Dynamic RAN Slicing for Service-Oriented Vehicular Networks via Constrained Learning

Wen Wu, *Member, IEEE*, Nan Chen, *Member, IEEE*, Conghao Zhou, *Graduate Student Member, IEEE*, Mushu Li, *Graduate Student Member, IEEE*, Xuemin Shen, *Fellow, IEEE*, Weihua Zhuang, *Fellow, IEEE*, and Xu Li

***Abstract*—In this paper, we investigate a radio access network (RAN) slicing problem for Internet of vehicles (IoV) services with different quality of service (QoS) requirements, in which multiple logically-isolated slices are constructed on a common roadside network infrastructure. A dynamic RAN slicing framework is presented to dynamically allocate radio spectrum and computing resource, and distribute computation workloads for the slices. To obtain an optimal RAN slicing policy for accommodating the spatial-temporal dynamics of vehicle traffic density, we first formulate a constrained RAN slicing problem with the objective to minimize long-term system cost. This problem cannot be directly solved by traditional reinforcement learning (RL) algorithms due to complicated *coupled constraints* among decisions. Therefore, we decouple the problem into a resource allocation subproblem and a workload distribution subproblem, and propose a *two-layer constrained* RL algorithm, named Resource Allocation and Workload diStribution (RAWS) to solve them. Specifically, an *outer layer* first makes the resource allocation decision via an RL algorithm, and then an *inner layer* makes the workload distribution decision via an optimization subroutine. Extensive trace-driven simulations show that the RAWS effectively reduces the system cost while satisfying QoS requirements with a high probability, as compared with benchmarks.**

***Index Terms*—RAN slicing, constrained reinforcement learning, vehicular networks, coupled constraint, workload distribution.**

I. Introduction

RECENT technology advances for vehicular communication networks have laid a solid foundation for diverse Internet of vehicles (IoV) services, e.g., autonomous driving [1], [2]. It is predicted that autonomous vehicles will represent 25% of the automotive market, which is valued up to 77 billion US dollars by 2025 [3]. When a vehicle is on the road, a large number of computation-intensive tasks of IoV services are required to be processed. Processing such computation-intensive tasks by vehicles requires expensive on-board computing facilities and degrades fuel efficiency [4]. As a remedy to these limitations, a potential solution is to explore the multi-access edge computing (MEC) paradigm [5], [6], in which vehicles can offload these computation tasks to computation-powerful roadside radio access networks (RANs) for prompt processing.

IoV services are diversified with different quality of service (QoS) requirements. For example, a delay-sensitive cooperative sensing service for autonomous driving has a stringent delay requirement, e.g., 100 *ms* [5]; while a resource-hungry high definition (HD) map creation service is delay-tolerant; and a mobile video streaming service for vehicle users requires high throughput [7]. To support these diversified IoV services with different QoS requirements, the promising RAN slicing approach for vehicular networks has emerged, which aims at constructing multiple logically-isolated slices on the roadside network infrastructure [8].

In the literature, there exist some works investigating RAN slicing in the context of vehicular networks. Campolo *et al.* propose a conceptual RAN slicing framework to support various vehicle-to-everything services [9]. Another work presents a radio spectrum slicing scheme for roadside content caching services [10]. While the existing works focus on resource allocation among slices, computation workload distribution is yet to be considered in the RAN slicing. When base stations (BSs) are densely deployed along the road, vehicles are able to access the computing resources and offload their computation tasks to multiple BSs. Due to spatially uneven vehicle traffic density, some BSs may be overloaded. Vehicles in the coverage of the overloaded BSs can offload computation tasks to other underutilized BSs for better service performance, i.e., *workload distribution*. Hence, a comprehensive RAN slicing policy should jointly consider resource allocation and workload distribution.

Developing an optimal RAN slicing policy encounters many challenges. Firstly, the radio spectrum and computing resource allocation and workload distribution decisions are *coupled*. The resource allocation decision determines the resource availability at BSs, thus affecting the workload distribution decision. In turn, the workload distribution decision yields a new task assignment between vehicles and BSs, consequently affecting the resource allocation decision.

Manuscript received July 15, 2020; revised September 27, 2020; accepted October 27, 2020. Date of publication December 2, 2020; date of current version June 17, 2021. This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) and in part by Huawei Canada. *(Corresponding author: Mushu Li.)*

Wen Wu, Nan Chen, Conghao Zhou, Mushu Li, Xuemin Shen, and Weihua Zhuang are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: w77wu@uwaterloo.ca; n37chen@uwaterloo.ca; c89zhou@uwaterloo.ca; m475li@uwaterloo.ca; sshen@uwaterloo.ca; wzhuang@uwaterloo.ca).

Xu Li is with Huawei Technologies Canada Inc., Ottawa, ON K2K 3J1, Canada (e-mail: xu.lica@huawei.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JSAC.2020.3041405.

Digital Object Identifier 10.1109/JSAC.2020.3041405

Efficient RAN slicing policy depends on the interaction between workload distribution and resource allocation. Secondly, IoV services are highly heterogeneous in terms of QoS requirements, hence RAN slicing is required to satisfy multiple *constraints*. Thirdly, since the system operates in the presence of spatial-temporal dynamics of vehicle traffic density, maximizing long-term performance requires future information of vehicle traffic density. However, RAN slicing decisions have to be made without *a priori* information. To address the above challenges, reinforcement learning (RL) is a potential approach to make proactive decisions via learning traffic dynamics [11]. Directly applying traditional RL algorithms cannot solve the complicated RAN slicing problem with *coupled constraints*, since RL algorithms make the resource allocation and workload distribution decisions separately and may violate the coupled constraints. Hence, a tailored RL algorithm is required for the RAN slicing problem with coupled constraints.

In this paper, we present a dynamic RAN slicing framework for vehicular networks, in which workload distribution and resource allocation decisions are made for IoV services with different QoS requirements. Considering spatial-temporal variations of vehicle traffic density, RAN slicing is formulated as a stochastic optimization problem to minimize the long-term overall system cost while satisfying coupled constraints. To solve the problem, firstly, we decouple the problem into a resource allocation subproblem and a workload distribution subproblem. The resource allocation subproblem aims at minimizing the long-term system cost, which can be reformulated as a Markov decision process (MDP). The workload distribution subproblem is further decomposed into multiple one-shot optimization problems with the objective of minimizing instantaneous service delay, which are proved to be convex. Secondly, leveraging the properties of two subproblems, we propose a *two-layer constrained* RL algorithm, named Resource Allocation and Workload diStribution (RAWS). First, the *outer layer* of the RAWS makes the resource allocation decision via an RL algorithm, and then the *inner layer* makes the workload distribution decision via a convex optimization subroutine, thereby satisfying the coupled constraints. Extensive trace-driven simulation results demonstrate that the RAWS can effectively reduce the overall system cost, as compared with the state-of-the-art RL benchmarks including the deep deterministic policy gradient (DDPG) algorithm [12] and the twin delayed DDPG (TD3) algorithm [13]. Particularly, the RAWS can satisfy QoS constraints with a high probability, even in a heavy traffic scenario.

The main contributions of this paper are summarized as follows:

- We present a dynamic RAN slicing framework for vehicular networks to support multiple IoV services and balance BSs' workloads. We formulate the RAN slicing as a constrained stochastic optimization problem. The objective is to dynamically make resource allocation and workload distribution decisions to minimize the long-term overall system cost while satisfying coupled constraints and resource capacity constraints;
- We decouple the constrained stochastic optimization problem into a resource allocation subproblem and a workload distribution subproblem, and propose a two-layer constrained RL algorithm to solve it. We also design a softmax-based actor network within the algorithm to generate the resource allocation decision satisfying resource capacity constraints that the total amounts of allocated resources cannot exceed resource capacities at each BS.

The remainder of this paper is organized as follows. Related works are reviewed in Section II. The system model, the RAN slicing framework, and the problem formulation are presented in Section III. The two-layer RAN slicing algorithm is proposed in Section IV. Simulation results are given in Section V. Finally, Section VI concludes this research work.

II. Related Work

Network slicing can be divided into core network slicing and RAN slicing. Different from core network slicing which has been widely investigated, RAN slicing is still in its infancy, which has garnered much attention from both industry and academia recently. In the industry, the third generation partnership project (3GPP) devotes to standardizing RAN slicing in the fifth generation network [14]. Recent standardization efforts are discussed in [15]. In addition, several proof-of-concept RAN slicing systems have been developed, e.g., Orion [16]. In the academia, there are a number of research works investigating resource allocation problems in the RAN slicing. Ye *et al.* propose a communication resource slicing scheme for a two-tier cellular network [17], in which radio spectrum is allocated to a delay-oriented machine-to-machine service and a throughput-oriented data service. Xiao *et al.* study computing resource allocation among slices in a fog computing framework [18]. Li *et al.* propose a hierarchical soft RAN slicing framework to enable resource sharing among BSs, which can enhance resource multiplexing gain [19]. Another important line of works focuses on RAN slicing in vehicular networks. A RAN slicing architecture is proposed for supporting diverse IoV services in [20], in which a software defined networking (SDN) controller jointly orchestrates communication, caching and computing resources. Within this architecture, a soft RAN slicing framework is proposed in [10] to support multiple roadside caching services, in which network resources of slices can be opportunistically reused to enhance resource multiplexing gain. Different from the existing works that focus on resource allocation among slices, this work deals with the impact of spatially uneven vehicle traffic density on the system performance. We propose a workload distribution mechanism for BSs' workload balance.

Advanced machine learning algorithms have been widely applied in the network slicing, such as slice resource demand prediction [21], virtual network function deployment [22], and traffic flow scheduling [23]. Very recently, RL-based algorithms are applied to deal with complicated resource allocation problems in RAN slicing [24]. In [25], a deep RL algorithm is proposed to dynamically deploy virtual network functions by adjusting computing and storage resources. Xiang *et al.* study the problem of content caching and mode selection for multiple RAN slices, in which an RL algorithm is proposed

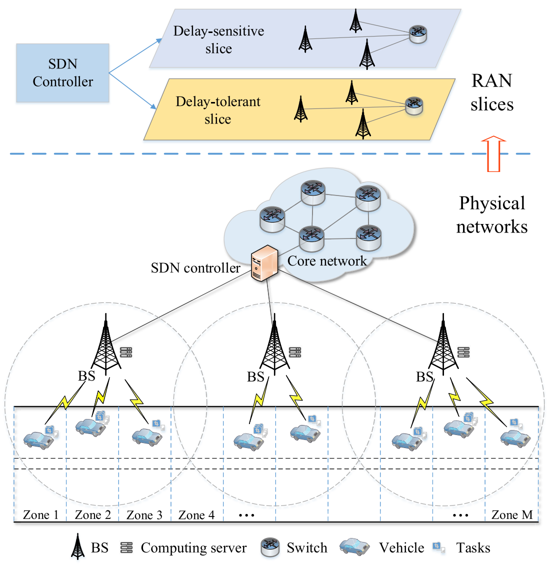


Fig. 1. Dynamic RAN slicing framework for vehicular networks to support different IoV services.

to address the challenges of time-varying channel and the unknown content popularity information [26]. Different from the existing studies, our work focuses on a *constrained* RAN slicing problem targeting at satisfying coupled constraints, which cannot be directly solved by traditional RL algorithms. We propose a novel two-layer constrained RL algorithm by leveraging the property of the considered problem.

III. System Model and Problem Formulation

In this section, we first establish the network model under consideration, and then present a dynamic RAN slicing framework for vehicular networks. Within the framework, we derive QoS constraints for the considered services. Finally, the overall system cost model is given to evaluate the performance of a RAN slicing policy.

A. Network Model

As shown in Fig. 1, we consider an MEC-enabled RAN along a road segment consisting of N BSs, each with a circular coverage of radius r. The set of BSs is denoted by $\mathcal{N}$. All the BSs are connected to an SDN controller, which is in charge of collecting network information and enforcing RAN slicing decisions. Each BS is equipped with a powerful computing server. Vehicles traversing the road segment can associate with a nearby BS and offload their computation tasks to BSs via wireless communication links. Then, BSs complete the computation tasks and send results to vehicles.

We adopt a zone-based model to characterize spatially uneven vehicle traffic density along the road. The considered road segment is divided into M zones with equal length L, and BSs can identify these zones according to their locations. The set of zones is denoted by $\mathcal{M}$. The vehicle density of these zones is denoted by $\{\rho_1, \rho_2, \ldots, \rho_M\}$, and the corresponding average vehicle velocity of these zones is denoted by $\{v_1, v_2, \ldots, v_M\}$. If a highway is considered, the relationship between the vehicle velocity and the vehicle density of a zone can be characterized via the fluid flow mobility model [27], i.e., $v_m = v_f\left(1 - \rho_m/\rho_{max}\right)$, where v_f and ρ_{max} denote the free flow speed and the maximum vehicle density of the road, respectively. Based on the availability of BSs, zones can be categorized into two types: (1) *non-overlapped zones* $\mathcal{M}_n$, which are covered by a single BS, e.g., Zone 2 in Fig. 1. Let binary variable $c_{m,n} \in \{0,1\}$ denote the association pattern, where $c_{m,n} = 1$ indicates that non-overlapped zone m is associated with BS n. Here, we consider the aggregated computation workload within each zone, since the RAN slicing framework operates based on macroscopic vehicle traffic [21]. The workload of zone m can only be distributed to BS n; and (2) *overlapped zones* $\mathcal{M}_o$, which are covered by multiple BSs, e.g., Zone 4 in Fig. 1. For the simplicity of analysis, vehicles in an overlapped zone can only associate with two nearest BSs. This assumption is reasonable since the channel condition deteriorates with the increase of connection distance. The association pattern is indicated by two binary variables, $a_{m,n} \in \{0,1\}$ and $b_{m,n'} \in \{0,1\}$, where $a_{m,n} = 1$ and $b_{m,n'} = 1$ denote that zone m is associated with BSs $\{n, n'\}$. The workload of the zone can be distributed to these two BSs. Let $\mathbf{B} \in \mathbb{R}^{|\mathcal{M}_o| \times K}$ denote the *workload distribution* decision. Each element, $\beta_{m,k} \in [0,1]$, represents the fraction of the workload of service k in zone m that is distributed to BS n. Correspondingly, the rest workload fraction, $1 - \beta_{m,k}$, is distributed to BS n'.

B. Dynamic RAN Slicing Framework

Two types of IoV services, denoted by $\mathcal{K} = \{u, e\}$, are considered. Here, $K = |\mathcal{K}| = 2$ represents the number of the considered services. One is a delay-sensitive service, denoted by service u, which has the maximum tolerable delay constraint. Taking the cooperative sensing service for example, the image captured by on-board cameras can be processed by BSs along with the sensing data from the roadside infrastructure. For road safety, the maximum tolerable delay of the cooperative sensing service is 100-150 *ms* [4]. The other service is delay-tolerant, denoted by service e. Taking the HD map creation service for example, vehicles collect and upload fresh HD map data, and then BSs detect the changes from the fresh data and update the HD map [5].

We present a dynamic RAN slicing framework to support the considered two services,[1] and mainly consider delay as the QoS metric of the considered services. As shown in Fig. 1, two slices are constructed: (1) a delay-sensitive slice, requires that service delay should be less than the maximum tolerable delay; and (2) a delay-tolerant slice, requires the stability of the queuing systems for computation tasks since they are offloaded and then processed. Detailed QoS constraints are presented for these two services in Subsection III-C.

The proposed RAN slicing framework operates in a time-slotted manner after the slices are constructed. Time is partitioned into multiple *slicing windows*, indexed by

[1]The presented RAN slicing framework can be extended to the case with multiple delay-sensitive and delay-tolerant services, in which increasing the number of services is to increase the dimension of decisions.

$t \in \mathcal{T} = \{1, 2, 3, \ldots, T\}$. Within a slicing window, RAN slicing decisions remain unchanged, and the vehicle traffic density is assumed to be stationary. In each slicing window, the SDN controller: (1) obtains the average vehicle density of all the zones in the current slicing window[2]; (2) makes workload distribution and resource allocation decisions; and (3) evaluates the system performance based on the feedback from BSs at the end of the slicing window. Specifically, the *resource allocation* decision includes the allocation of radio spectrum and computing resources for slices at all the BSs. The radio spectrum resource is allocated in a unit of subcarrier with bandwidth W. Let $S^t_{n,u}$ and $S^t_{n,e}$ denote the number of subcarriers allocated to the delay-sensitive and delay-tolerant services at BS n, respectively. The computing resource is allocated in a unit of virtual machine (VM) instance with central processing unit (CPU) frequency F [30]. Let $C^t_{n,u}$ and $C^t_{n,e}$ denote the number of VM instances allocated to the delay-sensitive and delay-tolerant services at BS n, respectively. In summary, RAN slicing decisions can be represented by the following matrices:

$$\begin{aligned}\mathbf{S}^t &= \left(S^t_{n,k} \in \mathbb{Z}^+ : n \in \mathcal{N}, k \in \mathcal{K}\right),\\ \mathbf{C}^t &= \left(C^t_{n,k} \in \mathbb{Z}^+ : n \in \mathcal{N}, k \in \mathcal{K}\right),\\ \mathbf{B}^t &= \left(\beta^t_{m,k} \in [0,1] : m \in \mathcal{M}_o, k \in \mathcal{K}\right), \end{aligned} \tag{1}$$

where $\mathbb{Z}^+$ represents the set of positive integers. Due to spatial-temporal variations of vehicle traffic density, the resource allocation and workload distribution decisions should be dynamically made in each slicing window.

C. QoS Constraints of Considered Services

In this subsection, we derive QoS constraints for the delay-sensitive and delay-tolerant services based on the queuing theory. A computation task is characterized by a three-parameter model $\{\xi_k, \eta_k, \lambda_k\}$, $\forall k \in \mathcal{K}$ [31], where ξ_k denotes the task data size (in bits), η_k denotes the average task computation intensity (in CPU cycles per task), and λ_k denotes the task arrival rate per vehicle. For notation simplicity, we omit t in $\beta^t_{m,k}$, ρ^t_m, $C^t_{n,k}$, and $S^t_{n,k}$ in this subsection.

1) Delay-Sensitive Service: The service delay consists of three parts: task offloading delay, task processing delay, and handover delay, as analyzed in the following.

Firstly, the task offloading delay represents the average time taken for offloading a task from a vehicle to a BS. The average transmission rate between BS n and the covered zones is denoted by R_n. Due to the stochastic wireless channel condition, the task transmission time from a vehicle to a BS is assumed to follow an exponential distribution with rate $S_{n,u}R_n/\xi_u$ [10]. Task arrivals from each vehicle are assumed to follow a Poisson process [31], and hence the aggregated task arrivals also follow a Poisson process. The rate of task arrivals at BS n is $\chi_{n,u} + \sum_{m\in\mathcal{M}_o} \psi_{m,n,u}\beta_{m,u}$, which consists of task arrivals from the non-overlapped zones and overlapped zones.

[2]The average vehicle density in the current slicing window can be predicted based on historical vehicle density information with high accuracy [28]. The vehicle density can be collected via many methods, such as using roadside loop detectors [29] or on-board global positioning system (GPS) sensors.

Here, $\chi_{n,u} = \sum_{m\in\mathcal{M}_n} c_{m,n}\lambda_u\rho_m L + \sum_{m\in\mathcal{M}_o} b_{m,n}\lambda_u\rho_m L$, and $\psi_{m,n,u} = (a_{m,n} - b_{m,n})\lambda_u\rho_m L$ are constants, where $\lambda_u\rho_m L$ denotes the workload of zone m of service u. The task offloading process can be modeled as an M/M/1 queue [32], and hence the average offloading delay at BS n is given by

$$D^o_{n,u} = \frac{1}{S_{n,u}R_n/\xi_u - \chi_{n,u} - \sum_{m\in\mathcal{M}_o}\psi_{m,n,u}\beta_{m,u}}, \forall n \in \mathcal{N}. \tag{2}$$

Constraint

$$S_{n,u} - \hat{\chi}_{n,s,u} - \kappa_{s,u}\sum_{m\in\mathcal{M}_o}\psi_{m,n,u}\beta_{m,u} \geq 0, \forall n \in \mathcal{N} \tag{3}$$

holds to ensure the stability of the task offloading queue at BS n, which also indicates radio spectrum allocation decision $S_{n,u}$ and workload distribution decision $\beta_{m,u}$ are coupled. In (3), $\kappa_{s,u} = \xi_u/R_n$ and $\hat{\chi}_{n,s,u} = \kappa_{s,u}\chi_{n,u}$ are constants.

Secondly, the task processing delay represents the average time of executing a task. The arrived tasks at the BS are placed in the computing queue until being processed. We assume the computation intensity per task follows an exponential distribution [6]. Hence, the departure process is a Poisson process with rate $C_{n,u}F/\eta_u$. Similar to the analysis in the task offloading delay, the task processing delay at BS n is given based on the M/M/1 queue theory, i.e.,

$$D^c_{n,u} = \frac{1}{C_{n,u}F/\eta_u - \chi_{n,u} - \sum_{m\in\mathcal{M}_o}\psi_{m,n,u}\beta_{m,u}}, \forall n \in \mathcal{N}, \tag{4}$$

where constraint

$$C_{n,u} - \hat{\chi}_{n,c,u} - \kappa_{c,u}\sum_{m\in\mathcal{M}_o}\psi_{m,n,u}\beta_{m,u} \geq 0, \forall n \in \mathcal{N} \tag{5}$$

holds to ensure the stability of processing queue at BS n. In (5), $\kappa_{c,u} = \eta_u/F$ and $\hat{\chi}_{n,c,u} = \kappa_{c,u}\chi_{n,u}$ are constants.

Thirdly, the average handover delay experienced by one task is defined as the ratio between the overall handover delay and the overall number of generated tasks of a vehicle in a road segment. The overall handover delay can be computed as the product of one-time handover delay and the number of handovers [33]. The number of handovers can be deemed as the number of BSs. The overall number of generated tasks is the product of the task arrival rate and the sojourn time that a vehicle drives through the road segment, which is given by $\lambda_u \sum_{m\in\mathcal{M}} L/v_m$. Hence, the average handover delay is

$$D^h_u = \frac{D_H N}{\lambda_u \sum_{m\in\mathcal{M}} L/v_m}, \tag{6}$$

where D_H denotes the one-time handover delay.

With (2), (4) and (6), taking different BSs' workloads into account, the average service delay is given by

$$\begin{aligned} D_u = D^h_u + \sum_{n\in\mathcal{N}} \frac{\chi_{n,u} + \sum_{m\in\mathcal{M}_o}\psi_{m,n,u}\beta_{m,u}}{\sum_{m\in\mathcal{M}}\lambda_u\rho_m L}\\ \times\left(D^o_{n,u} + D^c_{n,u}\right), \end{aligned} \tag{7}$$

where $\sum_{m\in\mathcal{M}}\lambda_u\rho_m L$ denotes the amount of all the zones' workloads . The service delay in slicing window t should be less than the maximum tolerable delay D^{th}, i.e., $D_u^t \leq D^{th}$. Similar to many existing works [6], [31], [34], we assume that the downlink latency is negligible in the service delay, since the data size representing output computation results is much smaller than that of input computation tasks in various applications.

2) Delay-Tolerant Service: The QoS constraint of the delay-tolerant service is to guarantee the queue stability, thereby avoiding queue overflow. The stability constraints are applied to the task offloading and task processing queues.

- Stability of task offloading queue: The analysis of the delay-tolerant service is similar to that of the delay-sensitive service in (3) based on the queuing theory. To guarantee the stability of the task offloading queue, constraint
$$S_{n,e} - \hat{\chi}_{n,s,e} - \kappa_{s,e} \sum_{m\in\mathcal{M}_o} \psi_{m,n,e}\beta_{m,e} \geq 0, \forall n \in \mathcal{N} \tag{8}$$
holds for BS n. In (8), $\kappa_{s,e} = \xi_e/R_n$, $\hat{\chi}_{n,s,e} = \kappa_{s,e}\chi_{n,e}$, $\chi_{n,e} = \sum_{m\in\mathcal{M}_n} c_{m,n}\lambda_e\rho_m L + \sum_{m\in\mathcal{M}_o} b_{m,n}\lambda_e\rho_m L$, and $\psi_{m,n,e} = (a_{m,n} - b_{m,n})\,\lambda_e\rho_m L$ are constants.
- Stability of task processing queue: Similar to the analysis for the delay-sensitive service in (5), the constraint on the allocated computing resource for the delay-tolerant service is given by
$$C_{n,e} - \hat{\chi}_{n,c,e} - \kappa_{c,e} \sum_{m\in\mathcal{M}_o} \psi_{m,n,e}\beta_{m,e} \geq 0, \forall n \in \mathcal{N}, \tag{9}$$
where $\kappa_{c,e} = \eta_e/F$ and $\hat{\chi}_{n,c,e} = \kappa_{c,e}\chi_{n,e}$ are constants.

D. Overall System Cost Model

The overall system cost is defined to evaluate the performance of a RAN slicing policy, which includes operation cost, slice reconfiguration cost, delay constraint violation cost, and system revenue in each slicing window.

1) Operation Cost: The operation cost refers to the cost of allocating subcarriers and VM instances for slices. Let U_o^t denote the operation cost in slicing window t, given by
$$U_o^t = \sum_{n\in\mathcal{N}}\sum_{k\in\mathcal{K}} \left(w_{o,s}S_{n,k}^t + w_{o,c}C_{n,k}^t\right). \tag{10}$$
In (10), $w_{o,s}$ and $w_{o,c}$ represent the unit costs of using a subcarrier and a VM instance, respectively, which should be set to be equivalent to account for radio spectrum and computing resource fairness.

2) Slice Reconfiguration Cost: When vehicle traffic density varies, the SDN controller may reconfigure slices and adjust the allocated network resources of slices, which renders the slice reconfiguration cost. Taking the computing resource adaption for example, VM adaption in practical systems (e.g., Docker [35]) involves booting a new VM instance or resizing an instantiated VM instance, which incurs additional delay for preparing resources [25] or brings hardware wear-and-tear [36]. Let U_r^t denote the slice reconfiguration cost in slicing window t, which measures the difference of resource allocation decisions across two subsequent slicing windows, i.e.,
$$U_r^t = \sum_{n\in\mathcal{N}}\sum_{k\in\mathcal{K}} \left(w_{r,s}\left[S_{n,k}^{t+1} - S_{n,k}^t\right]^+ + w_{r,c}\left[C_{n,k}^{t+1} - C_{n,k}^t\right]^+\right), \tag{11}$$
where function $\left[x^{t+1} - x^t\right]^+ = \max\{x^{t+1} - x^t, 0\}$ is applied to capture the amount of the increased resources at the BS when transiting from slicing window t to slicing window $t+1$. Note that the cost associated with reducing resource is omitted, since resource release can be completed quickly with negligible cost [36]. In (11), $w_{r,s}$ and $w_{r,c}$ represent the unit costs of subcarrier and VM instance reconfiguration, respectively, which should take larger values than that of resource usage in order to discourage frequent slice reconfiguration. Note that the reconfiguration of workload distribution decision does not incur system cost. This is because workload distribution decision represents the association pattern between zones' workloads and BSs, which does not consume physical resources.

3) Delay Constraint Violation Cost: The delay constraint violation cost refers to the penalty once the service delay exceeds the maximum tolerable delay constraint, i.e.,
$$U_q^t = w_v \mathbb{1}\{D_u^t > D^{th}\}, \tag{12}$$
where $\mathbb{1}\{x\}$ is an indicator function. If event x is true, $\mathbb{1}\{x\} = 1$; otherwise, $\mathbb{1}\{x\} = 0$. Here, w_v denotes the unit cost for delay constraint violation, which should take an extremely large value to penalize constraint violation.

4) System Revenue: The system revenue is related to the achieved service delay, which is given by
$$U_m^t = w_r\left[D^{th} - D_u^t\right]^+. \tag{13}$$
Here, $w_r > 0$ is the unit revenue, which should take a relatively large value to encourage low-latency services.

With (10)-(13), the overall system cost in slicing window t is given by
$$U^t = U_o^t + U_r^t + U_q^t - U_m^t. \tag{14}$$

Remark 1: The slice reconfiguration cost component belongs to *dynamic cost* that measures the performance of RAN slicing decisions across slicing windows, while the rest of cost components belong to *static cost* that measures the performance of RAN slicing decisions in the current slicing window.

E. Problem Formulation

Since vehicle traffic density varies spatially and temporally, minimizing the overall system cost in the long-term while satisfying QoS constraints is paramount, especially from the perspective of the network operator. Hence, the dynamic RAN slicing problem is formulated as follows:
$$\mathbf{P}_0: \min_{\{\mathbf{S}^t,\mathbf{C}^t,\mathbf{B}^t\}_{t\in\mathcal{T}}} \mathbb{E}\left[\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T} U^t\right]$$
$$\text{s.t.} \sum_{k\in\mathcal{K}} S_{n,k}^t \leq S_n^{\max}, \forall n \in \mathcal{N} \tag{15a}$$

$$\sum_{k\in\mathcal{K}} C_{n,k}^t \le C_n^{\max}, \forall n \in \mathcal{N} \tag{15b}$$

$$S_{n,k}^t, C_{n,k}^t \in \mathbb{Z}^+, \forall k \in \mathcal{K}, n \in \mathcal{N} \tag{15c}$$

$$0 \le \beta_{m,k}^t \le 1, \forall k \in \mathcal{K}, m \in \mathcal{M}_o \tag{15d}$$

$$(3), (5), (8), \text{and } (9),$$

where $S_n^{\max}$ and $C_n^{\max}$ denote the capacity of subcarriers and VM instances at BS n, respectively. The problem is to jointly allocate radio spectrum and computing resources, and distribute zones' workloads to minimize the average long-term system cost in an online manner. Constraints (15a) and (15b) are *resource capacity constraints*, which guarantee that the total amounts of allocated subcarriers and VM instances for all the services cannot exceed the capacity at each BS. Constraints (15c) and (15d) guarantee the feasibility of resource allocation and workload distribution decisions. Constraints (3), (5), (8) and (9) ensure the stability of queues for all the services, which indicates resource allocation and workload distribution decisions are coupled, i.e., *coupled constraints*.

Problem $\mathbf{P}_0$ belongs to stochastic optimization due to spatial-temporal variations of vehicle density. Due to the lack of future information, finding a global optimal solution via optimization methods is very difficult. To approach the optimal solution, one way is to use RL algorithms. Traditional RL algorithms can be applied to solve stochastic optimization problems with simple constraints, in which the space of decisions can be shaped separately according to their constraints. However, traditional RL algorithms make resource allocation and workload distribution decisions separately, which may violate the coupled constraints in problem $\mathbf{P}_0$. To solve the complicated problem with coupled constraints, we propose a two-layer constrained RL algorithm.

IV. Two-Layer Constrained RL Algorithm

In this section, we first decouple the problem into an outer-layer resource allocation subproblem and an inner-layer workload distribution subproblem, and then propose a two-layer constrained RL algorithm to solve them together.

A. Inner Layer: Workload Distribution Subproblem

The workload distribution subproblem is to minimize the long-term overall system cost via optimizing workload distribution decisions, i.e.,

$$\mathbf{P}_1: \min_{\{\mathbf{B}^t\}_{t\in\mathcal{T}}} \mathbb{E}\left[\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T} U^t\right]$$
$$\text{s.t. } (3), (5), (8), (9), \text{and } (15d).$$

According to the definition of the overall system cost in (14), workload distribution only impacts the static cost components, i.e., the delay bound violation cost and the system revenue, which are independent in each slicing window. Since the minimum value of the static cost components is achieved with the minimum service delay, the problem is to minimize the aggregated independent service delay in each slicing window. In addition, constraints are independent in each slicing window. Hence, optimizing long-term optimization problem $\mathbf{P}_1$ can be converted to individually optimizing multiple one-shot optimization problems, whose objectives are to minimize instantaneous service delay.

Moreover, each considered service has its own workload distribution decision variable, which can be optimized individually. For the delay-sensitive service, workload distribution variable $\beta_u^t = \{\beta_{m,u}^t\}_{m\in\mathcal{M}_o} \in \mathcal{R}^{|\mathcal{M}_o|\times 1}$ is optimized to minimize the average service delay in slicing window t, which yields the following one-shot optimization problem:

$$\mathbf{P}_{1,u}: \min_{\beta_u^t} D_u^t$$
$$\text{s.t. } (3), (5), \text{and } (15d).$$

Theorem 1: One-shot workload distribution problem $\mathbf{P}_{1,u}$ is a convex optimization problem.

Proof: Proof is provided in Appendix VI-A. ■

According to Theorem 1, if the feasible region of problem $\mathbf{P}_{1,u}$ is non-empty, denoted by $\mathcal{F}_u \neq \emptyset$, the optimal workload distribution decision for the delay-sensitive service can be directly obtained via convex optimization solvers, such as CVX [37] and Gurobi [38]. However, the optimization problem can be infeasible if the allocated resource is insufficient to satisfy queue stability constraints in (3) and (5). The infeasibility issue is solved by incorporating a penalty mechanism in the outer layer algorithm, which is detailed in Subsection IV-B.

Similarly, for the delay-tolerant service, workload distribution decision $\beta_e^t = \{\beta_{m,e}^t\}_{m\in\mathcal{M}_o}$ is optimized only to satisfy the queue stability constraints in (8) and (9). The corresponding problem, denoted by $\mathbf{P}_{1,e}$, is a simplified case of problem $\mathbf{P}_{1,u}$, which can also be solved by convex optimization solvers. The feasible region of problem $\mathbf{P}_{1,e}$ is denoted by $\mathcal{F}_e$. In summary, when the resource allocation decision is given and optimization problems are solvable, the optimal workload distribution decision can be directly made via a convex optimization subroutine.

Computational complexity analysis: In order to use convex optimization solvers, such as CVX, problem $\mathbf{P}_{1,u}$ is first transformed into a semi-definite programming (SDP) problem by introducing two sets of auxiliary variables. The computational complexity of solving the SDP problem by using the interior-point method is $\mathcal{O}\left(N_{ite}\,(3N-1)^{3.5}\right)$ [39], where N_{ite} represents the number of iterations before meeting the stopping criteria. Here, $3N-1$ is the number of variables in the transformed SDP problem, which includes workload distribution variable β_u^t of dimension $N-1$, and auxiliary variables of dimension $2N$.

B. Outer Layer: Resource Allocation Subproblem

The resource allocation subproblem is to allocate radio spectrum and computing resources with the objective of minimizing the average long-term system cost, i.e.,

$$\mathbf{P}_2: \min_{\{\mathbf{S}^t,\mathbf{C}^t\}_{t\in\mathcal{T}}} \mathbb{E}\left[\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T} U^t\right]$$
$$\text{s.t. } (15a), (15b), \text{and } (15c). \tag{18a}$$

The preceding problem is to design a resource allocation policy to minimize long-term system cost while satisfying constraints, which falls into the class of MDP with infinite horizon.

In the following, we reformulate the resource allocation subproblem as an MDP. Specifically, the SDN controller is modelled as an *agent*. In slicing window t, given observed *state* s^t, the SDN controller takes resource allocation *action* a^t. Then, the corresponding *reward* r^t is fed back from the environment, and the state evolves into new state s^{t+1} according to state transition probability $\Pr\left(s^{t+1}|s^t, a^t\right)$ [40]. In the MDP, the action, state, and reward are defined as follows.

- Action: Corresponding to the optimizing variables in problem $\mathbf{P}_2$, the action is the radio spectrum and computing resource allocation decisions, i.e.,

$$a^t = \left\{ \{S^t_{n,k}\}_{\substack{n\in\mathcal{N}\\k\in\mathcal{K}}}, \{C^t_{n,k}\}_{\substack{n\in\mathcal{N}\\k\in\mathcal{K}}} \right\}. \tag{19}$$

Each action element takes a positive integer value in order to satisfy constraint (15d).

- State: The system performance depends on the vehicle density of all the zones in the current slicing window $\{\rho^t_m\}_{m\in\mathcal{M}}$, and the resource allocation decision in the previous slicing window. Hence, the state is defined as

$$s^t = \left\{ \{\rho^t_m\}_{m\in\mathcal{M}}, \{S^{t-1}_{n,k}\}_{\substack{n\in\mathcal{N}\\k\in\mathcal{K}}}, \{C^{t-1}_{n,k}\}_{\substack{n\in\mathcal{N}\\k\in\mathcal{K}}} \right\}. \tag{20}$$

State space is $\left\{(\mathbb{R}^+)^M\right\} \times \left\{(\mathbb{Z}^+)^{KN}\right\} \times \left\{(\mathbb{Z}^+)^{KN}\right\}$, which is continuous.

- Reward: The following reward is defined to evaluate how good the action is under a state, i.e.,

$$r^t\left(s^t, a^t\right) = -\left(U^t \mathbb{1}\left\{\mathcal{F}_u \neq \emptyset\right\} + w_f \sum_{k\in\mathcal{K}} \mathbb{1}\left\{\mathcal{F}_k = \emptyset\right\} \right). \tag{21}$$

In (21), $w_f > 0$ is the penalty weight, which should take an extremely large value. The reward consists of two terms. The first term is consistent with the objective function in problem $\mathbf{P}_2$, which is achieved when the workload distribution subproblem for the delay-sensitive service is feasible. Obviously, the reward is related to the average service delay in the current slicing window. The second term, $w_f \sum_{k\in\mathcal{K}} \mathbb{1}\left\{\mathcal{F}_k = \emptyset\right\}$, is the penalty when the workload distribution subproblems are infeasible. Once the action (i.e., resource allocation decision) is determined, the workload distribution decision is made by solving the workload distribution subproblems in the inner layer. A poor resource allocation decision renders the infeasibility of the workload distribution subproblems, since insufficient network resources trigger the violation of queue stability constraints. To characterize the system performance in such a case, a penalty is incurred to discourage poor resource allocation decisions.

In the MDP setting, a resource allocation *policy* specifies how the SDN controller allocates radio spectrum and computing resources according to the current network information in each slicing window. Let Π denote the set of all the possible resource allocation policies. Our goal is to find the optimal resource allocation policy $\pi^\star \in \Pi$ that maximizes cumulative discounted reward within an infinite horizon of T slicing windows, i.e.,

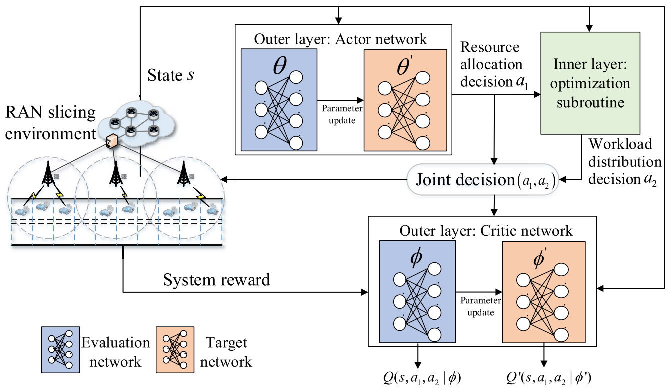

Outer layer: Actor network
θ
θ'
Parameter update
Resource allocation decision a_1
Inner layer: optimization subroutine
State s
RAN slicing environment
Workload distribution decision a_2
Joint decision (a_1, a_2)
Outer layer: Critic network
ϕ
ϕ'
System reward
Parameter update
Evaluation network
Target network
$Q(s,a_1,a_2\,|\,\phi)$
$Q'(s,a_1,a_2\,|\,\phi')$

Fig. 2. An illustration of the RAWS architecture. The actor network in the outer layer generates the resource allocation decision according to state s, and the optimization subroutine in the inner layer generates the workload distribution decision a_2 according to state s and resource allocation decision a_1.

$$\mathbf{P}'_2: \quad \max_{\pi\in\Pi} \mathbb{E}\left[\lim_{T\to\infty} \sum_{t=1}^{T} (\gamma)^t r^t\left(s^t, a^t\right) | \pi \right]$$
$$\text{s.t. (15a), and (15b),} \tag{22a}$$

where $\gamma \in (0,1)$ denotes the discount factor [40], and $(\gamma)^t$ denotes the t-th power of γ. The lacking of future information on the time-varying vehicle traffic density in vehicular networks results in the unknown state transition probability. In addition to the unknown state transition probability, continuous state space further makes problem $\mathbf{P}'_2$ unsolvable via conventional model-based algorithms, such as value iteration and policy iteration approaches [40]. Hence, a model-free RL algorithm which does not require state transition probability, can be applied to obtain an optimal resource allocation policy. The algorithm design is detailed in Subsection IV-C. Note that the objective function in problem $\mathbf{P}_2$ with the cumulative undiscounted reward can be well approximated by the objective function in problem $\mathbf{P}'_2$ with the cumulative discounted reward, when the discount factor approaches 1 [41], [42].

C. RAWS Algorithm

By leveraging the properties of two subproblems, we propose a two-layer constrained RL algorithm to solve problem $\mathbf{P}_0$, named RAWS, which is extended from the DDPG algorithm [12]. The two-layer architecture of RAWS is illustrated in Fig. 2. The *core idea* is that the outer layer and the inner layer make resource allocation and workload distribution decisions, respectively. Specifically, the actor network in the outer layer generates the resource allocation decision based on the current state, denoted by a_1, while the optimization subroutine in the inner layer generates the workload distribution decision based on the current state and the actor network's action, denoted by a_2. In this way, a joint decision (a_1, a_2) is obtained. The proposed decision making procedure complies with the properties of two subproblems. Then, a critic network

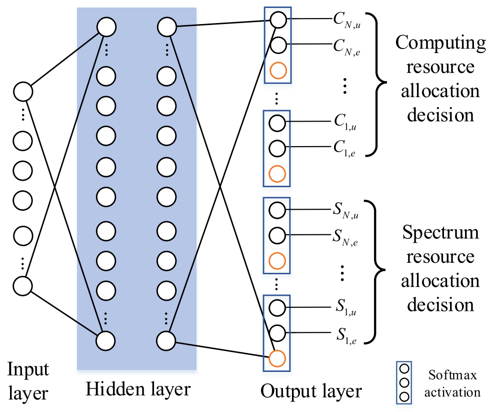


Fig. 3. Structure of the softmax-based actor network. Each tuple of three neurons in the output layer is activated by a softmax activation function, and the outputs of two of them are considered as the decisions.

evaluates the policy by estimating Q-values and guides the update of the actor network.

Within the RAWS, we also design a softmax-based actor network to generate resource allocation decisions satisfying the resource capacity constraints, whose structure is shown in Fig. 3. The *basic idea* is that the output layer of the actor network adopts the softmax function to activate a $(K+1)$-dimension vector, while only outputting its K elements as the decision that satisfying the resource capacity constraint. The softmax activation function can obtain a normalized output vector, i.e., $\sum_{i=1}^{K+1} x_i = 1$, where $\{x_i\}_{i=1,2,\ldots,K+1}$ is the output vector. Hence, the sum of its K elements satisfies inequality constraint $\sum_{i=1}^{K} x_i \leq 1$, which can be scaled up to resource capacity constraints, e.g., $\sum_{k=1}^{K} S_{n,k}^t \leq S_n^{\max}$.

In the following, we elaborate the RAWS in detail, which is given in Algorithm 1. In the initialization phase, actor evaluation network $\mu(s|\theta)$, and critic evaluation network $Q(s, a_1, a_2|\phi)$ are initialized, where θ and ϕ denote parameters of evaluation networks. The corresponding target actor and critic networks are denoted by $\mu'(s|\theta')$ and $Q'(s, a_1, a_2|\phi')$, respectively, whose parameters are θ' and ϕ'. The environment is reset with initialized state s^0. The RAWS operates in a discrete manner, which consists of two phases in each slicing window.

1. *Experience generation (Lines 5-12)*: The agent's experience is represented by a multi-dimensional tuple of the state transition and feedback reward, i.e., $\{s^t, a_1^t, a_2^t, r^t, s^{t+1}\}$, which is obtained via the following steps. The first step is to make the decisions. In slicing window t, the actor network makes resource allocation decision $a_1^t = \{\mathbf{S}^t, \mathbf{C}^t\}$ with respect to current state s^t, i.e.,

$$a_1^t = \mu(s^t|\theta) + \epsilon, \tag{23}$$

where $\epsilon = \mathcal{N}(0, \sigma^2)$ is added Gaussian noise for policy exploration. Then, the workload distribution decision $a_2^t = \mathbf{B}^t$ is made with respect to the current state and the resource allocation decision, by solving optimization problems $\{\mathbf{P}_{1,k}\}_{k\in\mathcal{K}}$ when they are feasible.[3] The second step is to store the experience. Once the joint decision is taken, reward

[3]The feasibility of optimization problems can be obtained via the status of optimization solvers.

Algorithm 1 Resource Allocation and Workload Distribution (RAWS) Algorithm

```
Input: Vehicle density of all the zones $\{\rho_m^t\}_{m\in\mathcal{M},t\in\mathcal{T}}$;
Output: Resource allocation and workload distribution
        decisions $\{\mathbf{S}^t, \mathbf{C}^t, \mathbf{B}^t\}_{t\in\mathcal{T}}$;
1  Initialization: Initialize critic and actor networks and the
   experience replay buffer;
2  for episode =1: $N_e$ do
3      Reset environment and obtain initial state $s^0$;
4      for slicing window t=1: $T$ do
5          ▷ Experience generation
6          Obtain resource allocation decision $a_1^t$ by (23);
7          if $\{\mathbf{P}_{1,k}\}_{k\in\mathcal{K}}$ are feasible then
8              Obtain workload distribution decision $a_2^t$ by
               solving optimization problems $\{\mathbf{P}_{1,k}\}_{k\in\mathcal{K}}$;
9          end
10         Execute joint decision $a^t = (a_1^t, a_2^t)$;
11         Observe reward $r^t$ and new state $s^{t+1}$;
12         Store transition $\{s^t, a_1^t, a_2^t, r^t, s^{t+1}\}$ in the
           experience replay buffer;
13         ▷ Neural network update
14         Sample a random minibatch of transitions from the
           experience replay buffer;
15         Update the critic evaluation network via minimzing
           the loss function in (24);
16         Update the actor evaluation network via the policy
           gradient in (25);
17         Update target networks by (26);
18     end
19 end
```

r^t is obtained from the environment, and the environment evolves to new state s^{t+1}. The corresponding transition tuple $\{s^t, a_1^t, a_2^t, r^t, s^{t+1}\}$ is stored in experience replay buffer B for training.

2. *Neural network update (Lines 13-17)*: In this phase, neural networks are updated offline based on the acquired experience, which is similar to that in DDPG. At the beginning, a mini-batch of N_m transitions are randomly chosen from the experience replay buffer as training data to update the parameters of neural networks. The detailed update procedure is given as follows.

Firstly, the critic evaluation network is updated by minimizing the loss function $L(\phi)$, which is defined as

$$L(\phi) = \frac{1}{N_m}\sum_{n=1}^{N_m}\left(y^n - Q\left(s^n, a_1^n, a_2^n|\phi\right)\right)^2. \tag{24}$$

In (24), $y^n = r^n + \gamma Q'\left(s^n, a_1^n, a_2^n|\phi'\right)$ is the update target obtained from the target networks.

Secondly, guided by the critic network, the actor evaluation network is updated via the following policy gradient (i.e., the gradient of policy's performance):

$$\nabla_\theta J \approx \frac{1}{N_m}\sum_{n=1}^{N_m} \nabla_{a_1} Q\left(s, a_1, a_2|\phi\right)|_{\substack{s=s^n \\ a_1=\mu(s^n|\theta)}} \nabla_\theta \mu\left(s^n|\theta\right). \tag{25}$$

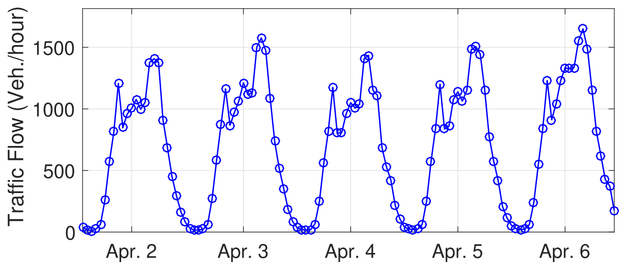


Fig. 4. Highway vehicle traffic flow trace.

Thirdly, target networks are updated by slowly tracking the evaluation networks, i.e.,

$$\begin{aligned}\theta' &= \tau\theta + (1-\tau)\theta',\\ \phi' &= \tau\phi + (1-\tau)\phi',\end{aligned} \tag{26}$$

where $\tau \in (0,1)$ denotes the update ratio of target networks.

Remark 2: Two unique designs are incorporated in the proposed RAWS algorithm: (1) the RAWS adopts a two-layer architecture, in which an actor network and an optimization subroutine are adopted to make joint decisions satisfying coupled constraints; and (2) the RAWS adopts a softmax-based actor network to generate resource allocation decisions satisfying resource capacity constraints. As such, the RAWS can reduce the long-term system cost while satisfying constraints.

V. Simulation Results

A. Simulation Setup

The performance of the proposed RAWS algorithm is evaluated by simulations based on two real-world vehicle traffic flow traces: (1) the highway trace, which is collected by Alberta Transportation.[4] The dataset records the hourly vehicle traffic flow on Highway 2A in Canada, over a period of three weeks from Apr. 2, 2020 to Apr. 22, 2020, as shown in Fig. 4. The vehicle traffic flow information is converted to vehicle density based on the Lighthill-Whitham-Richards (LWR) model [43]; and (2) the urban trace, which is collected by Didi Chuxing GAIA Initiative,[5] over a period of three weeks from Oct. 10, 2016 to Oct. 30, 2016. The dataset collects the trace of taxis that are equipped with GPS devices in urban areas of Xi'an, China. The vehicle's location information is collected every 2-4 seconds, which can be converted to the average vehicle density information.

For both traces, the duration of a slicing window is set to one hour. We consider a road segment with a length of 5 km, which is divided into 25 zones with equal length. Five BSs are deployed with a separation distance of 1 km, and the coverage radius of a BS is set to 800 m for service continuity, in order to cover the road segment. The channel path loss model for vehicular networks is given by PL(dB) $= 128.1 + 37.6\log_{10}(d)$, where d (in km) denotes the transmission distance between the vehicle and the BS [44]. The transmission power is set to 0.5 W. The two types of services are considered in the simulation. For the delay sensitive service, we consider the

[4]Alberta Transportation: http://www.transportation.alberta.ca/mapping/
[5]Didi Chuxing: https://gaia.didichuxing.com

TABLE I
Simulation Parameters

Parameter	Value	Parameter	Value
W	10 MHz	F	10 GHz
λ_e	$\{0.5, 1\}$ req/s	λ_u	$\{0.8 - 1.2\}$ req/s
D^{th}	100 *ms*	D_H	200 *ms*
ρ_{max}	120 veh/km	v_f	120 km/h
ξ_u, ξ_b	(0.6, 2) Mbit	η_u, η_b	(6, 2) $\times 10^8$ cycles
N_e	1000 episodes	$w_{o,s}, w_{o,c}$	1, 1
$w_{r,s}$, $w_{o,c}$	5, 5	w_v, w_r, w_f	200, 25, 200
$S_n^{\max}$	18	$C_n^{\max}$	18

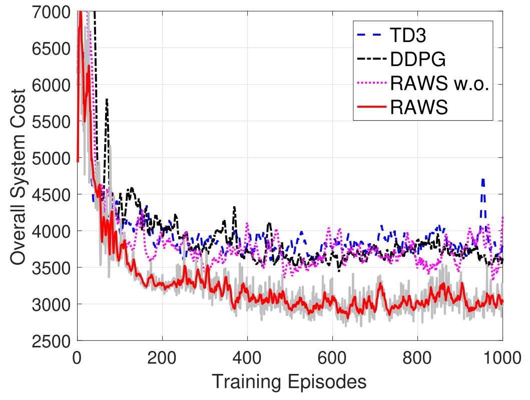
TABLE II
Parameters of RAWS

Parameters	Value	Parameters	Value
Actor learning rate	10^{-4}	Critic learning rate	10^{-3}
Actor hidden units	$(128, 64)$	Critic hidden units	$(128, 64)$
Actor output act.	Softmax	Hidden layer act.	ReLU
Optimizer	Adam	Replay buffer size	100,000
Policy noise (σ)	0.02	Target update ratio	0.005
Discount factor	0.75	Minibatch size	64

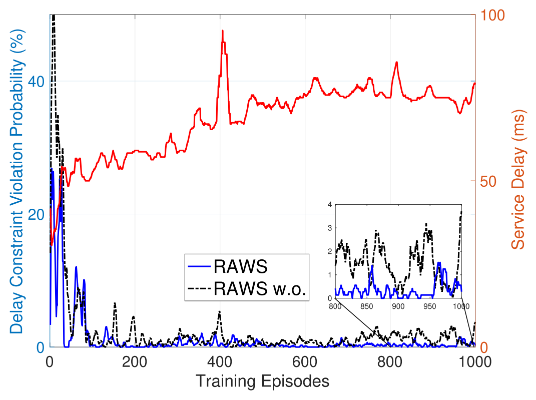
cooperative sensing service for autonomous driving, in which vehicles upload on-board video to BSs. The corresponding task data size is set to 0.6 Mbit, which is the data size of a one-second quarter common intermediate format (QCIF) video with 176×144 video resolution, 24.8 K pixels per frame and 25 frames per second [31]. The average computation intensity is 1,000 cycles/bit. After a task is processed, the result is fed back to vehicles. Hence, the maximum tolerable delay of the cooperative sensing service is set to 100 *ms* for safety consideration [4]. For the delay-tolerant service, we consider the HD map creation service. The task data size and average computation intensity are set to 2 Mbit and 100 cycles/bit, respectively. The task arrival rate is set to 1 request per second unless otherwise specified. The exemplary unit value setting for the operation cost, the slice reconfiguration cost, the delay constraint violation cost, and the system revenue is $\{1, 5, 200, 25\}$ based on the mentioned principle in Subsection III-D. Note that these values can be tuned based on the preference of the network operator. Important system parameters are listed in Table I. Parameters of the RAWS are listed in Table II.

We compare the RAWS with the following benchmarks:

1) **The DDPG [12] with decision shaping**: The decision shaping is performed to obtain feasible decisions. The resource allocation decision is shaped to satisfy the coupled constraints, while keeping the workload distribution decision unchanged. Taking the constraint in (3) for example, i.e., $S_{n,u} \geq \hat{\chi}_{n,s,u} + \kappa_{s,u}\sum_{m\in\mathcal{M}_o}\psi_{m,n,u}\beta_{m,u}$, radio spectrum allocation decision $S_{n,u}$ is complemented once it violates the bound on the right-hand side;
2) **The TD3 [13] with decision shaping**: Similar to the DDPG, the TD3 algorithm shapes decisions according to coupled constraints;
3) **The RAWS without workload distribution (RAWS w.o.)**: This algorithm is a simplified version of our



(a) Overall system cost



(b) Delay-sensitive service performance

Fig. 5. Convergence performance of all the learning-based algorithms for $\lambda_u = 1$.

proposed algorithm, in which the workload of an overlapped zone is equally distributed to two nearby BSs;

4) **Random**: This algorithm randomly selects feasible decisions in the decision space.

Note that the softmax-based actor network is applied in all the learning-based algorithms to satisfy the resource capacity constraints.

B. Performance Evaluation Over Highway Vehicle Traffic Flow Trace

The first week's data is used to train the RAWS, and then the rest of two weeks' data is used to evaluate its actual performance in terms of system cost, service delay, and QoS constraint violation probability. The measured performance is averaged over two weeks.

1) Convergence Performance: The convergence performance of the RAWS is shown in Fig. 5. Raw simulation points (e.g., grey curve) are processed by a five-point moving average to highlight the convergence trend (e.g., red curve). The number of training episodes represents the number of times that a algorithm has been trained. Firstly, the overall system cost with respect to training episodes is shown in Fig. 5(a). We observe a "bounded" overall system cost behavior of all the learning-based algorithms, which means that all of them have converged. More importantly, the RAWS achieves a lower overall system cost, as compared with all the benchmarks. The reason is that the RAWS is able to satisfy constraints without shaping decisions, while the DDPG and TD3 benchmarks do, thereby degrading the system performance. Secondly, the delay-sensitive service performance with respect to training episodes is shown in Fig. 5(b). On the one hand, the delay constraint violation probability quickly decreases from more than 20% to approximately 0.6% as the number of training episodes increases, which is lower than the benchmark. On the other hand, the red curve shows the average service delay also converges with respect to training episodes. Overall, the results indicate a good RAN slicing policy has been learned via interacting with a dynamic vehicular environment, which further verifies the convergence property of the RAWS.

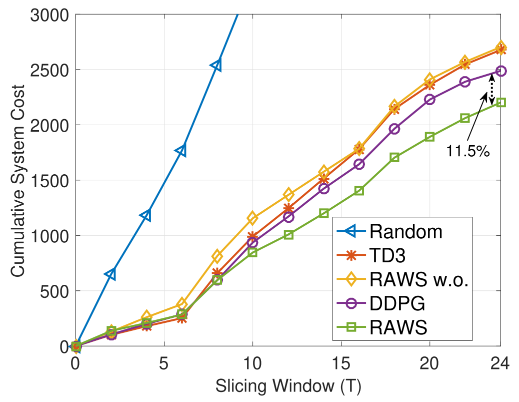
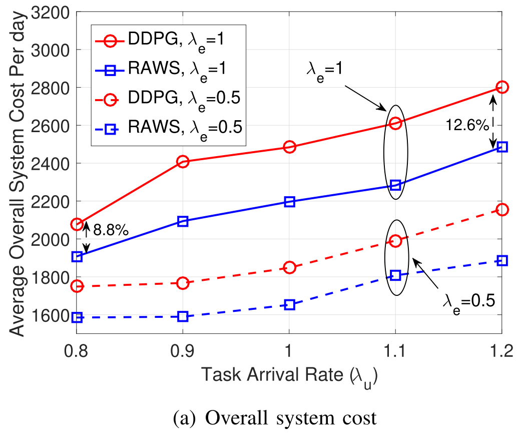


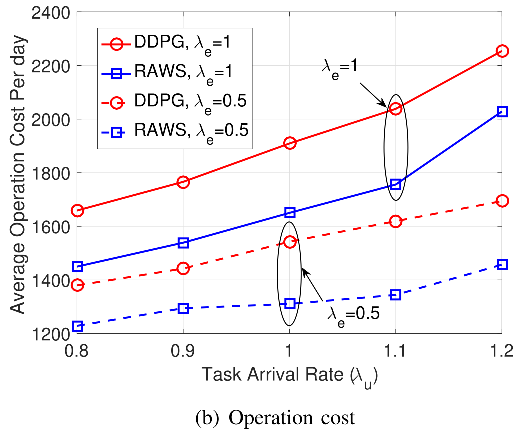
Fig. 6. Cumulative system cost within one day for $\lambda_u = 1$.

2) System Cost: When the learning-based algorithms are well-trained, we evaluate the cumulative overall system cost $\sum_{t=1}^{T} U^t$ within one day, as shown in Fig. 6. In the simulation, one day includes 24 slicing windows for the slicing window size of one hour. As expected, the RAWS incurs the lowest cost among all the algorithms. Specifically, the cumulative system cost incurred by the RAWS is approximately 11.5% less than that by the best benchmark, which validates its good performance in reducing system cost. In addition, the result shows that the performance gain increases with the number of slicing windows, implying that a high performance gain can be achieved for a large number of slicing windows.

As shown in Fig. 7(a), the average overall system cost per day of the different algorithms is compared with respect to the task arrival rate of the delay-sensitive service. Obviously, the average system cost increases with the increase of the task arrival rate, because more tasks consume more network resources. In addition, as compared with the benchmarks, performance gain achieved by the RAWS gradually increases with respect to the task arrival rate. The underlying reason is that the network resource utilization of the RAWS is higher than that of the benchmarks, especially in heavy traffic scenarios (e.g., $\lambda_u = 1.2$). In Fig. 7(b), average operation cost component with respect to the task arrival rate is evaluated. We can see that the proposed RAWS algorithm incurs a



(a) Overall system cost



(b) Operation cost

Fig. 7. Overall system cost and operation cost comparison with respect to the task arrival rate of the delay-sensitive service.

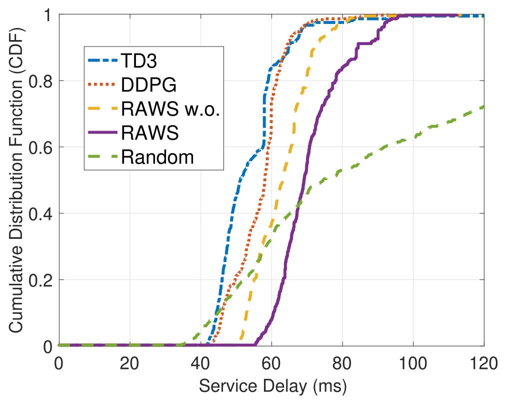


Fig. 8. Cumulative distribution function of service delay for $\lambda_u = 1$.

TABLE III

QoS Constraint Violation Probability With Different Task Arrival Rates

λ_u	RAWS	DDPG	TD3	RAWS w.o.	Random
0.9	0.28%	1.11%	1.11%	0.56%	31.67%
1.0	0.28%	0.56%	0.56%	1.67%	35.56%
1.1	0.28%	0.28%	0.28%	2.22%	38.06%
1.2	0.56%	0.56%	1.11%	3.89%	40.83%
Mean	**0.35%**	0.63%	0.76%	2.08%	36.53%

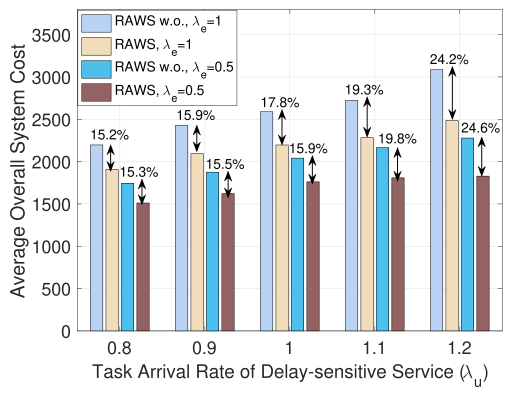


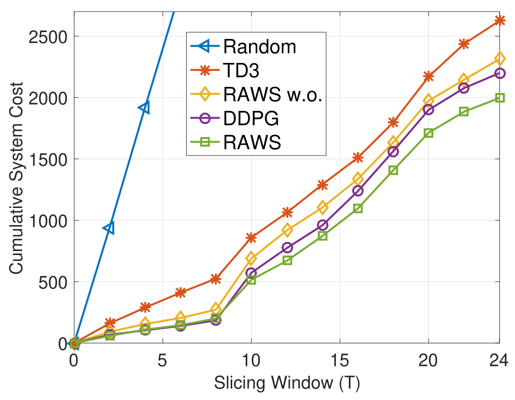
Fig. 9. Impact of workload distribution on the overall system cost with respect to task arrival rate.

lower operation cost. Specifically, average performance gain over different task arrival rates is about 13% for $\lambda_e = 1$, as compared with the DDPG benchmark.

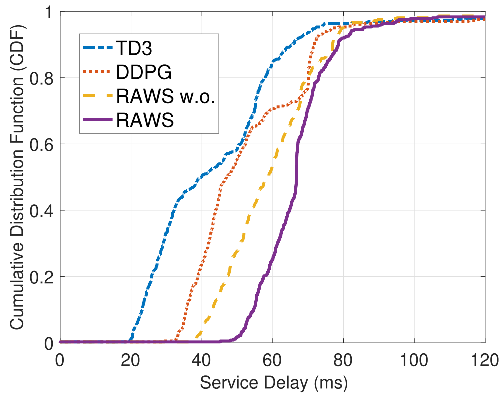
3) Service Delay: As shown in Fig. 8, the cumulative distribution functions of the service delay of the different algorithms are presented. Two important observations can be obtained from the simulation results. Firstly, the average service delay of the RAWS is about 70.9 *ms*, which is larger than that of the DDPG (56.8 *ms*) and TD3 (61.8 *ms*). The average service delay of the RAWS is closer to the maximum tolerable delay constraint (100 *ms*) than those of the benchmarks, as the network resource is more efficiently utilized in the RAWS under the constraint. Secondly, all the learning-based algorithms, including the DDPG, TD3 and RAWS, can effectively satisfy the service delay constraint. More importantly, the delay bound violation probability of the RAWS is very low (0.28%), validating the RAWS can satisfy the delay constraint with a high probability.

4) QoS Constraint Violation Probability: As listed in Table III, QoS constraint violation probabilities of different algorithms are compared for task arrival rates. A QoS constraint violation event is triggered by either exceeding the maximum tolerable delay for the delay-sensitive service or violating queue stability constraints for both services. It can be seen that the average violation probability of the RAWS over task arrival rates is 0.35%, which is lower than those of the benchmarks. It is interesting to note that, the RAWS without workload distribution performs even worse than the learning-based benchmarks (DDPG and TD3). The workload distribution mechanism provides more flexibility to the system. It balances spatially uneven workload among BSs by distributing vehicles' workloads from overloaded BSs to underutilized BSs, thereby reducing QoS constraint violation probability.

5) Impact of Workload Distribution: As shown in Fig. 9, the impact of workload distribution in terms of the system cost is evaluated. The RAWS can effectively reduce the overall system cost as compared with that without workload distribution. In addition, the performance gain increases with the increase of the task arrival rate. For example, when



(a) Cumulative system cost



(b) Service delay

Fig. 10. Cumulative system cost and service delay comparison among the different algorithms for $\lambda_u = 1$ over the urban vehicle traffic flow trace.

$\lambda_e = 1$, the performance gain increases from 15.2% for $\lambda_u = 0.8$ to 24.2% for $\lambda_u = 1.2$. This is because the proposed algorithm with workload distribution can better utilize network resources to reduce system cost than that without workload distribution, especially in a heavy traffic scenario.

C. Performance Evaluation Over Urban Vehicle Traffic Flow Trace

In addition to the highway vehicle traffic flow trace, we further evaluate the performance of the proposed algorithm over the urban vehicle traffic flow trace, in terms of the cumulative system cost and service delay. In Fig. 10(a), the cumulative system cost is presented with respect to the slicing window. Similar to the simulation results over the highway trace, the proposed RAWS algorithm achieves the minimum cumulative system cost among all the algorithms. Specifically, the RAWS reduces the cumulative system cost within one day by approximately 9.2% as compared with the best benchmark algorithm. In Fig. 10(b), the cumulative distribution functions of the service delay of different algorithms are compared, which shows that the RAWS algorithm can guarantee the service delay constraint (100 *ms*) with a very high probability.

VI. Conclusion

In this paper, we have presented a dynamic RAN slicing framework for vehicular networks to support diverse IoV services with different QoS requirements. We have proposed the RAWS, a two-layer constrained RL algorithm, to make the workload distribution and resource allocation decisions. Trace-driven simulation results have been provided to demonstrate that the proposed algorithm can effectively reduce the system cost, especially in the heavy traffic scenario, while satisfying QoS requirements with a high probability. The RAWS differs from traditional optimization-based methods, with the ability to adapt to time-varying vehicle traffic density without future information. In addition to RAN slicing, the design principle of the RAWS that integrates optimization and RL can be applied to other constrained stochastic optimization problems. For the future work, we will investigate the optimal slicing window size for RAN slicing. In addition, for the implementation in large-scale vehicular networks, we will develop a distributed and low-complexity learning based algorithm.

Appendix

A. Proof of Theorem 1

For notation simplicity, we omit t in the proof. With (2), (4) and (7), the objective function in problem $\mathbf{P}_{1,u}$ can be rewritten as

$$f(\beta_{m,u}) = D_u^h + \frac{1}{\sum_{m\in\mathcal{M}} \rho_m L\lambda_u} \cdot \sum_{n\in\mathcal{N}} \left(\frac{\omega_{n,s,u}}{\omega_{n,s,u} - \chi_{n,u} - \sum_{m\in\mathcal{M}_o} \psi_{m,n,u}\beta_{m,u}} + \frac{\omega_{n,c,u}}{\omega_{n,c,u} - \chi_{n,u} - \sum_{m\in\mathcal{M}_o} \psi_{m,n,u}\beta_{m,u}} - 2 \right), \tag{27}$$

where $\omega_{n,s,u} = S_{n,u}R_n/\xi_u$ and $\omega_{n,c,u} = C_{n,u}F/\eta_u$. The second-order derivative of the objective function is given by

$$\frac{\partial^2 f(\beta_{m,u})}{\partial^2 \beta_{m,u}} = \frac{1}{\sum_{m\in\mathcal{M}} \rho_m L\lambda_u} \cdot \sum_{n\in\mathcal{N}} \left(\frac{2\omega_{n,s,u}\psi_{m,n,u}^2\beta_{m,u}}{\left(\omega_{n,s,u} - \chi_{n,u} - \sum_{m\in\mathcal{M}_o} \psi_{m,n,u}\beta_{m,u}\right)^3} + \frac{2\omega_{n,c,u}\psi_{m,n,u}^2\beta_{m,u}}{\left(\omega_{n,c,u} - \chi_{n,u} - \sum_{m\in\mathcal{M}_o} \psi_{m,n,u}\beta_{m,u}\right)^3} \right). \tag{28}$$

Since $\beta_{m,u} \geq 0$, we have $\partial^2 f(\beta_{m,u})/\partial^2\beta_{m,u} \geq 0$ when the stability constraints in (3) and (5) are satisfied. As the constraints of the problem are linear, $\mathbf{P}_{1,u}$ is a convex optimization problem [45]. Hence, Theorem 1 is proved.

Acknowledgment

The authors would like to thank the undergraduate research assistant, Anita Hu, for conducting the simulations. They would also like to thank Jie Gao (Marquette University), Qihao Li, Jaya Rao, and Weisen Shi for many valuable suggestions throughout the work.

References

[1] L. Liang, H. Peng, G. Y. Li, and X. Shen, "Vehicular communications: A physical layer perspective," *IEEE Trans. Veh. Technol.*, vol. 66, no. 12, pp. 10647–10659, Dec. 2017.

[2] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected vehicles: Solutions and challenges," *IEEE Internet Things J.*, vol. 1, no. 4, pp. 289–299, Aug. 2014.

[3] Microsoft. (2018). *Global Autonomous Driving Market Outlook*. [Online]. Available: https://info.microsoft.com/ww-landing-Global-Autonomous-Driving-Market-Outlook-Report-eBook.html?lcid=en-us

[4] S.-C. Lin *et al.*, "The architectural implications of autonomous driving: Constraints and acceleration," in *Proc. 23rd Int. Conf. Archit. Support Program. Lang. Oper. Syst.*, Mar. 2018, pp. 751–766.

[5] J. Zhang and K. B. Letaief, "Mobile edge intelligence and computing for the Internet of vehicles," *Proc. IEEE*, vol. 108, no. 2, pp. 246–261, Feb. 2020.

[6] J. Xu, L. Chen, and P. Zhou, "Joint service caching and task offloading for mobile edge computing in dense networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2018, pp. 207–215.

[7] J. Qiao, Y. He, and X. S. Shen, "Improving video streaming quality in 5G enabled vehicular networks," *IEEE Wireless Commun.*, vol. 25, no. 2, pp. 133–139, Apr. 2018.

[8] X. Shen *et al.*, "AI-assisted network-slicing based next-generation wireless networks," *IEEE Open J. Veh. Technol.*, vol. 1, pp. 45–66, Jan. 2020.

[9] C. Campolo, A. Molinaro, A. Iera, and F. Menichella, "5G network slicing for Vehicle-to-Everything services," *IEEE Wireless Commun.*, vol. 24, no. 6, pp. 38–45, Dec. 2017.

[10] S. Zhang, H. Luo, J. Li, W. Shi, and X. Shen, "Hierarchical soft slicing to meet multi-dimensional QoS demand in cache-enabled vehicular networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2150–2162, Mar. 2020.

[11] M. Li, J. Gao, L. Zhao, and X. Shen, "Deep reinforcement learning for collaborative edge computing in vehicular networks," *IEEE Trans. Cognit. Commun. Netw.*, early access, Jun. 17, 2020, doi: 10.1109/TCCN.2020.3003036.

[12] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," in *Proc. ICLR*, 2016, pp. 1–14.

[13] S. Fujimoto, H. Van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. ICML*, 2018, pp. 1587–1596.

[14] *Telecommunication Management; Study on Management and Orchestration of Network Slicing for Next Generation Network (Release 15)*, document 3GPP TR 28.801 V2.0.1, Sep. 2017.

[15] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2429–2453, 3rd Quart., 2018.

[16] X. Foukas, M. K. Marina, and K. Kontovasilis, "Orion: RAN slicing for a flexible and cost-effective multi-service mobile network architecture," in *Proc. 23rd Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2017, pp. 127–140.

[17] Q. Ye, W. Zhuang, S. Zhang, A.-L. Jin, X. Shen, and X. Li, "Dynamic radio resource slicing for a two-tier heterogeneous wireless network," *IEEE Trans. Veh. Technol.*, vol. 67, no. 10, pp. 9896–9910, Oct. 2018.

[18] Y. Xiao and M. Krunz, "Dynamic network slicing for scalable fog computing systems with energy harvesting," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 12, pp. 2640–2654, Dec. 2018.

[19] J. Li *et al.*, "A hierarchical soft RAN slicing framework for differentiated service provisioning," *IEEE Wireless Commun.*, early access, Apr. 22, 2020, doi: 10.1109/MWC.001.2000010.

[20] W. Zhuang, Q. Ye, F. Lyu, N. Cheng, and J. Ren, "SDN/NFV-empowered future IoV with enhanced communication, computing, and caching," *Proc. IEEE*, vol. 108, no. 2, pp. 274–291, Feb. 2020.

[21] C. Gutterman, E. Grinshpun, S. Sharma, and G. Zussman, "RAN resource usage prediction for a 5G slice broker," in *Proc. 20th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, Jul. 2019, pp. 231–240.

[22] J. Pei, P. Hong, M. Pan, J. Liu, and J. Zhou, "Optimal VNF placement via deep reinforcement learning in SDN/NFV-enabled networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 263–278, Feb. 2020.

[23] L. Gu, D. Zeng, W. Li, S. Guo, A. Y. Zomaya, and H. Jin, "Intelligent VNF orchestration and flow scheduling via model-assisted deep reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 279–291, Feb. 2020.

[24] X. Chen *et al.*, "Multi-tenant cross-slice resource orchestration: A deep reinforcement learning approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2377–2392, Oct. 2019.

[25] J. S. Pujol Roig, D. M. Gutierrez-Estevez, and D. Gunduz, "Management and orchestration of virtual network functions via deep reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 304–317, Feb. 2020.

[26] H. Xiang, S. Yan, and M. Peng, "A realization of fog-RAN slicing via deep reinforcement learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 4, pp. 2515–2527, Apr. 2020.

[27] W. L. Tan, W. C. Lau, O. Yue, and T. H. Hui, "Analytical models and performance evaluation of drive-thru Internet systems," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 1, pp. 207–222, Jan. 2011.

[28] L. Han, K. Zheng, L. Zhao, X. Wang, and X. Shen, "Short-term traffic prediction based on deepcluster in large-scale road networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 12, pp. 12301–12313, Dec. 2019.

[29] C. Chen, K. Petty, A. Skabardonis, P. Varaiya, and Z. Jia, "Freeway performance measurement system: Mining loop detector data," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1748, no. 1, pp. 96–102, Jan. 2001.

[30] M. Ghaznavi, A. Khan, N. Shahriar, K. Alsubhi, R. Ahmed, and R. Boutaba, "Elastic virtual network function placement," in *Proc. IEEE 4th Int. Conf. Cloud Netw. (CloudNet)*, Oct. 2015, pp. 255–260.

[31] Y. Sun, S. Zhou, and J. Xu, "EMM: Energy-aware mobility management for mobile edge computing in ultra dense networks," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2637–2646, Nov. 2017.

[32] X. Ma, A. Zhou, S. Zhang, and S. Wang, "Cooperative service caching and workload scheduling in mobile edge computing," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Jul. 2020, pp. 2076–2085.

[33] H. Ibrahim, H. ElSawy, U. T. Nguyen, and M.-S. Alouini, "Mobility-aware modeling and analysis of dense cellular networks with C-plane/U-plane split architecture," *IEEE Trans. Commun.*, vol. 64, no. 11, pp. 4879–4894, Nov. 2016.

[34] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[35] D. Bernstein, "Containers and cloud: From LXC to docker to kubernetes," *IEEE Cloud Comput.*, vol. 1, no. 3, pp. 81–84, Sep. 2014.

[36] L. Wang, L. Jiao, J. Li, and M. Muhlhauser, "Online resource allocation for arbitrary user mobility in distributed edge clouds," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2017, pp. 1281–1290.

[37] M. Grant and S. Boyd. (2014). *CVX: MATLAB Software for Disciplined Convex Programming, Version 2.1*. [Online]. Available: http://cvxr.com/cvx

[38] GUROBI OPTIMIZATION. (2014). *Inc. GUROBI Optimizer Reference Manual, 2015*. [Online]. Available: http://www.gurobi.com

[39] Y. Ye, *Interior Point Algorithms: Theory and Analysis*, vol. 44. Hoboken, NJ, USA: Wiley, 2011.

[40] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[41] X. Chen, Z. Han, H. Zhang, G. Xue, Y. Xiao, and M. Bennis, "Wireless resource scheduling in virtualized radio access networks using stochastic learning," *IEEE Trans. Mobile Comput.*, vol. 17, no. 4, pp. 961–974, Apr. 2018.

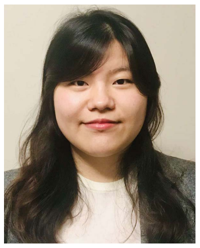
[42] H. He, H. Shan, A. Huang, Q. Ye, and W. Zhuang, "Edge-aided computing and transmission scheduling for LTE-U-Enabled IoT," *IEEE Trans. Wireless Commun.*, early access, Aug. 25, 2020, doi: 10.1109/TWC.2020.3017207.

[43] P. Kachroo and K. Özbay, *Feedback Control Theory for Dynamic Traffic Assignment*. Cham, Switzerland: Springer, 1999.

[44] L. Liang, H. Ye, and G. Y. Li, "Spectrum sharing in vehicular networks based on multi-agent reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2282–2292, Oct. 2019.

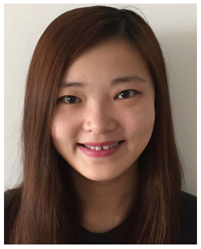
[45] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

Wen Wu (Member, IEEE) received the B.E. degree in information engineering from the South China University of Technology, Guangzhou, China, in 2012, the M.E. degree in electrical engineering from the University of Science and Technology of China, Hefei, China, in 2015, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2019. Since 2019, he has been working as a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, University of Waterloo. His research interests include millimeter-wave networks and AI-empowered wireless networks.



Nan Chen (Member, IEEE) received the bachelor's degree in electrical engineering and automation from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2014, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2019. Her current research interests include electric vehicle charging/discharging scheme design in smart grid, next-generation wireless networks, and machine learning application in vehicular cyber-physical systems.

Conghao Zhou (Graduate Student Member, IEEE) received the B.S. degree from Northeastern University, Shenyang, China, in 2017, and the M.S. degree from the University of Illinois at Chicago, Chicago, IL, USA, in 2018. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research interests include space-air-ground integration networks and machine learning in wireless networks.



Mushu Li (Graduate Student Member, IEEE) received the B.Eng. degree from the University of Ontario Institute of Technology (UOIT), Canada, in 2015, and the M.A.Sc. degree from Ryerson University, Canada, in 2017. She is currently pursuing the Ph.D. degree in electrical engineering with the University of Waterloo, Canada. Her research interests include the system optimization in VANETs and machine learning in wireless networks. She was a recipient of the Natural Science and Engineering Research Council of Canada Graduate Scholarship (NSERC-CGS) in 2018 and the Ontario Graduate Scholarship (OGS) in 2015 and 2016.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990.

He is currently a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research interests include network resource management, wireless network security, the Internet of Things, 5G and beyond, and vehicular ad hoc and sensor networks. He is a registered Professional Engineer of Ontario, Canada, a Fellow of the Engineering Institute of Canada, the Canadian Academy of Engineering, and the Royal Society of Canada, a Foreign Member of Chinese Academy of Engineering, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society. He received the R.A. Fessenden Award from IEEE, Canada, in 2019, the Award of Merit from the Federation of Chinese Canadian Professionals (Ontario) in 2019, the James Evans Avant Garde Award from the IEEE Vehicular Technology Society in 2018, the Joseph LoCicero Award in 2015, the Education Award from the IEEE Communications Society in 2017, and the Technical Recognition Award from Wireless Communications Technical Committee in 2019 and AHSN Technical Committee in 2013. He has also received the Excellent Graduate Supervision Award from the University of Waterloo in 2006 and the Premier's Research Excellence Award (PREA) from the Province of Ontario, Canada, in 2003. He served as the Technical Program Committee Chair/Co-Chair for IEEE GLOBECOM'16, IEEE INFOCOM'14, IEEE VTC'10 Fall, IEEE GLOBECOM'07, and the Chair for the IEEE Communications Society Technical Committee on Wireless Communications. He is the elected Vice President of IEEE Communications Society for Technical & Educational Activities, the Vice President for Publications, the Member-at-Large on the Board of Governors, the Chair of the Distinguished Lecturer Selection Committee, a member of IEEE ComSoc Fellow Selection Committee. He was/is the Editor-in-Chief of the IEEE INTERNET OF THINGS JOURNAL, *IEEE Network*, *IET Communications*, and *Peer-to-Peer Networking and Applications*.

Weihua Zhuang (Fellow, IEEE) has been with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, since 1993, where she is currently a Professor and a Tier I Canada Research Chair in Wireless Communication Networks.

Dr. Zhuang is a Fellow of the Royal Society of Canada, the Canadian Academy of Engineering, and the Engineering Institute of Canada. She was a recipient of the 2017 Technical Recognition Award from the IEEE Communications Society Ad Hoc and Sensor Networks Technical Committee, and a co-recipient of several Best Paper Awards from IEEE conferences. She is an Elected Member of the Board of Governors and Vice President—Publications of the IEEE Vehicular Technology Society. She was an IEEE Communications Society Distinguished Lecturer from 2008 to 2011. She was the Editor-in-Chief of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY from 2007 to 2013, the Technical Program Chair/Co-Chair of IEEE VTC Fall 2017 and Fall 2016, and the Technical Program Symposia Chair of IEEE GLOBECOM 2011.

Xu Li received the B.Sc. degree from Jilin University, China in 1998, the M.Sc. degree from the University of Ottawa in 2005, and the Ph.D. degree from Carleton University in 2008, all in computer science. He is currently a Senior Principal Researcher at Huawei Technologies Canada. Prior to joining Huawei, he worked as a Research Scientist (with tenure) at Inria, France. He contributed extensively to the development of 3GPP 5G standards through more than 90 standard proposals. He has published more than 100 refereed scientific articles and holds over 30 issued U.S. patents. His current research interest includes 5G. He is/was on the editorial boards of *IEEE Communications Magazine*, the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the *Transactions on Emerging Telecommunications Technologies* (Wiley) and a number of other international archive journals.