrival rate. It is observed that MADDPG achieves nearly complete task completion when the task arrival rate is low. In contrast, Single-hop and Multi-hop+Greedy exhibit unstable performance in highly dynamic...
network environments with randomly distributed MDs and frequently moving vehicles. The adaptability of MADDPG allows it to efficiently handle task arrivals even under varying network conditions. Fig. 5b demonstrates that tasks can be nearly completed when the number of MDs varies from 10 to 20. This observation holds true for all three task offloading mechanisms. However, MADDPG’s success rate remains consistently higher in comparison to Single-hop and Multi-hop+Greedy. This is due to MADDPG’s ability to dynamically select destination ESs, providing better utilization of available resources. In Fig. 5c, success rate of the three task offloading mechanisms is evaluated against the number of vehicles. It is evident that MADDPG maintains a high success rate within the range of 2 to 10 vehicles. Once again, this demonstrates MADDPG’s adaptability in making optimal multi-hop task routing decisions based on the changing environment. Fig. 5d presents success rate of the three task offloading mechanisms against the number of ESs. As the number of ESs increases, MADDPG’s success rate approaches 1, indicating near-complete task completion. On the other hand, Single-hop and Multi-hop+Greedy exhibit significantly lower success rates. This superiority of MADDPG is attributed to its capability to adaptively select destination ESs, allowing it to utilize more ES resources effectively.

VII. CONCLUSION

In this paper, we have presented a novel multi-hop task routing framework for vehicle-assisted collaborative edge computing. Our approach enables edge servers that are multi-hop away to share computing workloads, leading to enhanced system capacity and improved resource utilization. By taking into account practical factors such as vehicular mobility, spectrum availability, and computing capabilities, we have established efficient associations between users and edge servers through potentially multi-hop transmissions facilitated by vehicle relaying, a scenario that has been rarely explored before. Utilizing the multi-agent deep deterministic policy gradient (MADDPG) method, each end user acts as an agent to make offloading decisions in an efficient and adaptive manner. This results in achieving high aggregated throughput while ensuring that each user meets its specific end-to-end latency requirements and resource constraints in the dynamic vehicular network environment. Extensive simulations have been conducted to evaluate the performance of the proposed MADDPG-based task routing scheme, and the results demonstrate its effectiveness and efficiency.
REFERENCES


Yiqin Deng

Yiqin Deng is a visiting researcher at the University of Florida, Gainesville, from 2019 to 2021. Her research interests include edge/fog computing, Internet of Vehicles, and resource management.
Haixia Zhang (M’08–SM’11) received the B.E. degree from the Department of Communication and Information Engineering, Guilin University of Electronic Technology, Guilin, China, in 2001, and the M.Eng. and Ph.D. degrees in communication and information systems from the School of Information Science and Engineering, Shandong University, Jinan, China, in 2004 and 2008, respectively. From 2006 to 2008, she was with the Institute for Circuit and Signal Processing, Munich University of Technology, Munich, Germany, as an Academic Assistant. From 2016 to 2017, she was a Visiting Professor with the University of Florida, Gainesville, FL, USA. She is currently a Full Professor with Shandong University, Jinan, China. Dr. Zhang is actively participating in many professional services. She is an editor of the IEEE Transactions on Wireless Communications, IEEE Wireless Communication Letters, and China Communications and serves/served as Symposium Chairs, TPC Members, Session Chairs, and Keynote Speakers of many conferences. Her research interests include wireless communication and networks, industrial Internet of Things, wireless resource management, and mobile edge computing.

Xianhao Chen received the B.Eng. degree in electronic information from Southwest Jiaotong University in 2017, and the Ph.D. degree in electrical and computer engineering from the University of Florida in 2022. He is currently an assistant professor with the Department of Electrical and Electronic Engineering, the University of Hong Kong. He received the 2022 ECE graduate excellence award for research from the University of Florida. His research interests include wireless networking, edge intelligence, and machine learning.

Yuguang Fang (S’92, M’97, SM’99, F’08) received the MS degree from Qufu Normal University, China in 1987, a PhD degree from Case Western Reserve University, Cleveland, Ohio, USA, in 1994, and a Ph.D degree from Boston University, Boston, Massachusetts, USA in 1997. He joined the Department of Electrical and Computer Engineering at University of Florida in 2000 as an assistant professor, then was promoted to associate professor in 2003, full professor in 2005, and distinguished professor in 2019, respectively. Since August 2022, he has been the Chair Professor of Internet of Things with the Department of Computer Science at City University of Hong Kong. Dr. Fang received many awards including the US NSF CAREER Award (2001), US ONR Young Investigator Award (2002), 2018 IEEE Vehicular Technology Outstanding Service Award, IEEE Communications Society AHSN Technical Achievement Award (2019), CISTC Technical Recognition Award (2015), and WTC Recognition Award (2014), and 2010-2011 UF Doctoral Dissertation Advisor/Mentoring Award. He held multiple professorships including the Changjiang Scholar Chair Professorship (2008-2011), Tsinghua University Guest Chair Professorship (2009-2012), University of Florida Foundation Preeminence Term Professorship (2010-2022), and University of Florida Research Foundation Professorship (2017-2020, 2006-2009). He served as the Editor-in-Chief of IEEE Transactions on Vehicular Technology (2013-2017) and IEEE Wireless Communications (2009-2012) and serves/served on several editorial boards of journals including Proceedings of the IEEE (2018-present), ACM Computing Surveys (2017-present), ACM Transactions on Cyber-Physical Systems (2020-present), IEEE Transactions on Mobile Computing (2003-2008, 2011-2016, 2019-present), IEEE Transactions on Communications (2000-2011), and IEEE Transactions on Wireless Communications (2002-2009). He served as the Technical Program Co-Chair of IEEE INFOCOM’2014. He is a Member-at-Large of the Board of Governors of IEEE Communications Society (2022-2024) and the Director of Magazines of IEEE Communications Society (2018-2019). He is a fellow of ACM, IEEE, and AAAS.