A Reinforcement Learning Framework for Vehicular Network Routing Under Peak and Average Constraints

Nan Geng, Qinbo Bai, Chenyi Liu, Tian Lan, Vaneet Aggarwal, Yuan Yang, and Mingwei Xu

***Abstract*—Providing provable performance guarantees in vehicular network routing problems is crucial to ensure safely and timely delivery of information in an environment characterized by high mobility, dynamic network conditions, and frequent topology changes. While Reinforcement Learning (RL) has shown great promise in network routing, existing RL-based solutions typically support decision-making with either peak constraints or average constraints, but not both. For network routing in intelligent transportation, such as advanced vehicle control and safety, both peak constraints (e.g., maximum latency or minimum bandwidth guarantees) and average constraints (e.g., average transmit power or data rate constraints) must be satisfied. In this paper, we propose a holistic framework for RL-based vehicular network routing, which maximizes routing decisions under both average and peak constraints. The routing problem is modeled as a Constrained Markov Decision Process and recast into an optimization based on Constraint Satisfaction Problems (CSPs). We prove that the optimal policy of a given CSP can be learned by an extended Q-learning algorithm while satisfying both peak and average latency constraints. To improve the scalability of our framework, we further turn it into a decentralized implementation through a cluster-based learning structure. Applying the proposed RL algorithm to vehicular network routing problems under both peak and average latency constraints, simulation results show that our algorithm achieves much higher rewards than heuristic baselines with over 40% improvement in average transmission rate, while resulting in zero violation in terms of both peak and average constraints.**

***Index Terms*—Constrained Markov Decision Process, Peak and Average Latency Constraints, Vehicular Network Routing**

I. Introduction

Vehicular networks as a key enabler for intelligent transportation have received growing attention from both industry and academia in recent years [1], [2]. It is expected that an unprecedented amount of data will be shared through real-time communications between vehicles and infrastructure to support various new services such as advanced vehicle control and safety. Traffic routing in vehicular networks that are characterized by high-mobility nodes, dynamic channel conditions, and frequent topology changes requires solving a challenging online optimization problem [2]–[10]. To this end, learning techniques – especially reinforcement learning (RL) – have been employed for online decision making in vehicular network routing problems and showed great promise [11]–[14].

Vehicular network routing must consider two distinct classes of design objectives/constraints – one focusing on *long-term average* performance metrics or utilities (such as average download bandwidth and average latency [2], [15]) and the other focusing on *peak* constraints and guarantees (such as minimum necessary data-rate and maximum-allowable latency [4]). However, existing RL algorithms often focus on either optimization problems with peak constraints (e.g., [16]–[19]) or average constraints (e.g., [20]–[25], but not both. We note that meeting both peak and average constraints is crucial for supporting a wide range of future intelligent transportation applications, in order to achieve diversified and complex performance requirements.

To this end, we propose a holistic framework that (i) develops a novel reinforcement learning algorithm to handle both peak and average constraints, and (ii) optimizes vehicular network routing with respect to both data rate and latency objectives. In particular, an RL agent (i.e., a network controller) is trained by interacting with an environment (i.e., a vehicular network) – the agent observes environment states (e.g., network statistics), takes actions (i.e., routing decisions) according to the current policy, obtains the resulting reward (i.e., optimization objectives), and self-teaches to learn the optimal policy. RL algorithms can efficiently exploit the underlying patterns in the available data and learn the optimal stochastic policy, making them well-suited for vehicular network routing problems. However, existing RL algorithms lack the ability to cope with both peak and average constraints.

We consider the downlink transmission in vehicular networks, where vehicles can establish connections with base stations through Vehicle-to-Infrastructure (V2I) links, as well as proximate vehicles through Vehicle-to-Vehicle (V2V) links. The routing problem is to decide how to deliver data from base station to each individual vehicle – either through a direct V2I link or through a neighboring vehicle as a V2V relay, with the goal of maximizing the long-term average utility of transmission data rates (e.g., the proportional fairness utility). A key feature of our framework is that we consider the routing problem under both peak and average latency constraints. That is, the long-term average transmission latency with respect to all vehicles should not exceed a given bound, resulting in an average latency constraint similar to [20], while the maximum latency experienced by any vehicle at any time should also satisfy a peak latency constraint [16].

Nan Geng, Chenyi Liu, Yuan Yang and Mingwei Xu are with Tsinghua University; Qinbo Bai and Vaneet Aggarwal are with Purdue University; Tian Lan is with George Washington University.

Email: nan_geng@sina.com, bai113@purdue.edu, liucheny19@mails.tsinghua.edu.cn, tlan@gwu.edu, vaneet@purdue.edu,yangyuan_thu@mail.tsinghua.edu.cn, xumw@tsinghua.edu.cn.

We note that it is challenging to straightforwardly deal with both the peak and average constraints together due to their very different mathematical forms. In this paper, a new RL algorithm is proposed to solve the constrained vehicular network routing problem while both peak and average constraints are provably satisfied. To this end, we model the routing problem as a Constrained Markov Decision Process (CMDP) and recast it into an optimization based on Constraint Satisfaction Problems (CSPs). CSP aims at finding a feasible solution which (i) makes the objective value of the routing problem no smaller than a target value and (ii) satisfies both the peak and average constraints, allowing the solution of the routing problem to be approximated by solving a sequence of CSPs. To solve a CSP, first, the peak constraints are absorbed into the average constraints of the CSP through the use of constraint-sensitive functions [16], [17], which generate a penalty whenever any peak constraints are violated. Next, the transformed CSP with only average constraints is solved through an equivalent zero-sum Markov-bandit game solvable by an extended Q-learning algorithm [20]. Finally, we prove that the optimal policy of a CSP can be learned by the algorithm while satisfying both peak and average constraints.

To improve its scalability in large-scale vehicular networks, we further turn the proposed algorithm into a decentralized implementation. Vehicles are partitioned into clusters similar to [2], [26], [27]. Each cluster is then modeled as an individual learning agent that makes independent routing decisions based on only local network information. In particular, different reward structures are developed for intra-cell clusters located in a single cell and for inter-cell clusters traversing multiple cells, in order to permit a minimum level of necessary coordination for vehicles in inter-cell clusters. We note that the use of multiple agents allows us to amortize the training overhead through the use of a multiple Q-table structure for effectively reducing the state and action space of the learning problem. Finally, we evaluate the proposed algorithm by conducting extensive numerical simulations, which show that our approach achieves much higher rewards than some heuristic baselines for vehicular network routing with over 40% improvement in average transmission rate, while resulting in zero violation in terms of both peak and average constraints.

The key contributions of our paper are as follows:

- We propose a holistic framework for vehicular network routing with the goal of optimizing data-rate utilities under both peak and average latency constraints.
- A new RL algorithm is developed to recast the optimization problem (under peak and average constraints) into an optimization based on CSPs that can be optimally solved by an extended Q-learning algorithm.
- The proposed algorithm yields a decentralized implementation through the design of different reward structures for intra-cell and inter-cell clusters.
- Our evaluation results show significant reward and transmission rate improvement over heuristic baselines, while satisfying both peak and average latency constraints.

Next, we introduce the related work in Sec. II and the problem formulation in Sec. III. We describe the proposed algorithm in Sec. IV. The algorithm is turned into a decentralized manner in Sec. V. Finally, we do evaluation in Sec. VI and conclude the paper in Sec. VII.

II. Related Work

Routing algorithms in vehicular networks. Lots of routing algorithms have been developed for vehicular networks, including cooperative routing by coordinating V2I and V2V communication [2], [3], [28], traffic routing in vehicular ad hoc networks [4]–[6], and association problems of vehicles [7], [8], [15]. In particular, the authors in [2] propose a decentralized vehicle cluster management algorithm to maximize the overall transmission rate, while a software-defined network (SDN) structure to enhance the vehicles' cooperation to maximize the number of vehicles that retrieve their requested data is proposed in [3]. Then the authors in [4] present a class of routing protocols for vehicular ad hoc networks (VANETs) in city environments based on the effective selection of road intersections through which a packet must pass to reach the gateway to the Internet. For secure routing, the authors in [5] propose a novel secure and reliable multi-constrained QoS aware routing algorithm for VANETs, which uses ant colony optimization (ACO) technique to compute feasible routes in VANETs subject to multiple QoS constraints determined by the data traffic type. An adaptive quality-of-service (QoS)-based routing protocol is considered in [6] for VANETs called AQRV, which adaptively chooses the intersections to satisfy the QoS constraints and fulfill the best QoS in terms of three metrics. The authors in [15] propose a scheme for a road side unit placement problem that reduces network latency while ensuring good network capacity. However, the existing approaches either ignore latency limitations [2], [3], [28] or consider only one kind of latency constraints/objectives [4]–[6], [15]. Besides, routing in highly dynamic vehicular networks is usually recognized as a very challenging online optimization problem. In this paper, we consider the routing problem with both peak and average latency constraints, which have never been solved efficiently. We propose an RL-based approach with theoretical performance guarantees to solve it.

ML in vehicular networks. There are already some explorations of leveraging ML techniques to solve vehicular network problems such as resource allocation [1], [29], network security [30], [31], energy saving [32], [33], routing optimization [11]–[14]. For routing problems, existing work [11]–[14] mainly focuses on performance improvement without considering QoS (e.g., latency) constraints. To the best of our knowledge, we are the first to leverage RL to develop routing algorithms under both peak and average latency constraints in vehicular networks.

Constrained RL algorithms. Previous RL algorithms either focus on optimization problems with peak constraints [16]–[19], or consider the problems with average constraints [20]–[25], but not both. There are no RL-based algorithms for dealing with both peak and average constraints with theoretical guarantees, to the best of our knowledge. In this paper, we develop a new RL algorithm for this purpose and prove that it yields an optimal policy.

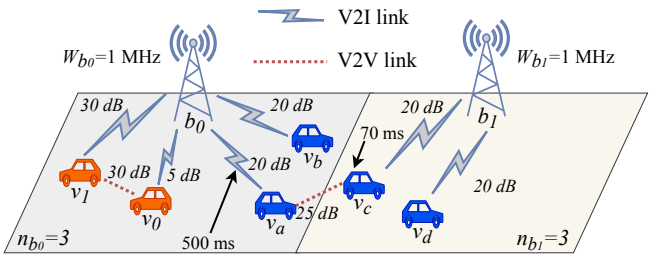

Figure 1: An illustration of the vehicular network routing. The *orange* vehicles belong to an intra-cell cluster, and the *blue* vehicles belong to an inter-cell cluster. The path $b_0 \to v_1 \to v_0$ is assigned to v_0 for a higher transmission rate. The path $b_1 \to v_c \to v_a$ is assigned to v_a for load balancing among cells and small transmission latency. The other vehicles are connected through the direct V2I links.

III. Problem Formulation

A. Vehicular Network Model

We focus on the downlink transmission in vehicular networks. Data can be delivered from base station to each individual vehicle – either through a direct V2I link or through a neighboring vehicle as a V2V relay. Next, we present our model for vehicular network routing.

We consider a vehicular network with B base stations, where each base station b covers a service area (called a cell) and has an amount of W_b cellular resource (e.g., time slots or bandwidth) for V2I communication. Any vehicle v in a cell can establish a cellular V2I link with the corresponding base station. The downlink transmission capacity $c_{b,v}$ of the V2I link between b and v is typically modeled based on the Shannon rate, i.e.,

$$c_{b,v} = W_{b,v} \log(1 + SINR_{b,v}), \tag{1}$$

where $W_{b,v}$ means the cellular resource allocated to v by base station b and $SINR_{b,v}$ denotes the signal-to-interference-plus-noise ratio (SINR) perceived by v associated to b [2]. We define V2I link latency $d_{b,v}$ as the time needed to deliver data from the base station b to the connected vehicle v.

The vehicles in the network are grouped into clusters similar to [26], [27]. Particularly, the clusters located in a single cell are called intra-cell clusters, and those traversing multiple cells are called inter-cell clusters. The vehicles in the same cluster can establish V2V links with each other. Since V2V and V2I links often utilize different sets of wireless resources and protocols, we assume that there is no interference between them. The capacity $c_{v',v}$ of the V2V link between v' and v is also modeled based on the Shannon rate, i.e.,

$$c_{v',v} = W_{v',v} \log(1 + SINR_{v',v}), \tag{2}$$

where $W_{v',v}$ means the fixed resource allocated to v' and $SINR_{v',v}$ denotes the SINR of the V2V link. V2V link latency $d_{v',v}$ is defined as the time needed to transmit data from one vehicle v' to another vehicle v through their established V2V link.

For the vehicle v choosing the direct V2I link for data transmission, it is served by its associated base station b, and its achievable downlink transmission rate x_v can be computed by Eq. (1) directly, i.e., $x_v = c_{b,v}$. For the vehicle v using a relay path consisting of the V2I link between v' and its serving base station b', as well as the V2V link between v and v', its data rate is determined by the minimum of its V2I and V2V link rates, and thus we have $x_v = \min\{c_{b',v'}, c_{v',v}\}$, where $c_{b',v'}$ and $c_{v',v}$ are computed by Eq. (1) and (2), respectively. In this paper, we assume each base station b allocates cellular resources evenly to the vehicles served by b either directly or indirectly following [2]. Let n_b denote the total number of vehicles served by base station b. The vehicles served by b can obtain an equal amount of resource, i.e., $\frac{W_b}{n_b}$, for data transmission. We assume each V2V link is assigned a fixed bandwidth resource for communication, while V2I and V2V communications use different resource pools. Finally, to model the latency, we denote d_v as the downlink transmission latency of vehicle v. For v using direct connection,

$$d_v = d_{b,v}.$$

For v using indirect connection, d_v is simply computed by

$$d_v = d_{b',v'} + d_{v',v},$$

which is a combination of the V2I and V2V link latencies.

In our vehicular network model, both link capacity and latency could be time-varying and driven by the underlying physical network. In this work, we focus on the vehicular network routing problem with dynamic capacity and latency. A similar model is also used in papers such as [2], [34], [35]. We note that the cooperative V2I and V2V communication model enables flexible routing decisions and thus improved performance. Fig. 1 shows the operations of our proposed solution. In particular, our RL-based algorithm observes states of the vehicular network (e.g., connectivity, network conditions, and latency) and generates a routing decision for each vehicle, with the goal of optimizing certain performance objectives under average and peak constraints. For instance, the orange vehicles belong to an intra-cell cluster, and the blue vehicles belong to an inter-cell cluster. In the intra-cell cluster, the downlink data to v_0 is delivered through path $b_0 \to v_1 \to v_0$ for improving transmission rate since $SINR_{b_0,v_1}$ and $SINR_{v_1,v_0}$ along the path are both greater than $SINR_{b_0,v_0}$ of vehicle v_0. In the inter-cell cluster, v_a is connected by path $b_1 \to v_c \to v_a$ either for load balancing between cells or if the path has a smaller latency $d_{b_1,v_c} + d_{v_c,v_a}$ than that of the direct V2I link. The other vehicles are connected through direct V2I links. Under the above routing decisions, $n_{b_0} = n_{b_1} = 3$, and every vehicle gets $\frac{1}{3}$ MHz resource from the base station serving it.

B. Formulating the Routing Problem as CMDP

In this paper, we model the routing problem with both peak and average latency constraints as a CMDP with the goal of maximizing the long-term average utility of transmission rates.

Formally, the CMDP with finite state and action space can be described by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, r)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}$ is the transition between states, $\gamma \in (0, 1)$ is the discount factor, and $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function. The CMDP is a discrete decision process. Let π be the policy such that $\pi(a_t|s_t)$ specifies the probability of choosing action $a_t \in \mathcal{A}$ (i.e., routing decisions) in state $s_t \in \mathcal{S}$ (e.g., network statistics) at time slot t. The reward under pair

(s_t, a_t) is the received utility $r(s_t, a_t) = U(s_t, a_t)$. In this paper, we use the current state of the vehicular network as the input state of CMDP (e.g., current data rate, latency, and traffic routing), while the actions of CMDP are defined as routing decisions (e.g., the selection of relay node) for the next time slot. We focus on the logarithm utility that is shown to achieve proportional fairness [36], while our framework is general and can incorporate other utilities. The chosen utility function is used to define the rewards of the CMDP. More details of the policy model design is shown in Section V-A. The reward is computed by

$$r(s_t, a_t) = \sum_v \log(x_v(s_t, a_t) + \epsilon), \tag{3}$$

where ϵ is a small positive constant (e.g., 10^{-50} in the paper) for dealing with the extreme case where transmission rate $x_v(s_t, a_t) = 0$. The goal of the agent is to find a stationary policy π so as to maximize the cumulative discounted reward Φ, which is defined as

$$\Phi = \mathbf{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right].$$

Different from previous work, our CMDP is modeled under both peak and average latency constraints. In particular, the maximum transmission latency experienced by any vehicle at any time should satisfy a peak latency constraint. Formally,

$$\forall v, t : u_v - d_v(s_t, a_t) \geq 0, \tag{4}$$

where u_v is the upper latency bound of vehicle v. We define u_v as the *peak bound*. Besides, the long-term average transmission latency with respect to all vehicles should not exceed a given bound. Let $\bar{d}(s_t, a_t)$ be the average transmission latency of all the vehicles at time slot t. The average constraint is

$$\sum_{t \geq 0} \gamma^t (\bar{u} - \bar{d}(s_t, a_t)) \geq 0, \tag{5}$$

where $\bar{u}$ is the upper bound for $\bar{d}(s_t, a_t)$. Correspondingly, $\bar{u}$ is defined as the *average bound*.

Then, our CMDP problem can be formulated as

$$\text{PA-CMDP: } \max_{\pi} \quad \Phi, \quad \text{s.t. Eq. (4) and Eq. (5).}$$

For brevity, we refer to the CMDP problem with peak and average constraints as the PA-CMDP problem.

Solving the PA-CMDP problem is very challenging, and the existing RL algorithms cannot handle the existence of both peak and average constraints. In this paper, we propose a new RL algorithm with theoretical guarantees for the PA-CMDP problem in Sec. IV. Then, we turn the algorithm into a decentralized implementation in Sec. V.

IV. Constrained RL and Convergence Analysis

Our key idea is to approximate the solution to the PA-CMDP problem by solving a sequence of CSPs. First, the PA-CMDP problem with both peak and average constraints is converted into an optimization based on CSPs. Second, given a CSP, the peak constraints are absorbed into the average constraints through the use of constraint-sensitive functions, which yields a modified CSP with only average constraints. Next, the modified CSP is solved through an equivalent zero-sum Markov-bandit game using an extended Q-learning algorithm. Finally, we prove that the policy learned by the algorithm is optimal to the original CSP while both peak and average constraints can be satisfied.

A. Constraint Satisfaction Problem

Finding an optimal solution for the CMDP problem, including long-term average constraints is difficult [20]. We choose to transform the PA-CMDP problem into an optimization based on CSPs where the objective function is converted into an average constraint by setting a target objective value. By searching the best target objective value through some searching methods, the optimal solution can be approximated by solving a sequence of CSPs.

Firstly, the PA-CMDP problem can be generalized as:

$$\begin{aligned} \max_{\pi} \quad & \mathbf{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right] \\ s.t. \quad & f^i(s_t, a_t) \geq 0, \quad \forall t \geq 0, i \in [I], \qquad (6) \\ & \mathbf{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t h^j(s_t, a_t) \right] \geq 0, \quad \forall j \in [J], \qquad (7) \end{aligned}$$

where $[I] = \{1, 2, ..., I\}$ and $[J] = \{1, 2, ..., J\}$. So there are I peak constraints and J average constraints. The functions f^i and h^j return the values with respect to the pair (s_t, a_t). The peak constraints of Eq. (4) and the average constraint of Eq. (5) in the PA-CMDP problem can be easily formulated as the formation of Eq. (6) and Eq. (7), respectively.

The above problem can be converted to a CSP for finding a feasible policy π which (i) makes the objective value no smaller than a target value δ and (ii) satisfies both the peak and average constraints. Formally, the CSP can be shown as

$$\begin{aligned} \text{CSP:} \quad & \text{Find} \quad \pi \\ s.t. \quad & f^i(s_t, a_t) \geq 0, \quad \forall t \geq 0, i \in [I], \\ & \mathbf{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right] \geq \delta, \qquad (8) \\ & \mathbf{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t h^j(s_t, a_t) \right] \geq 0, \quad \forall j \in [J]. \qquad (9) \end{aligned}$$

The main difference from the generalized PA-CMDP problem is that the objective function is converted into an average constraint of Eq. (8). So, there are total of $1 + J$ average constraints in the CSP. It is worth noticing that the new average constraint of Eq. (8) has the same form of (9) if we define

$$h^0(s_t, a_t) = r(s_t, a_t) - (1 - \gamma)\delta.$$

Suppose the PA-CMDP problem is feasible, and let Φ^* be the optimal objective value of the PA-CMDP problem. For a CSP where $\delta = \Phi^*$, obviously, the optimal solution of the CSP is also an optimal solution to the PA-CMDP problem. However, Φ^* is not known in advance. We find that if the CSP is feasible for a given δ, a feasible solution can be found,

otherwise, there is no feasible solution. This allows us to approximate the optimal solution of the PA-CMDP problem by running an existing searching method (e.g, the Bisection method) for setting δ and solving a sequence of corresponding CSPs.[1] In the following parts of this section, we mainly show how to solve a CSP for a given δ.

B. Modified Constraint Satisfaction Problem

To deal with the peak constraints, we propose constraint-sensitive functions by folding the peak constraints into the average constraint functions, which is similar to [16]. Particularly, a penalty will be added to h^j ($j \in \{0\} \cup [J]$) whenever any peak constraints are violated. The constraint-sensitive functions denoted by $\widetilde{h}^j$ are expressed as

$$\widetilde{h}^j(s_t, a_t) = h^j(s_t, a_t) - \beta \cdot \mathbf{1}_{\sum_{i=1}^{I}[f^i]_- < 0}, \forall j \in \{0\} \cup [J], \tag{10}$$

where $[f^i]_-$ is defined as $[f^i]_- = \min\{0, f^i\}$ and $\mathbf{1}$ is the indicator function. When any peak constraints are violated, $\sum_{i=1}^{I}[f^i]_- < 0$ will hold and a penalty $-\beta$ ($\beta > 0$) will be punished to each h^j. Obviously, $\widetilde{h}^j = h^j$ when all the peak constraints are satisfied.

By using the constraint-sensitive functions, we transform the CSP into a modified CSP (M-CSP):

$$\text{M-CSP:} \quad \text{Find } \pi$$
$$s.t.\ \mathbf{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t \widetilde{h}^j(s_t, a_t)\right] \geq 0, \quad \forall j \in \{0\} \cup [J].$$

In the next subsection, we present the learning algorithm to solve the M-CSP optimally. Our convergence analysis will show that the learned optimal policy of the M-CSP is optimal to the CSP problem while both peak and average constraints can be satisfied.

C. Zero-Sum Markov-Bandit Game and Learning Algorithm

The solution of an M-CSP is obtained by solving an equivalent zero-sum Markov-Bandit game where the agent solves a Markov decision process problem and its opponent tackles a bandit optimization problem. Formally, the zero-sum Markov-Bandit game can be described by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathbb{P}, R)$, where $\mathcal{S}, \mathcal{A}$, and $\mathbb{P}$ are same as the definitions in our CMDP described in Sec. III-B. $\mathcal{O} = \{0\} \cup [J]$ is a finite action space for the agent's opponent, and the opponent action $o \in \mathcal{O}$. The reward function $R : \mathcal{S} \times \mathcal{A} \times \mathcal{O} \to \mathbb{R}$ is decided not only by the state and action but the opponent action. The reward function $R(s_t, a_t, o)$ is defined as

$$R(s_t, a_t, o) := \widetilde{h}^o(s_t, a_t), \quad \forall o = j \in \{0\} \cup [J]. \tag{11}$$

We define the value function with any stationary policy π as

$$V(s) = \min_{o \in \mathcal{O}} \mathbf{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t R(s_t, \pi(s_t), o)\right] \tag{12}$$

[1]A heuristic δ-search method will be proposed in Section V-A for the practical deployment of our framework in real networks.

Algorithm 1 Constrained Q-learning (CQL) Algorithm

Input: Learning rate $\alpha_t(s, a, o)$ for all $(s, a, o) \in \mathcal{S} \times \mathcal{A} \times \mathcal{O}$, Initial state s_0. **Output:** $Q(s_t, a_t, o_t)$

1: $Q(s, a, o) \leftarrow 0$; Initialize a_0 randomly;
2: **for** Iteration $t = 0, ...$ **do**
3: Take action a_t and observe next state s_{t+1};
4: $(\pi_{t+1}, o_t) = \arg\max_{\pi_{t+1}} \min_{o_t \in \mathcal{O}} Q(s_{t+1}, \pi_{t+1}(s_{t+1}), o_t)$;
5: $Q(s_t, a_t, o_t) \leftarrow (1 - \alpha_t(s_t, a_t, o_t))Q(s_t, a_t, o_t) + \alpha_t(s_t, a_t, o_t)[R(s_t, a_t, o_t) + \gamma\mathbf{E}[Q(s_{t+1}, \pi_{t+1}(s_{t+1}), o_t)]$;
6: Sample a_{t+1} from the distribution $\pi_{t+1}(\cdot|s_{t+1})$;

for the initial state $s_0 = s \in \mathcal{S}$. According to the definition of $R(s_t, a_t, o)$, maximizing $V(s)$ is actually maximizing the minimum margin of the average constraints in M-CSP. Here, the margin of an average constraint is defined as the left hand of the average constraint. So, for a feasible M-CSP, a policy π resulting in the largest $V(s)$ must be an optimal solution of the M-CSP.

Let $Q(s, a, o)$ be the Q-table (i.e., the state-action value function) which takes action a initially and continues with the policy π. Formally,

$$\begin{aligned} Q(s, a, o) &= R(s, a, o) + \mathbf{E}_\pi\left[\sum_{t=1}^{\infty}\gamma^t R(s_t, \pi(s_t), o)\right] \\ &= R(s, a, o) + \gamma\mathbf{E}_\pi[Q(s_+, \pi(s_+), o)], \end{aligned} \tag{13}$$

where s_+ denotes the new state after the execution of a under s. Let Q^* be the optimal Q-table resulting in the maximum value of $V(s)$. We can obtain the optimal policy π^* by solving the max-min problem modeled as

$$\pi^* = \arg\max_{\pi} \min_{o \in \mathcal{O}} \mathbf{E}\left[Q^*(s, \pi(s), o)\right]. \tag{14}$$

Under π^*, the maximum value of $V(s)$ can be achieved. The key to getting π^* lies in how to obtain Q^*.

The Constrained Q-learning (CQL) algorithm for learning Q^* is presented in Algorithm 1. The input includes $Q(s, a, o)$ with each value set by 0, the learning rates $\alpha_t(s, a, o)$, and the initial state s_0. Line 1 takes the initial action randomly. Line 3 takes action a_t and observes the next state s_{t+1}. Line 4 computes the policy π_{t+1} and the opponent action o_t by solving a max-min problem. Line 5 updates the Q-table. Finally, line 6 samples the next action a_{t+1} from the policy π_{t+1} gotten from line 4.

The following theorem shows that the optimal Q-table Q^* can be learned through Algorithm 1.

Theorem 1. *Suppose the Markov decision process (S, A, P) in the zero-sum Markov-Bandit game is unichain and that R is bounded. Let $\alpha_t(s, a, o)$ satisfy*

$$0 \leq \alpha_t(s, a, o) < 1, \quad \sum_{k=0}^{\infty}\alpha_t(s, a, o) = \infty,$$
$$\sum_{k=0}^{\infty}\alpha_t^2(s, a, o) < \infty, \quad \forall (s, a, o) \in S \times A \times O.$$

Algorithm 1 converges to Q^* with probability 1.

Theorem 1 can be proved following the same argument as Theorem 1 proposed in [20]. Briefly, the proof relies on the proposition that the random process $\{\Delta_t\}$ ($\{\Delta_t\}$ takes values in $\mathcal{R}$ and is defined as $\Delta_t(x) = (1-\alpha_t(x))\Delta_t(x) + \alpha_t(x)F_t(x)$) converges to zero with probability 1 under some assumptions on $\alpha_t(x)$, F_t, and Δ_t. Here, x denotes the tuple of (s, a, o) in our problem, and F_t is the sigma algebra $\sigma(\Delta_\tau, F_{\tau-1}, \alpha_{\tau-1}, \tau \leq t)$. Then, under our assumptions, $\Delta_t = Q_t - Q^*$ satisfies all the conditions of the proposition and thus converges to zero with probability 1, i.e., Q_t converges to Q^* with probability 1. Due to page limitations, we opt not to repeat the proof and refer the readers to [20] for more details.

According to Theorem 1, the optimal policy π^* of the zero-sum Markov-bandit game can be obtained by Eq. (14) and Eq. (13). Since $V(s)$, the minimum margin of the average constraints in M-CSP, is maximized under π^*, π^* is also an optimal policy of M-CSP.

D. Convergence Analysis

Now, we prove that π^* learned by Algorithm 1 is an optimal feasible solution to the CSP. First, we give a theorem about average constraints, which is shown as follows:

Theorem 2. *If the M-CSP is feasible for a given δ, π^* learned by Algorithm 1 is a feasible solution, and the average constraints in both the M-CSP and the corresponding CSP can be satisfied under π^*.*

Proof. If the M-CSP is feasible, then, by the definition of the M-CSP, there is a policy π such that for all $o \in \mathcal{O}$

$$\mathbf{E}\left[\sum_{t=0}^{\infty}\gamma^t R(s_t, \pi(s_t), o)\right] \geq 0,$$

which means that $V(s) = \min_{o\in\mathcal{O}} \mathbf{E}\left[\sum_{t=0}^{\infty}\gamma^t R(s_t, \pi(s_t), o)\right] \geq 0$. Since π^* achieves the maximum value of $V(s)$ according to Eq. (12) and Eq. (14), it is obvious that $V(s) \geq 0$ under π^* and the average constraints are all satisfied. According to the definition of $\widetilde{h}^j$, we have $h^j \geq \widetilde{h}^j$ ($\forall j \in \{0\} \cup [J]$). Thus, the average constraints in the CSP can also be satisfied when the average constraints in the corresponding M-CSP are fulfilled, which completes the proof. □

Next, we show the theoretical results of peak constraints. Let Π be the set of feasible policies for the M-CSP, and any policy $\pi \in \Pi$ satisfies the average constraints. Obviously, the optimal policy π^* learned by Algorithm 1 belongs to Π if the M-CSP is feasible. We define policy set $\Pi^+ \subseteq \Pi$, and any policy $\pi \in \Pi^+$ always generates actions that satisfy the peak constraints in the CSP. We also define policy set $\Pi^- \subseteq \Pi$, and any policy $\pi \in \Pi^-$ will output actions violating at least one peak constraint at some time. Obviously, $\Pi^+ \cup \Pi^- = \Pi$. We can get the following theorem, whose detailed proof is provided in Appendix A.

Theorem 3. *Suppose for a given δ, the CSP problem is feasible and the conditions in Theorem 1 are satisfied. Let $|h^j(s_t, a_t)| < u_r$ ($\forall j \in \{0\} \cup [J]$). If β satisfies*

$$\beta \geq \frac{2u_r}{1-\gamma}, \tag{15}$$

π^* for the M-CSP belongs to Π^+ such that the peak constraints of the CSP will always be satisfied under π^*.

Theorem 3 show that π^*, the optimal policy learned by Algorithm 1 for the M-CSP, is a feasible solution to the CSP (i.e., both the peak and average constraints are satisfied).

Next, we show that π^* also results in the optimal value function of the CSP. We define V^* as the optimal value function of the M-CSP under π^*. Similar to Eq. (12), we define the value function of the CSP as

$$\hat{V}(s) = \min_{j\in\{0\}\cup[J]} \mathbf{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t h^j(s_t, \pi(s_t))\right].$$

Let $\hat{V}^*$ be the optimal value function of the CSP with both the peak and average constraints satisfied. Then, we have the following theorem:

Theorem 4. *Suppose Theorem 3 holds. The optimal policy π^* for the M-CSP is also optimal for the CSP, and we have $\hat{V}^* = V^*$.*

Proof. Let π^*_{CSP} denote the CSP's optimal policy corresponding to $\hat{V}^*$. We have

$$\begin{aligned}\hat{V}^* &= \min_{j\in\{0\}\cup[J]} E_{\pi^*_{CSP}}\left[\sum_{t=0}^{\infty}\gamma^t h^j(s_t, a_t)\right] \\ &= \min_{o\in\{0\}\cup[J]} E_{\pi^*_{CSP}}\left[\sum_{t=0}^{\infty}\gamma^t R(s_t, a_t, o)\right] \\ &= V_{\pi^*_{CSP}} \leq V^*,\end{aligned}$$

where $V_{\pi^*_{CSP}}$ denotes the value function of M-CSP under π^*_{CSP}. The first and the third equalities hold according to the definition of $\hat{V}$ and V, respectively. The second equality holds because under π^*_{CSP}, all the peak constraints in the CSP are satisfied. The inequality holds as V^* is the optimal V.

Similarly, we can have

$$V^* = \hat{V}_{\pi^*} \leq \hat{V}^*.$$

where $\hat{V}_{\pi^*}$ is the value function of the CSP under the optimal policy π^* of the M-CSP. Combining the above two results, we have $\hat{V}^* \leq V^* \leq \hat{V}^*$. We conclude that $\hat{V}^* = V^*$, which completes the proof. □

According to Theorem 2, Theorem 3, and Theorem 4, the optimal policy of CSP can be obtained by Algorithm 1 while both the peak and average constraints are satisfied.

V. Decentralized Algorithm Implementation

In this section, we turn the algorithm proposed in Sec. IV into a decentralized implementation to improve the scalability of our framework in large-scale vehicular networks. Besides, we propose a multiple Q-table structure for effectively reducing the state and action space of the learning agents.

A. Distributed CQL Algorithm Implementation

Recall that vehicles can act as relays only for other vehicles within the same cluster. Therefore, the routing subproblems of individual clusters have independent action space, while the actions jointly determine the rewards received. We capitalize on the observation and propose a decentralized CQL (DCQL) algorithm. Particularly, we divide the routing problem of the entire network by modeling each cluster as an individual agent. Each agent focuses on optimizing the local objective and learns to satisfy the local peak and average constraints with respect to local vehicles. We design the state space, action space, and reward function with respect to the cluster-based agents carefully, which is critical to the success of the DCQL algorithm. Since the agents for intra-cell and inter-cell clusters have different impacts on cellular resource allocation of V2I communication, we develop the two kinds of agents differently. Our detailed designs are as follows:

State space: In our design, each vehicle has two states: using direct V2I link for data delivery and using a combination of V2I and V2V communication for data delivery. The state space of an agent is a combination of states of all the local vehicles, and we do not assume access to more fine-grained vehicle/network state information.

Action space: Agent actions decide whether each vehicle uses a relay and which relay is selected. Although a vehicle can establish V2V links with any of the local vehicles, not every local vehicle is suitable to be a relay. In real vehicular networks, whether a vehicle can be a candidate relay depends on many factors, such as channel quality (e.g., SINR), moving direction, and moving speed [26]. In our design, we consider choosing a fixed number of candidate relays for each cluster based on the SINR of both V2I links and V2V links. For intra-cell clusters, we choose the top $y\%$ (e.g., 80%) of potential relays from the local vehicles based on each vehicle's smallest SINR of its V2V links, and then find the first n_r potential vehicles with the largest V2I link SINR as candidate relays. For inter-cell clusters, we pick $n_r/2$ relays using the same way from the vehicles (of this cluster) located in each serving cell.[2] Such a method provides the opportunity of achieving load balancing and having inter-cell transmission paths among the cells serving this cluster. The setting of n_r introduces a tradeoff between potential network performance and the overhead of the learning algorithm. A larger n_r value provides more routing choices but induces larger action space. A larger action space usually causes the model to take more time to converge. In this paper, we set $n_r = 4$ because the marginal gain of using a higher n_r value quickly diminishes in our evaluation. That is, a larger n_r brings little performance improvement but at the cost of more convergence steps.

Reward function: Recall that the cellular resource available at each base station is evenly apportioned among all vehicles served by the base station (either directly or indirectly). It is not hard to see that the routing decisions made by intra-cell and inter-cell clusters have different impacts on cellular resource allocation – the actions taken by vehicles in intra-cell clusters do not affect how cellular resources are apportioned (as the traffic paths are always contained in the cell), while actions by inter-cell clusters affect the load at connected base stations (as the traffic paths may traverse multiple serving cells). Therefore, we develop different reward structures for the two kinds of clusters in order to permit a minimum level of necessary coordination for vehicles in inter-cell clusters. Particularly, for an intra-cell cluster c, we compute the reward of c's agent by using Eq. (3) directly but with respect to only local vehicles, i.e., $r_c^{local} = \sum_{v \in c} \log(x_v + \epsilon)$. In contrast, the agents for inter-cell clusters need to cooperate for load balancing among the relevant base stations. We propose a reward structure that allows inter-cell clusters c sharing the same cells to exchange local rewards r_c^{local} computed by Eq. (3) and to obtain their final reward $r_c^{cooperative}$ by calculating the average value of the received local rewards.

In the above distributed implementation, each agent only considers local peak and average constraints because they can only obtain local network information directly. Particularly, the constraint-sensitive function values of any cluster c will be computed by Eq. (10) with respect to local vehicles (also local constraints), which is similar to the process of calculating local rewards for c.

As described previously, we design to take a search method to find the optimal δ and solving a sequence of CSPs to approximate the optimal solution of the original routing problem. However, such a method may take much time to train the agents and thus limits the efficiency of our decentralized framework in real networks. It is expected that the learning algorithm can learn the optimal δ and the corresponding policy $\pi*$ self-adaptively. Actually, if we can obtain $E[r(s,a)]$, we can set $\delta = E[r(s,a)]/(1-\gamma)$ during the learning process, which can satisfy the above requirement. In practice, we take a heuristic δ-search algorithm by setting

$$\delta_t = \eta \cdot \delta_{t-1} + (1-\eta) \cdot r_t/(1-\gamma) \tag{16}$$

at each time step, where $\eta \in [0,1]$ is an adjustable parameter. Note that, each agent will estimate its own δ_t according to local rewards.

B. The Design of Q-table

We observe that the state and action spaces of each individual agent can still be large even when a decentralized implementation approach has been taken. More precisely, the sizes of state and action spaces increase exponentially with the size of clusters (i.e., # of the vehicles in clusters). Large state and action spaces will lead to the explosion of Q-table size, and in turn, have slow convergence for large-size problems.

To address the issue, we propose a Multiple Q-table (MQ-table) structure, which can efficiently reduce state and action spaces. Particularly, instead of using one Q-table in an agent, we maintain a Q-table for each individual vehicle in the corresponding cluster. Each Q-table only considers the state space and action space of the corresponding vehicle, so each Q-table requires a small space for storing values. As a result, the state and action spaces of each MQ-table increase linearly

[2]If less than $n_r/2$ vehicles of the cluster are located in a serving cell, we pick the maximum number of relays available while selecting the remaining relays evenly from other serving cells.

with the size of clusters. Reduced state and action spaces will lead to faster and better convergence of the learning algorithm.

The Q-tables in an agent are updated together using the same method shown in Algorithm 1, and the constraint-sensitive function values need to be computed with respect to local peak constraints for table updates. Recall that a penalty will be added to the constraint-sensitive function when any of the local peak constraints are violated. For each agent, the Q-table of a vehicle has to be updated using a small constraint-sensitive function value when any other vehicles in the same cluster violate the peak constraints regardless of the vehicle's own peak constraint. Since Q-tables have independent action space of individual vehicles, a wrong penalty may lead to some good actions not being chosen thereafter. To address the issue, we propose to compute the constraint-sensitive function values for updating Q-tables separately by considering only the peak violation of individual vehicles. More precisely, let h_c^j ($j \in \{0\} \cup [J]$) be the local constraint-sensitive function values of a cluster c. When updating the Q-table of vehicle v, we compute the constraint-sensitive function value $\widetilde{h}_{c,v}^j$ by

$$\widetilde{h}_{c,v}^j = h_c^j - \beta \cdot \mathbf{1}_{f_v<0},$$

where f_v denotes the peak constraint function of vehicle v. We can find the violation part only indicates the violation of v's own peak constraint.

VI. Simulations

We conduct numerical simulations with environment dynamics (e.g., dynamic link latency and cluster movement) to evaluate the performance of our proposed algorithm.

A. Simulation Setup

1) Network Generation: We consider a network with three cells (also three base stations). Any two of them are neighbors. The base stations' cellular resource available for V2I communication is 1 MHz, 2 MHz, and 3 MHz, respectively. We consider different amounts of cellular resources here for simulating unbalanced communication loads among base stations. For each V2V link, we assigned a fixed bandwidth resource, i.e., 10 kHz. By default, there are 50 clusters in the whole network. To simulate a dynamic environment for the vehicular network, we consider a number of sources of randomness and vary the configurations in real time during our experiment. The size of clusters follows a uniform distribution, i.e., $[15, 20]$. The SINR of V2I links and V2V links is uniformly distributed within $[0, 30]$ (dB) and $[15, 30]$ (dB), respectively. We choose four candidate relays as described previously. According to previous researches [37], [38], network latency usually follows a long-tail distribution. In our simulation, we leverage Log-Normal distribution Log-N(μ, σ) and uniformly distributed noise $U(\cdot, \cdot)$, to generate latencies of V2I links and V2V links and simulate latency dynamics. Particularly, at the beginning, we randomly set the initial latency of a V2I link and a V2V link to 10 + Log-N($\mu = 3.5$, $\sigma = 0.5$) (ms) and 10 + Log-N($\mu = 3.0$, $\sigma = 0.5$) (ms), respectively. Then, at each time slot, a V2I/V2V link latency is randomly set to d·(1 + $U(-10\%, 10\%)$) (ms) for simulating latency dynamics, where d indicates the initial link latency generated at the beginning.

Table I: The default settings of network generation.

Description	Setting
# of cells (base stations)	3, any two are neighbors
Cellular resource	1 MHz, 2 MHz, and 3 MHz, *resp.*
Resource of each V2V link	10 kHz
# of clusters	50
Size of clusters	uniform distribution $[15, 20]$
SINR of V2I links	uniform distribution $[0, 30]$ (dB)
SINR of V2V links	uniform distribution $[15, 30]$ (dB)
Ratio of intra-cell clusters	70%
# of candidate relays	4
Initial latency of V2I links	10 + Log-N($\mu = 3.5$, $\sigma = 0.5$) (ms)
Initial latency of V2V links	10 + Log-N($\mu = 3.0$, $\sigma = 0.5$) (ms)
Dynamic latency model	d·(1 + $U(-10\%, 10\%)$

Table II: The default settings of our algorithm.

Parameter	Setting
Peak bound u_v	identically 100 ms
Average bound $\bar{u}$	60 ms
T	10k
γ	0.5
α_t	$1 \cdot t^{-\frac{2}{3}}$
β	a large value satisfying Eq. (15)
η	0.999

Next, we need to scatter these clusters across the network. In our simulations, 70% of the generated clusters are intra-cell clusters, and the others are inter-cell clusters. An intra-cell cluster is located at each cell with identical probability, i.e., 1/3. Each inter-cell cluster is assumed to traverse two cells. Note that, there are three shared edges between any two hexagon cells. An inter-cell cluster is located at each border edge of two cells with identical probability, i.e., 1/3. Vehicles in an inter-cell cluster are located at each side of the border randomly, but there should be at least one vehicle at each side of the border. The default settings for generating the vehicular networks in our evaluation are listed in Table I. We note that the default settings of the environment in this paper are consistent with the real world vehicular network. For instance, the spectrum settings of base stations are similar to [10], and the vehicular network settings are similar to [9].

Further, we evaluate the effect of a different number of candidate relays on the performance of DCQL and show the results in Table. III. We note that a small set of candidate relays will limit the action space of the DCQL. Thus, the model may not be able to satisfy all constraints. However, if the set of candidate relays is too large, it may cause the model to be more difficult to converge, resulting in performance degradation. In this work, we set the number of candidate relays to 4 to balance the model convergence and optimality.

2) Algorithm Settings: The default settings of our algorithm can be found in Table II. To make agents converge quickly, we leverage ε-greedy for action explorations. We also discretize policies to accelerate the process of solving max-min problems in Algorithm 1 [20].

3) Baselines: We compare the distributed algorithm (i.e., DCQL) with five representative baselines.

- Direct-V2I: Downlink data is delivered to the destination vehicle through the directly connected V2I link.
- Max-SINR: Each vehicle chooses the vehicle with the

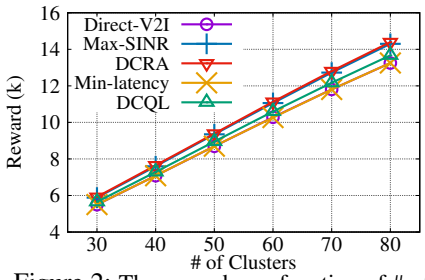

Figure 2: The reward as a function of # of clusters (the y-axis is actually in log scale due to the fairness utility).

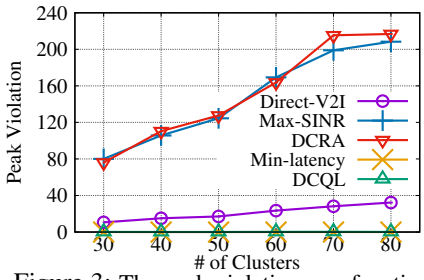

Figure 3: The peak violation as a function of the number of clusters.

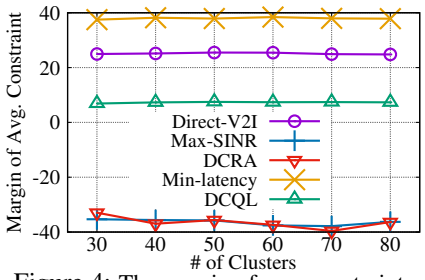

Figure 4: The margin of avg. constraint as a function of the number of clusters.

largest SINR in the same cluster as the relay. This routing algorithm provides a good transmission rate without considering latency.

- DCRA [2]: A state-of-the-art decentralized cooperative relaying algorithm where each cluster optimizes relay choices iteratively. The objective is to improve the transmission rate with the consideration of proportional fairness.
- Min-latency: Each vehicle chooses the relay that results in the smallest transmission latency.
- QL: A Q-learning based vehicular network routing algorithm without considering peak and average constraints.

We do not compare performance with existing constrained RL algorithms [16]–[23]. They focus on either peak or average constraints, but not both, in the paradigm of single-agent RL. It is not straightforward to extend them to a multi-agent implementation under both peak and average constraints for scalable vehicular network routing.

4) Metrics: We consider four metrics. First, we compare the reward defined in Eq. (3). Second, we evaluate the Geometric Mean of transmission rates (*GM of Rates*) with respect to all the vehicles in the network. We note that GM of Rates reflects the average transmission rate of a vehicle in the vehicular network. Since the reward function of Eq. (3) is the summation of the logarithm of each vehicle's rate (known as the proportional-fair utility function), maximizing this reward is equivalent to maximizing GM of Rates. Third, we compare *peak violation* which denotes the total number of violated peak constraints at the same time. The fourth metric is the left hand of the average constraint in Eq. (5), i.e., $\sum_{t\geq 0}\gamma^t(\bar{u}-\bar{d}(s_t,a_t))$, which is named as *the margin of average constraint*. We note that the margin of average constraint reflects the average latency of the vehicles in a period of time. More precisely, the margin of the average constraint equals to 2 times the difference between the average latency and the average latency constraint. It is estimated by an extra Q-table similar to the typical Q-learning algorithm. A non-negative margin value implies that the average constraint is satisfied. Note that, all numbers reported in our evaluation are the average of 10 independent simulation results.

B. Simulation Results

1) We first evaluate the effectiveness of our algorithm by varying the numbers of clusters: Fig. 2 shows the reward as a function of the number of clusters. We can see the reward values increase when there are more clusters in the network. This is because of the monotonicity of the reward function defined in Eq. (3) for a fixed amount of cellular resources. We hasten to point out that the reward improvement achieved by DCQL should be interpreted in the "decibel" sense. This is because of the use of log functions in the reward function, and thus the y-axis is actually in log scale. More detailed comparisons will be presented later through the analysis of breakdown numbers listed in Table IV.

Table III: The results of different settings of the number of candidate relays.

Number of Candidate Relays	2	4	6	8	10
Reward	9191	9147	9082	9033	9003
GM of Rates (Kbps)	46.04	43.74	40.53	38.28	36.97
Peak Violation	4	1	0	0	0
Margin of Avg. Constraint	7.65	7.19	9.76	10.61	11.12

Fig. 3 shows the peak violation as a function of the number of clusters. We can see that the peak violation of DCQL and Min-latency is zero for all the settings. Direct-V2I that uses direct V2I links still violates peak constraints since some V2I links may sometimes have a large latency. Max-SINR and DCRA achieve the worst performance. For example, when there are 80 clusters, the peak violation of Max-SINR and DCRA is around 220, implying that 15.7% of vehicles cannot meet their peak constraints.

Fig. 4 shows the value of average constraint as a function of the number of clusters. We can find that Direct-V2I, Min-latency, and DCQL get constraint margins larger than zero, and thus the average constraint is satisfied. In contrast, the average constraint is violated by Max-SINR and DCRA since they only focus on transmission data rate improvement and ignore latency constraints. Evaluation of DCQL with tighter latency constraints is provided in Appendix B.

2) We also evaluate the effectiveness of our algorithm by setting different sizes of clusters: The results are shown in Fig. 5-7. The x-axis is the range of the uniform distribution of the number of vehicles in a cluster. We can have similar observations to those in Fig. 2-4. Particularly, both peak and average constraints can be satisfied under DCQL.

3) We finally observe the convergence process under environment dynamics induced by cluster movement: At the beginning of the convergence test, the agents learn their policies from scratch. At the 10 thousandth update step, the environment changes, and the agents have to re-converge.

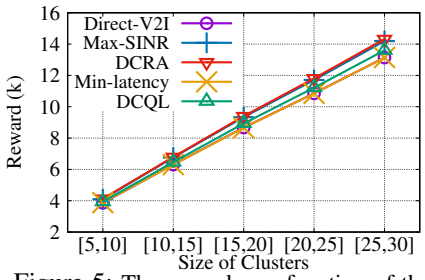

Figure 5: The reward as a function of the size of clusters (the y-axis is actually in log scale due to the fairness utility).

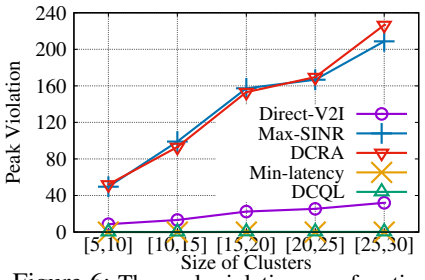

Figure 6: The peak violation as a function of the size of clusters.

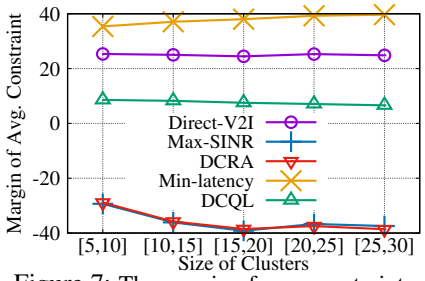

Figure 7: The margin of avg. constraint as a function of the size of clusters.

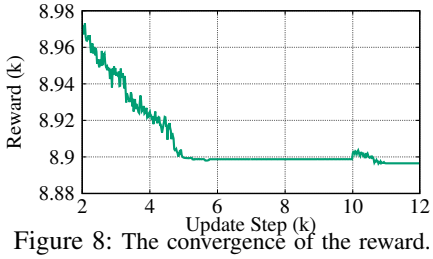

Figure 8: The convergence of the reward.

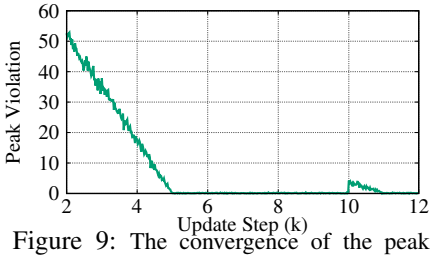

Figure 9: The convergence of the peak violation.

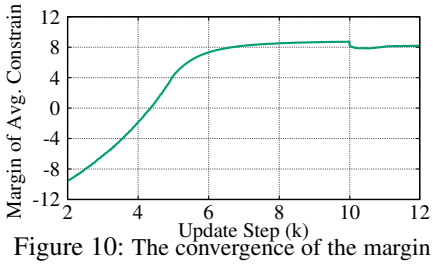

Figure 10: The convergence of the margin of avg. constraint.

When the environment dynamic happens, 10% of intra-cell clusters become inter-cell clusters randomly, and 10% of inter-cell clusters become intra-cell clusters randomly. Besides, the SINR values and link latencies in these clusters are regenerated. Fig. 8-10 show the convergence of reward, peak violation, and the margin of average constraint, respectively. We can find that all the three metrics can converge well within 5,000 steps when the agents are trained from scratch. Note that, the reward value decreases with the increment of update steps, which is because some rate performance has to be sacrificed for satisfying the constraints. After the environment changes, the agents can update policies quickly. Particularly, peak violation becomes zero within 1,000 steps, which is much faster than that in the initial learning process.

We also evaluate the time efficiency and storage efficiency of DCQL during the training process for future implementation in physical reality. We find that it takes 385.72s for the initial training process (10,000 steps) and 39.76s for the convergence after environment dynamics (1000 steps). We also note that when the environment changes, we suffer an acceptable performance degradation before the agents converge again. Moreover, the whole training process only requires 72MB of physical memory. We note that with the decentralized Q-table, one vehicle only needs to maintain a small Q-table. Such a decentralized Q-table design enables us to deploy our approach on vehicles with negligible memory cost.

VII. Conclusion

In this paper, we propose a holistic routing framework which maximizes the long-term average utility of transmission rates under both peak and average latency constraints. A new RL algorithm is proposed to solve the problem through an optimization based on Constraint Satisfaction Problems. We turn the algorithm into a decentralized implementation to improve the scalability of our framework. Simulation results show that our algorithm achieves much higher rewards than heuristic baselines while resulting in zero violation in terms of both peak and average constraints. We note that the proposed framework can be applicable to a wide range of routing problems and network utility optimization problems with different design objectives. For instance, integrating the proposed framework with cellular network optimization and vehicle mobility models will be considered in our future work.

Moreover, we consider reducing the cost of training and implementing our RL-based approach as future directions of this work, as well as cross-layer optimization of vehicular network. Using RL for network routing often requires training the algorithms with large data. There are separate lines of research training RL from small data and leveraging transfer learning techniques. We also believe a decentralized implementation would be needed to improve the scalability of the algorithm.

References

[1] H. Ye, G. Y. Li, and B.-H. F. Juang, "Deep reinforcement learning based resource allocation for V2V communications," *IEEE TVT*, vol. 68, no. 4, pp. 3163–3173, 2019.

[2] S. Kassir, G. de Veciana, N. Wang, X. Wang, and P. Palacharla, "Enhancing cellular performance via vehicular-based opportunistic relaying and load balancing," in *IEEE INFOCOM*, 2019.

[3] K. Liu, J. K. Ng, V. C. Lee, S. H. Son, and I. Stojmenovic, "Cooperative data scheduling in hybrid vehicular ad hoc networks: VANET as a software defined network," *IEEE/ACM TON*, vol. 24, no. 3, pp. 1759–1773, 2015.

[4] H. Saleet, R. Langar, K. Naik, R. Boutaba, A. Nayak, and N. Goel, "Intersection-based geographical routing protocol for VANETs: A proposal and analysis," *IEEE TVT*, vol. 60, no. 9, pp. 4560–4574, 2011.

[5] M. H. Eiza, T. Owens, and Q. Ni, "Secure and robust multi-constrained QoS aware routing algorithm for VANETs," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 1, pp. 32–45, 2015.

[6] G. Li, L. Boukhatem, and J. Wu, "Adaptive quality-of-service-based routing for vehicular ad hoc networks with ant colony optimization," *IEEE TVT*, vol. 66, no. 4, pp. 3249–3264, 2016.

[7] F. Castro, A. Martins, N. Capela, and S. Sargento, "Multihoming for uplink communications in vehicular networks," in *IEEE Wireless Days*, 2017, pp. 230–237.

[8] S. Corroy, L. Falconetti, and R. Mathar, "Dynamic cell association for downlink sum rate maximization in multi-cell heterogeneous networks," in *IEEE ICC*, 2012.

[9] L. Zhao, W. Zhao, A. Hawbani, A. Y. Al-Dubai, G. Min, A. Y. Zomaya, and C. Gong, "Novel online sequential learning-based adaptive routing for edge software-defined vehicular networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 5, pp. 2991–3004, 2020.

[10] O. S. Oubbati, N. Chaib, A. Lakas, and S. Bitam, "On-demand routing for urban vanets using cooperating uavs," in *2018 International Conference on Smart Communications in Network Technologies (SaCoNeT)*. IEEE, 2018, pp. 108–113.

[11] L. Yao, J. Wang, X. Wang, A. Chen, and Y. Wang, "V2X routing in a VANET based on the hidden Markov model," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 3, pp. 889–899, 2017.

[12] G. Xue, Y. Luo, J. Yu, and M. Li, "A novel vehicular location prediction based on mobility patterns for routing in urban VANET," *EURASIP Journal on Wireless Communications and Networking*, vol. 2012, no. 1, p. 222, 2012.

[13] F. Zeng, R. Zhang, X. Cheng, and L. Yang, "Channel prediction based scheduling for data dissemination in VANETs," *IEEE Communications Letters*, vol. 21, no. 6, pp. 1409–1412, 2017.

[14] Z. Li, C. Wang, and C.-J. Jiang, "User association for load balancing in vehicular networks: An online reinforcement learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 8, pp. 2217–2228, 2017.

[15] Z. Ahmed, S. Naz, and J. Ahmed, "Minimizing transmission delays in vehicular ad hoc networks by optimized placement of road-side unit," *Springer Wireless Networks*, pp. 1–10, 2020.

[16] Q. Bai, V. Aggarwal, and A. Gattami, "Provably efficient model-free algorithm for mdps with peak constraints," *Accepted to Journal of Machine Learning Research*, 2022.

[17] A. Gattami, "Reinforcement learning of Markov decision processes with peak constraints," *arXiv preprint arXiv:1901.07839*, 2019.

[18] P. Geibel, "Reinforcement learning for MDPs with constraints," in *Springer, European Conference on Machine Learning*, 2006.

[19] G. Karmakar, A. Kabra, and K. Ramamritham, "Coordinated scheduling of thermostatically controlled real-time systems under peak power constraint," in *IEEE RTAS*, 2013.

[20] A. Gattami, Q. Bai, and V. Aggarwal, "Reinforcement learning for constrained markov decision processes," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 2656–2664.

[21] Q. Bai, A. S. Bedi, M. Agarwal, A. Koppel, and V. Aggarwal, "Achieving zero constraint violation for constrained reinforcement learning via primal-dual approach," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 4, 2022, pp. 3682–3689.

[22] L. Prashanth and M. Ghavamzadeh, "Variance-constrained actor-critic algorithms for discounted and average reward MDPs," *Machine Learning*, vol. 105, no. 3, pp. 367–417, 2016.

[23] Y. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh, "A Lyapunov-based approach to safe reinforcement learning," in *NeurIPS*, 2018.

[24] M. Agarwal, Q. Bai, and V. Aggarwal, "Regret guarantees for model-based reinforcement learning with long-term average constraints," in *The 38th Conference on Uncertainty in Artificial Intelligence*, 2022.

[25] Q. Bai, A. S. Bedi, and V. Aggarwal, "Achieving zero constraint violation for constrained reinforcement learning via conservative natural policy gradient primal-dual algorithm," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[26] C. Cooper, D. Franklin, M. Ros, F. Safaei, and M. Abolhasan, "A comparative survey of VANET clustering techniques," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 657–681, 2016.

[27] S. Ucar, S. C. Ergen, and O. Ozkasap, "Multihop-cluster-based IEEE 802.11p and LTE hybrid architecture for VANET safety message dissemination," *IEEE TVT*, vol. 65, no. 4, pp. 2621–2636, 2015.

[28] H. Shan and W. Zhuang, "Multihop cooperative communication for vehicular ad hoc networks," in *IEEE CHINACOM*, 2011.

[29] M. A. Salahuddin, A. Al-Fuqaha, and M. Guizani, "Reinforcement learning for resource provisioning in the vehicular cloud," *IEEE Wireless Communications*, vol. 23, no. 4, pp. 128–135, 2016.

[30] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PloS one*, vol. 11, no. 6, p. e0155781, 2016.

[31] A. Taylor, S. Leblanc, and N. Japkowicz, "Anomaly detection in automobile control network data with long short-term memory networks," in *IEEE DSAA*, 2016.

[32] R. F. Atallah, C. M. Assi, and J. Y. Yu, "A reinforcement learning technique for optimizing downlink scheduling in an energy-limited vehicular network," *IEEE TVT*, vol. 66, no. 6, pp. 4592–4601, 2016.

[33] R. Atallah, C. Assi, and M. Khabbaz, "Deep reinforcement learning-based scheduling for roadside communication networks," in *IEEE WiOpt*, 2017.

[34] F. Li, X. Song, H. Chen, X. Li, and Y. Wang, "Hierarchical routing for vehicular ad hoc networks via reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1852–1865, 2018.

[35] K. L. K. Sudheera, M. Ma, and P. H. J. Chong, "Link stability based optimized routing framework for software defined vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 2934–2945, 2019.

[36] T. Lan, D. Kao, M. Chiang, and A. Sabharwal, "An axiomatic theory of fairness in network resource allocation." IEEE INFOCOM, 2010.

[37] N. Larson, D. Baltrunas, A. Kvalbein, A. Dhamdhere, K. Claffy, and A. Elmokashfi, "Investigating excessive delays in mobile broadband networks," in *ACM AllThingsCellular*, 2015.

[38] A. Kurian, "Latency analysis and reduction in a 4G network," 2018.

APPENDIX A
PROOF OF THEOREM 3

We consider a general decision process of the agent in the M-CSP. Let $t_0 \geq 0$ be a general time point during the decision process, and the actions taken before t_0 do not violate any peak constraints in CSP. So, $R(s_t, a_t, j) = \widetilde{h}^j(s_t, a_t) = h^j(s_t, a_t)$ when $t < t_0$. We denote the total discounted reward before the step t_0 as $V_{t<t_0} = \min_o \mathbf{E}[\sum_{t=0}^{t_0-1} \gamma^t R(s_t, a_t, o)]$. Particularly, $V_{t<t_0} = 0$ when $t_0 = 0$.

Next, we consider two cases. Case i): For $t \geq t_0$, the agent takes actions that always satisfy the peak constraints. Let V^+ be the total discounted reward of the M-CSP in the first case. Then we have

$$\begin{aligned} V^+ &\overset{(a)}{=} V_{t<t_0} + \min_o \mathbf{E}\Big[\sum_{t=t_0}^{\infty} \gamma^t R(s_t, a_t, o)\Big] \\ &\overset{(b)}{>} V_{t<t_0} + \sum_{t=t_0}^{\infty} \gamma^t (-u_r) \\ &= V_{t<t_0} - \gamma^{t_0} \frac{u_r}{1-\gamma}. \end{aligned}$$

where step (a) holds according to the definition of V and step (b) holds due to $R(s_t, a_t, j) = \widetilde{h}^j(s_t, a_t) = h^j(s_t, a_t) > -u_r$ for $j \in \{0\} \cup [J]$.

Case ii): For $t = t_0$, the agent takes an action that violates at least one of the peak constraints, and for $t > t_0$, the agent takes actions that may or may not violate the peak constraints. Let V^- be the total discounted reward of M-CSP in the second case. Then we have

$$\begin{aligned} V^- &= V_{t<t_0} + \min_o \mathbf{E}\Big[\sum_{t=t_0}^{\infty} \gamma^t R(s_t, a_t, o)\Big] \\ &\overset{(a)}{<} V_{t<t_0} + \gamma^{t_0}(u_r - \beta) + \sum_{t=t_0+1}^{\infty} \gamma^t u_r \\ &= V_{t<t_0} + \gamma^{t_0}(\frac{u_r}{1-\gamma} - \beta) \\ &\overset{(b)}{\leq} V_{t<t_0} - \gamma^{t_0} \frac{u_r}{1-\gamma} \end{aligned} \tag{17}$$

where step (a) holds because for $j \in \{0\} \cup [J]$, $R(s_{t_0}, a_{t_0}, j) = h^j(s_{t_0}, a_{t_0}) - \beta < u_r - \beta$ and $R(s_t, a_t, j) < u_r$ $(t > t_0)$. Step (b) comes from the requirement of β directly. The above equations obtain an upper bound of V^- when at least one peak constraint is violated at $t = t_0$.

Combining the above two cases, we have $V^+ > V^-$. Since $t_0 \geq 0$ is a general time point in the decision process of M-CSP, we conclude that for any given $\pi \in \Pi^-$ there always exists a $\pi \in \Pi^+$ that gets a larger total discounted reward value than the given $\pi \in \Pi^-$. Consider that π^* is the optimal policy of the M-CSP and results in the largest V. We have $\pi^* \in \Pi^+$. That is to say, the actions outputted by π^* always satisfy the peak constraints of the CSP, which completes the proof.

APPENDIX B
DCQL PERFORMANCE WITH TIGHTER LATENCY CONSTRAINTS

In this section, we will validate how DCQL performs when we consider tighter latency constraints, i.e., setting the peak bound and the average bound to relatively smaller values. By default, we set the peak bound and the average bound to 100 ms and 60 ms, respectively. Now, we consider two other settings of (peak bound, average bound): (100 ms, 55 ms) and (95 ms, 60 ms), respectively. Table IV shows some result numbers of our simulations. Our observations are as follows:

- Reward. Similar to the results in Fig. 2 and Fig. 5, all the approaches get similar rewards. DCQL has larger rewards than Direct-V2I and Min-latency but gets smaller rewards than DCRA, Max-SINR, and QL, because unlike DCRA, Max-SINR and QL, DCQL has to satisfy both latency constraints simultaneously. DCRA achieves the largest reward and particularly outperforms Max-SINR. Compared with Max-SINR, DCRA enables load balancing among base stations and results in better fairness. Besides, we can find that DCQL gets smaller rewards when we set a smaller peak bound (95 ms) or a smaller average bound (55 ms). This is because the agents have to sacrifice some rate performance to satisfy the constraints.
- GM of Rates. Although DCQL suffers some loss of transmission rate due to the latency constraints, it still improves the GM of rates by 43.6% and 44.4% under the default bound settings compared to Direct-V2I and Min-latency, respectively.
- Peak Violation. We get similar results to Fig. 3 and Fig. 6. DCQL and Min-latency satisfy peak constraints for all the three settings. Max-SINR, DCRA, and QL lead to very large peak violations. For example, the peak violation of Max-SINR and DCRA is around 173 when the peak bound is set to 95 ms, which means there are averagely 173 vehicles (out of 875 vehicles on average in total) whose transmission latency exceeds the required threshold.
- Margin of Avg. Constraint. The results are similar to those shown in Fig. 4 and Fig. 7. Direct-V2I, Min-latency, and DCQL satisfy the average constraint for all the three settings. However, DCQL gets the margin values much closer to zero compared with Direct-V2I and Min-latency, and thus DCQL does not lose too much performance for meeting the latency objectives. In contrast, Max-SINR, DCRA, and QL violate the average constraint all the time since they greedily choose the relays with the goal of improving data rate objective only.

Table IV: The results of different settings of the average bound and the peak bound.

Metric	Reward			GM of Rates (Kbps)			Peak Violation			Margin of Avg. Constraint		
Peak Bound (ms)	100	100	95	100	100	95	100	100	95	100	100	95
Avg. Bound (ms)	60	55	60	60	55	60	60	55	60	60	55	60
Direct-V2I	8,666			27.56			22	22	27	25.16	15.16	25.29
Max-SINR	9,297			58.08			147	144	181	-37.19	-47.01	-36.94
DCRA	9,378			63.76			140	137	175	-36.27	-46.07	-35.89
Min-latency	8,667			27.69			0	0	0	38.19	28.19	38.20
QL	9,076			44.67			108	109	139	-21.32	-31.04	-20.42
DCQL	8,924	8,890	8,907	37.35	35.86	36.60	0	0	0	7.89	1.48	10.38

* Results marked in green mean the constraint(s) is/are satisfied.