Efficient Content Delivery in User-Centric and Cache-Enabled Vehicular Edge Networks with Deadline-Constrained Heterogeneous Demands

Md Ferdous Pervej, *Graduate Student Member, IEEE*, Richeng Jin, *Member, IEEE*, Shih-Chun Lin, *Member, IEEE*, and Huaiyu Dai, *Fellow, IEEE*

***Abstract*—Modern connected vehicles (CVs) frequently require diverse types of content for mission-critical decision-making and onboard users' entertainment. These contents are required to be fully delivered to the requester CVs within stringent deadlines that the existing radio access technology (RAT) solutions may fail to ensure. Motivated by the above consideration, this article exploits content caching in vehicular edge networks (VENs) with a software-defined user-centric virtual cell (VC) based RAT solution for delivering the requested contents from a proximity edge server. Moreover, to capture the heterogeneous demands of the CVs, we introduce a preference-popularity tradeoff in their content request model. To that end, we formulate a joint optimization problem for content placement, CV scheduling, VC configuration, VC-CV association and radio resource allocation to minimize long-term content delivery delay. However, the joint problem is highly complex and cannot be solved efficiently in polynomial time. As such, we decompose the original problem into a cache placement problem and a content delivery delay minimization problem given the cache placement policy. We use deep reinforcement learning (DRL) as a learning solution for the first sub-problem. Furthermore, we transform the delay minimization problem into a priority-based weighted sum rate (WSR) maximization problem, which is solved leveraging maximum bipartite matching (MWBM) and a simple linear search algorithm. Our extensive simulation results demonstrate the effectiveness of the proposed method compared to existing baselines in terms of cache hit ratio (CHR), deadline violation and content delivery delay.**

***Index Terms*—Connected vehicle (CV), content caching, delay minimization, software-defined networking (SDN), user-centric networking, vehicular edge network (VEN).**

I. Introduction

ADVANCED driver-assistance systems (ADAS) and infotainment are two premier features for modern connected vehicles (CVs). With advanced radio access technologies (RATs), delivering the Society of Automotive Engineers (SAE) level 5 automation on the road seems more pragmatic day by day. Different government organizations - such as the U.S. Department of Transportation's National Highway Traffic Safety Administration in the United States [1], the Department for Transport in the U.K. [2], etc., set firm regulations for the CVs to ensure public safety on the road. For swift decision-making to satisfy the safety requirements, the CVs need fast, efficient, and reliable communication and data processing. As such, an efficient vehicular edge network (VEN) must ensure uninterrupted and ubiquitous wireless connectivity on the road. Note that a VEN is an edge network that mainly focuses on communication among vehicles and/or between vehicles and infrastructure [3]. To deliver above services, the VEN demands advanced machine learning (ML) tools for resource management complementary to a RAT solution, such as the 5G new-radio (NR) vehicle-to-everything (V2X) communication [4].

With increased automation, in-car entertainment is also becoming a priority for modern CVs [5]. Modern CVs are expected to have many new features, such as vehicular sensing, onboard computation, virtual personal assistant, virtual reality, vehicular augmented reality, autopilot, high-definition (HD) map collection, HD content delivery, etc., [6], [7] that are interconnected for both ADAS and infotainment. For these demands, by exploiting the emerging content caching [8], the centralized core network can remarkably gain by not only ensuring local content distribution but also lessening the core network congestion [9], [10]. As such, VENs can reduce end-to-end latency significantly by storing the to-be-requested contents at the network edge [11], which is vital for the CVs' mission-critical delay-sensitive applications. A practical RAT on top of content caching can, therefore, bring a promising solution for SAE level 5 automation on the road. Moreover, owing to these multifarious requirements, it is also critical to explore the efficacy of content caching with

Manuscript received 14 February 2022; revised 1 October 2022 and 29 March 2023; accepted 29 July 2023. Date of publication 1 August 2023; date of current version 17 January 2024. This work was supported in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LQ23F010021, in part by the Ng Teng Fong Charitable Foundation in the form of ZJU-SUTD IDEA under Grant 188170-11102, in part by the North Carolina Department of Transportation under Grant TCE2020-03, in part by the National Science Foundation under Grant CNS-2210344, in part by the Department of the Air Force under Grant FA9453-23-P-A044, in part by the NC Space Grant, in part by Lockheed Martin Space, in part by Cisco Systems, and in part by the US National Science Foundation under Grants CNS-1824518 and ECCS-2203214. The review of this article was coordinated by Dr. Guiyi Wei. *(Corresponding author: Richeng Jin.)*

Md Ferdous Pervej, Shih-Chun Lin, and Huaiyu Dai are with the Department of Electrical and Computer Engineering, NC State University, Raleigh, NC 27695 USA (e-mail: mpervej@ncsu.edu; slin23@ncsu.edu; hdai@ncsu.edu).

Richeng Jin is with the Department of Information and Communication Engineering, the Zhejiang-Singapore Innovation and AI Joint Research Lab, and the Zhejiang Provincial Key Lab of Information Processing, Communication, and Networking, Zhejiang University, Hangzhou 310007, China (e-mail: richengjin@zju.edu.cn).

Digital Object Identifier 10.1109/TVT.2023.3300954

limited cache storage and different types of content classes, each class with multiple contents, in the content library.

For diverse applications, such as mobile broad bandwidth and low latency (MBBLL), massive broad bandwidth machine-type (mBBMT), massive low-latency machine-type (mLLMT) communications, etc., the CVs urgently need an efficient RAT solution [12]. In the meantime, regardless of the applications, the VEN must ensure omnipresent connectivity to the CVs and deliver their requested contents timely. The so-called user-centric networking [13], [14], [15], [16] is surging nowadays with its ability to shift network resources towards network edge. Note that a user-centric approach is based on the idea of serving users by creating virtual cells (VCs) [17], [18], [19]. While the network-centric approach serves a user from only one base station, the user-centric approach enables serving a user from a VC that may contain multiple transmission points [17], [18], [19]. The latter approach can, thus, not only provide ubiquitous connectivity but also provide higher throughput with minimized end-to-end latency for the end-users [20]. As such, a user-centric approach can combat the frequent changes in received signal strength - often experienced in VENs due to high mobility, by ensuring multipoint data transmission and receptions.

While the user-centric networking approach can bring universal connectivity and MBBLL/mBBMT/mLLMT solutions for the CVs, it induces a more complex network infrastructure. To ensure multipoint data transmission and reception, efficient baseband processing is required. Moreover, as the traditional hardware-based and closed network-centric approach is inflexible, the user-centric approach demands the use of software-defined networking [21], which can offer more efficient and agile node associations and resource allocations in the user-centric approach. With proper system design, it is possible to create VCs with multiple low-powered access points (APs) to ensure that the throughput and latency requirements of the CVs are satisfied. Moreover, amalgamating content caching with the user-centric RAT solution can indeed ensure timely payload delivery for stringent delay-sensitive application requirements of modern CVs. However, this requires a joint study for - content placement, CV scheduling, VC formulation, VC association with the scheduled CV, and radio resource allocation of the APs in the VCs.

A. Related Work

In literature, there exist several works [22], [23], [24], [25], [26], [27], [28], [29] that considered cache-enabled VENs from the traditional network-centric approach. Huang et al. proposed a content caching scheme for the Internet of vehicles (IoVs) in [22]. They developed a delay-aware content delivery scheme exploiting both vehicle-to-infrastructure (V2I) and vehicle-to-vehicle (V2V) links. The authors minimized content delivery delays for the requester vehicles by jointly optimizing cache placement and vehicle associations. Nan et al. also proposed a delay-aware caching technique assuming that the vehicles could either a) decide to wait for better delivery opportunities, or b) get associated with the roadside unit (RSU) that has the content, or c) use one RSU as a relay to extract the content from the cloud in [23]. The authors exploited deep reinforcement learning (DRL) to minimize content delivery cost. However, these assumptions are not suitable for CVs because time-sensitivity plays a crucial role in the quick operation of CVs. [24] proposed quality-of-service ensured caching solution by bounding the content into smaller chunks.

Dai et al. leveraged blockchain and DRL to maximize caching gain [25]. Lu et al. proposed a federated learning approach for secure data sharing among the IoVs [26]. However, [25], [26] assumed that the data rate is perfectly known without any proper resource allocations for the RAT. Zhang et al. addressed proactive caching by predicting user mobility and demands in [27]. Similar prediction-based modeling has also been extensively studied in [8], [30], [31]. Moreover, [27] only analyzed cache hit ratio without incorporating any underlying RAT. Fang et al. considered a static popularity-based cooperative caching solution for roaming vehicles, which assumed constant velocity and downlink data rate and minimized content extraction delay [28]. Liu et al. considered coded caching for a typical heterogeneous network with one macro base station (MBS) overlaid on top of several RSUs [29]. Vehicles trajectory, average residence time within RSU's coverage, and system information were assumed to be perfectly known to the MBS in [29]. Owing to the time-varying channel conditions in VENs, the authors further considered a two-time scaled model. Particularly, they assumed that content requests only arrive at the large time scale (LTS) slot, whereas MBS could decide to orchestrate resources in each small time scale (STS) slot - within the LTS slot. However, although [29] assumed LTS and STS considering time-varying wireless channels, it did not consider any communication model. Therefore, the study presented in [29] did not reflect delay analysis in VENs.

The study presented in [22], [23], [24], [25], [26], [27], [28], [29] mostly considered that the content catalog consist of a fixed number of contents from a single category. In reality, each content belongs to a certain category, and the catalog consists of contents from different categories. Besides, these studies mainly assumed that the users request contents based on popularity. However, each CV may have a specific need for a particular type of content. For example, some CVs may need to have frequent operational information, whereas other CVs may purely consume entertainment-related content. Therefore, a VEN shall consider individual CV's preference, as well as the global popularity.

Some literature also exploited user-centric RAT solutions for VENs [14], [17], [18], [19], [20], [32]. Considering the high mobility of the vehicles, [14] proposed an approach for user-centric VC creation and optimized resource allocation to ensure maximized network throughput. A power-efficient solution for the VC of the VENs was also proposed in [20]. Lin et al. proposed heterogeneous user-centric (HUC) clustering for VENs in [32]. Particularly, the authors considered creating HUC using both traditional APs and vehicular APs. The goal of [32] was to study how HUC migration helps in VEN. Considering both horizontal handover (HO) and vertical HO, [32] studied the tradeoff between throughput and HO overhead. Xiao et al. showed that dynamic user-centric virtual cells could be used

to multicast the same message to a group of vehicles in [17]. Particularly, [17] assumed that a group of vehicles could be considered as a hotspot (HS). If all vehicles inside the HS are interested in the same multicasted message, multiple APs could formulate a VC to serve the HS. [17] optimized power allocation to balance the signal-to-interference-plus-noise ratio for the vehicles in the HS. Shahin et al. also performed similar studies in [18], [19]. Instead of serving a single user, they created HS for V2X broadcast groups. They then maximized the total active HS in the network using admission control, transmission weight selection and power control [18], [19].

B. Motivations and Our Contributions

As ubiquitous connectivity is essential for CVs, the existing RAT solutions may not be sufficient to meet the strict requirements of CVs for higher automation. Existing literature shows that VC-based user-centric networking can bring additional burdens that need rigorous studies, such as mobility and HO management [32]. Moreover, as multicasting delivers a common signal, the study presented in [17], [18], [19] is not suitable for CV-specific independent data requirements in delay-sensitive applications. However, an alternative software-defined networking approach with advanced ML algorithms can potentially bring the RAT solution [14], [20], [33]. Moreover, [14], [20] considered that all APs could serve all users, which may not be possible due to limited coverage and other resource constraints. Inspired by the user-centric VC-based studies [14], [17], [18], [19], [20], [32], our proposed VEN can deploy a close proximity edge server that acts as the software-defined controller. The to-be-requested contents can be prefetched through the edge servers to ensure local delivery. Besides, multiple low-powered APs can be placed as RSUs. The controller can determine the user-centric VC configuration and the corresponding resource orchestration to meet the requirements of the CVs by controlling these APs.

In comparison to the above studies, in this work, we have considered a practical communication model, introduced a preference-popularity tradeoff in content request models, considered a multi-class content catalog, introduced a new VC formation strategy that exploits all possible ways of partitioning the low-powered APs, introduced a duration of interest (DoI) for which the edge server cannot update the cache storage due to practical hardware and overhead constraints, and devised a joint cache placement and user-centric RAT solution. Particularly, our contributions are

- Considering the stringent requirements of the regulatory organizations, we propose a new software-defined user-centric RAT solution that partitions the low-powered APs to form VC, and provides ubiquitous and reliable connectivity to the CVs on the road.
- To ensure fast decision-making for mission-critical operations and uninterrupted onboard entertainment, we exploit content prefetching at the edge server, with multiple classes in the content catalog, while introducing preference-popularity tradeoff into individual content requests owing to the CVs' heterogeneous preferences. Moreover, we introduce a DoI for which the cached contents remain idle due to practical limitations and leverage our proposed RAT solution to deliver the requested contents within a hard deadline.
- We introduce a joint content placement, CV scheduling, VC configuration, CV-VC association and radio resource allocation problem to minimize content delivery delays.
- To tackle the grand challenges of the joint optimization problem, we decompose it into a cache placement sub-problem and a delay minimization sub-problem - given the cache placement policy. We propose a novel DRL solution for the first sub-problem. We then transform the second sub-problem to a weighted sum rate (WSR) maximization problem due to practical limitations and solve the transformed problem using maximum weighted bipartite matching (MWBM) and a simple linear search algorithm.
- Through analysis and simulation results, we verify that our proposed solution achieves better performance than the existing baselines in terms of cache hit ratio (CHR), deadline violation and content delivery delay.

The rest of the paper is organized as follows: Section II introduces our proposed software-defined user-centric system model. Section III presents the caching model, followed by the joint problem formulation in Section IV. Problem transformations are detailed in Section V, followed by our proposed solution in Section VI. Section VII presents extensive simulation results and discussions. Finally, Section VIII concludes the paper. The important notations are listed in Table I.

II. Sofware-Defined User-Centric Communication Model

A. Communication System Model

This article considers a software-defined cache-enabled VEN. An edge server - controlled by a software-defined controller, is placed in proximity to the edge CVs. The edge server has dedicated radio resources with limited local cache storage and is connected to the cloud. Several low-powered APs are deployed as RSUs to provide omnipresent wireless connectivity to the CVs. These APs are connected to the edge server with high-speed wired links. The software-defined centralized controller can control the edge server and perform user scheduling, node associations, precoding, channel estimations, resource allocations, etc. In other words, the edge server acts as the baseband unit. Besides, unlike the legacy system models, we consider a user-centric approach that uses multiple APs to serve the scheduled CVs. These APs are used as RSUs that only perform radio transmissions over the traditional Uu interface [34]. Denote the vehicle and AP set by $\mathscr{U} = \{u\}_{u=1}^{U}$ and $\mathscr{B} = \{b\}_{b=1}^{B}$, respectively. The VEN operates in slotted times. Given B APs at fixed locations, unlike the traditional network-centric approach, the proposed VEN partitions the AP set $\mathscr{B}$ into $W(t) \leq B$ subsets of APs at each slot t. Without loss of generality, we define each subset as a VC. The proposed VEN can assign such a VC to a scheduled CV.

Let there be a set $\mathscr{A}(W(t)) = \{a\}_{a=1}^{A_{W(t)}}$ that defines the possible ways to partition the B APs into $W(t)$ subsets of APs,

TABLE I
IMPORTANT NOTATIONS UTILIZED IN THIS PAPER

Parameter	Definition
$\mathscr{B}, B, b$	Set of APs, total number of APs, b^{th} AP
$\mathscr{U}, U, u$	Set of CVs, total number of CVs, u^{th} CV
$W_{\max}$, $W(t)$	Maximum possible VCs with B APs, total created VCs in slot t
$\mathscr{B}_{vc}(W(t)), A_{W(t)}$, $\mathscr{B}^a_{vc}(W(t))$	All VC configurations set, total possible VC configurations in $\mathscr{B}_{vc}(W(t))$, VC sets under a^{th} configuration
VC^i_a	i^{th} VC of $\mathscr{B}^a_{vc}(W(t))$
$\mathrm{I}^{i,a}_b(t)$	Indicator function that defines whether AP b is in VC^i_a
$\mathrm{I}_u(t)$	Indicator function that defines whether CV u is scheduled at slot t
$\mathrm{I}^{i,a}_u(t)$	Indicator function that defines whether VC VC^i_a is selected for user u at time t
$\bar{Z}$, Z, z, ω	Total network bandwidth, total orthogonal pRB, z^{th} pRB, size of the pRBs
$\mathrm{I}^{b,u}_z(t)$	Indicator function that defines whether pRB z is assigned to AP b to serve CV u at time t
$\psi^u_b(t)$, $\tau^u_b(t)$, $\tilde{\mathbf{h}}^{u,z}_b(t)$, $\mathbf{h}^{u,z}_b(t)$	Large scale fading, log-Normal shadowing, fast fading channel response, entire channel response, respectively, from AP b to CV u during slot t over the z^{th} pRB
$\mathbf{H}^u_b(t)$	Stacked $\mathbf{h}^{u,z}_b$s over all pRBs for CV u and AP b during slot t
$x^u_b(t)$, $\mathbf{s}^u_b(t)$, $\mathbf{w}^{u,z}_b$	Intended signal, symbol, and beamforming vector of AP b for CV u, respectively
P_b	Transmission power of AP b
$y^z_u(t)$, $\Gamma^z_u(t)$	Downlink received signal and downlink SNR at CV u over pRB z, respectively, during slot t
κ	Transmission time interval
$R_u(t)$	Downlink achievable rate at CV u during slot t
$\mathscr{C}$, C, c	Content class set, total content class, c^{th} content class
$\mathscr{F}_c$, F, f_c, $\mathscr{F}$	Content set in class c, total content in a class, f^{th} content of class c, entire content library
$\mathscr{G}_{f_c}$, G_c, g_{fc}	Set of the content features, total number of features, g^{th}_{fc} feature, respectively, of content f of class c
Υ, n	DoI, cache (re)-placement or DoI change counter
Θ^t_u	Bernoulli random variable that defines whether CV u places a content request at time t
Ψ_t	Total content requests from all CVs during time t
$\mathrm{I}^{f_c}_u(t)$	Indicator function that defines whether CV u requests content f_c during time t
$\Omega^{f'}_{f_c}$	Cosine similarity index of content f_c and f'_c
S, Λ, Λ^c	content size, cache storage size of edge server, cache storage to be filled with content from class c
$\mathrm{I}_{fc}(n)$	Indicator function that defines whether content f_c is stored during cache placement counter n
ε_u	Highest probability for content exploitation of CV u
p^u_c	CV u's probability of selecting content class c
p^{fc}_c	Global popularity of content f_c of class c
$d^{m,t}_{u,fc}$, $d^{q,t}_{u,fc}$, $d^{s,t}_{u,fc}$	Content extraction delay from cloud, wait time of $\mathrm{I}^{fc}_u(t)$ before being scheduled, transmission delay
d^{max}_f, $\hat{\mathrm{d}}^{fc}_u(t)$	Maximum allowable delay by the CV, hard-deadline for the edge server to completely offload the requested content
$\bar{d}(t)$	Average delays for all $\mathrm{I}^{fc}_u(t)$s
$m_{ca}, m_{joint}(n)$	Cache placement action, possible action space
$\mathbb{1}_{\mathrm{I}^{fc}_u}(t)$	Cache hit event for $\mathrm{I}^{fc}_u(t)$
$h(t)$, $\mathrm{CHR}(t)$	Total cache hit during slot t, cache hit ratio during slot t
π_{ca}	Cache placement policy of the edge server
$\mathscr{F}^t_{\mathrm{SoI}}$	Slots of interests during slot t
$\mathrm{T}^{t-d^{max}_f+\zeta}_{u,\mathrm{rem}}$, $\mathrm{P}^{t-d^{max}_f+\zeta}_{u,\mathrm{rem}}$	Remaining time to the deadline and payload for the requested content in slot $t-d^{max}_f+\zeta$ at the current slot t
$\mathscr{U}^t_{\mathrm{val}}$, $\mathscr{P}^t_{\mathrm{rem}}$, $\mathscr{D}^t_{\mathrm{rem}}$	Valid CV set during slot t, and their minimum remaining deadline and payload sets, respectively
$\phi_u(t)$	Normalized weights of the CVs in valid CV set during slot t
$\mathscr{U}^t_{\mathrm{sch}}$	Scheduled CV set during slot t
$\bar{R}(t)$	Weighted sum rate of the VEN during slot t
$\mathbf{P}^{top}(n)$	Top-most popular and their Λ^c-top similar contents matrix
$\mathbf{P}^u_{req}(n)$	Content-specific requests history matrix of u in past DoI
$\mathbf{P}^u_{hit}(n)$	Content-specific local cache hit history matrix of u in past DoI
$\mathbf{P}_f(n)$	Measured popularity of contents during n based on past DoI
x^n_{ca}, r^n_{ca}	State and instantaneous reward of the edge server during π_{ca} learning
$\boldsymbol{\theta}_{ca}$, $\boldsymbol{\theta}^-_{ca}$	Online DNN and offline DNN of the edge server for learning the CPP
mem_{ca}, mem^{max}_{ca}	Edge server's memory buffer, maximum length of mem_{ca} for learning π_{ca}
G, $\mathbf{R}_t$	Bipartite graph, weighted data rate matrix
$e(b,z)$, $R_t[b,z]$	Edge connecting vertex b and z, and corresponding weights
$R^{bit}_u(t)$	Possible transmissible bits for CV u during slot t

i.e., VCs. Denote the i^{th} VC of $a \in \mathscr{A}(W(t))$ by $VC^i_a \subset \mathscr{B}$ and the set of VCs by $\mathscr{B}^a_{vc}(W(t)) = \{VC^i_a\}^{W(t)}_{i=1}$. Furthermore, for each $a \in \mathscr{A}(W(t))$, the VC must obey the following rules:

$$VC^i_a \neq \emptyset,\ \forall a \text{ and } i, \tag{1a}$$

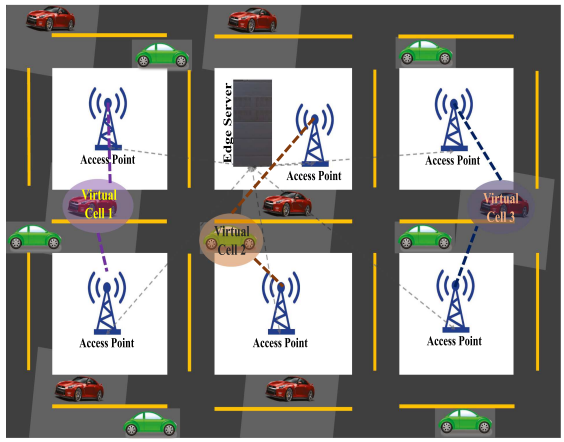

Fig. 1. Proposed user-centric cache-enabled vehicular edge network.

$$VC^i_a \bigcap VC^{i'}_a = \emptyset,\ \forall a \text{ and } i \neq i', \tag{1b}$$

$$\bigcup^{W}_{i=1} VC^i_a = \mathscr{B},\ \forall a. \tag{1c}$$

The first rule in (1a) means that the VC^i_as must not be empty, while the second rule in (1b) means the subsets are mutually exclusive. Besides, the last rule in (1c) represents that the union of the VC^i_as must yield the original AP set $\mathscr{B}$. When $W(t) = 3$ and $B = 6$, for one possible $a \in \mathscr{A}(W(t))$, the VC set $\mathscr{B}^a_{vc}(W(t))$ is shown in Fig. 1 by the filled ovals.

Then, for a given $W(t)$, the VEN can formulate the total VC configuration pool $\mathscr{B}_{vc}(W(t)) = \{\mathscr{B}^a_{vc}(W(t))\}^{A_{W(t)}}_{a=1}$. Moreover, the VEN can partition the B APs into $W(t)$ VCs following the above rules in $A_{W(t)} = W(t)!B_{vc}(B, W(t))$ ways, where $B_{vc}(B, W(t))$ is calculated as

$$B_{vc}(B, W(t)) = \frac{1}{W(t)!}\sum^{W(t)}_{\bar{w}=1}(-1)^{\bar{w}}\binom{W(t)}{\bar{w}}(W(t)-\bar{w})^B. \tag{2}$$

Note that $B_{vc}(B, W(t))$ is commonly known as the Stirling number of the second kind [35].

To this end, as the VCs contain different AP configurations, denote the VC and AP mapping by

$$\mathrm{I}^{i,a}_b(t) = \begin{cases} 1, & \text{if AP } b \text{ is in } VC^i_a \text{ during slot } t, \\ 0, & \text{otherwise}, \end{cases}. \tag{3}$$

The edge server selects the total $W(t)$ number of VCs to form and their configuration $\mathscr{B}^a_{vc}(W(t)) \in \mathscr{B}_{vc}(W(t))$. The VCs $VC^i_a \in \mathscr{B}^a_{vc}(W(t))$ are assigned to serve the scheduled CVs. Denote the CV scheduling and VC-CV association decisions by the following indicator functions:

$$\mathrm{I}_u(t) = \begin{cases} 1, & \text{if CV } u \text{ is scheduled in slot t}, \\ 0, & \text{otherwise}, \end{cases}. \tag{4}$$

$$\mathrm{I}^{i,a}_u(t) = \begin{cases} 1, & \text{if } VC^i_a \text{ is selected for CV } u \text{ in slot } t, \\ 0, & \text{otherwise}, \end{cases}. \tag{5}$$

Note that (5) means that if $\mathrm{I}^{i,a}_u(t) = 1$, i.e., VC^i_a is assigned to CV u, then the CV is connected to all APs inside this VC^i_a.

We contemplate that the VEN operates in frequency division duplex (FDD) mode and has a fixed $\bar{\mathrm{Z}}$ Hz bandwidth. The edge server uses this dedicated $\bar{\mathrm{Z}}$ Hz bandwidth and divides it into Z orthogonal physical resource blocks (pRBs). Let the set of the orthogonal pRBs be $\mathscr{Z} = \{z\}_{z=1}^{Z}$ for the downlink infrastructure-to-vehicle (I2V) communication. Denote the size of a pRB by ω, while we introduce the following indicator function for the pRB allocation

$$\mathrm{I}_z^{b,u}(t) = \begin{cases} 1, & \text{if pRB z is assigned to AP b when} \\ & \mathrm{I}_u^{i,a}(t) = 1 \text{and} \mathrm{I}_{\mathrm{b}}^{\mathrm{i,a}}(\mathrm{t}) = 1, \\ 0, & \text{otherwise,} \end{cases} \quad . \tag{6}$$

B. Communication Channel Modeling

We consider single antenna CVs whereas, the APs are equipped with $L > 1$ antennas. Let us denote the channel response at a CV u from the AP b, over pRB z, as follows:

$$\mathbf{h}_b^{u,z}(t) = \sqrt{\psi_b^u(t)}\tau_b^u(t)\breve{\mathbf{h}}_b^{u,z}(t) \in \mathbb{C}^{L\times 1}, \tag{7}$$

where $\sqrt{\psi_b^u(t)}$, $\tau_b^u(t)$ and $\breve{\mathbf{h}}_b^{u,z}(t) = [h_{b,1}^{u,z}(t), \ldots, h_{b,L}^{u,z}(t)]^T \in \mathbb{C}^{L\times 1}$ are large scale fading, log-Normal shadowing and fast fading channel responses from the L antennas, respectively. Besides, $h_{b,l}^{u,z}(t)$ is the l^{th} row of $\breve{\mathbf{h}}_b^{u,z}(t)$ that denotes the channel between u and the l^{th} antenna of AP b at time t over pRB z. Note that we consider the urban macro (UMa) model [36] for modeling the path losses following 3rd Generation Partnership Project (3GPP) standard [34]. Then, for all pRBs in the system, we express the wireless channels from AP b to CV u, at time t, as $\mathbf{H}_u^b(t) = [\mathbf{h}_b^{u,1}(t), \ldots, \mathbf{h}_b^{u,Z}(t)] \in \mathbb{C}^{L\times Z}$. We consider the edge server has perfect channel state information (CSI)[1] and all transceivers can mitigate the Doppler effect.

C. Transceiver Modeling

The transmitted signal at the AP b for CV u is $\mathbf{s}_b^u(t) = \sqrt{P_b}x_b^u(t)\mathbf{w}_b^{u,z}(t) \in \mathbb{C}^{L\times 1}$, where P_b is the transmission power of AP b. Besides, $x_b^u(t)$ and $\mathbf{w}_b^{u,z}(t) \in \mathbb{C}^{L\times 1}$ are the unit powered intended signal and corresponding precoding vector over pRB z, respectively, of the AP b for u during slot t. Then, the transmitted signals for CV u from all APs is $\mathbf{s}_u(t) = [\mathbf{s}_1^u(t), \ldots, \mathbf{s}_B^u(t)]^T \in \mathbb{C}^{L\times B}$. Moreover, as each AP transmits over orthogonal pRBs, the proposed VEN does not have any interference. To this end, we calculate the received signal at CV u, over pRB z, as

$$y_u^z(t) = \sum\nolimits_{i=1}^{W(t)} \mathrm{I}_u^{i,a}(t) \times \left(\sum\nolimits_{b=1}^{B} \mathrm{I}_b^{i,a}(t)\cdot \mathrm{I}_z^{b,u}(t)\left[\mathbf{h}_b^{u,z}(t)^H \mathbf{s}_b^u(t) + \eta_b^u(t)\right]\right), \tag{8}$$

where $\eta_b^u(t) \sim CN(0, \sigma_b^2)$ is zero mean circularly symmetric Gaussian distributed noise with variance σ^2. The corresponding signal-to-noise ratio (SNR), over pRB z, is

$$\Gamma_u^z(t) = \frac{\sum_{i=1}^{W(t)} \mathrm{I}_u^{i,a}(t)\left(\sum_{b=1}^{B} \mathrm{I}_b^{i,a}(t)\cdot \mathrm{I}_z^{b,u}(t)\left[P_b\left|\mathbf{h}_b^{u,z}(t)^H \mathbf{w}_b^{u,z}(t)\right|^2\right]\right)}{\omega\left[\sum_{i=1}^{W(t)} \mathrm{I}_u^{i,a}(t)\cdot\left(\sum_{b=1}^{B} \mathrm{I}_b^{i,a}(t)\cdot\sigma_b^2\right)\right]}. \tag{9}$$

Therefore, the total downlink achievable capacity for CV u is

$$R_u(t) = \sum\nolimits_{z=1}^{Z} \omega \cdot \log_2\left(1 + \Gamma_u^z(t)\right). \tag{10}$$

III. Edge Caching Modeling

A. Definitions and Assumptions

To avoid cross-domain nomenclature, we present necessary terms and our assumptions in the following.

Definition 1 (Content): The source file that the CVs request is defined as content. These files can contain CVs' operational information, geographic information, map/navigation information, weather conditions, compressed file with sensory data, local news, video/audio clips, etc.

Definition 2 (Content Class): Each content belongs to a class that defines the type/category of the content. Let there be F contents in each class and the set of the content class be $\mathscr{C} = \{c\}_{c=1}^{C}$, where $C \in \mathbb{Z}^+$. Denote the content set of class c by $\mathscr{F}_c = \{f_c\}_{f=1}^{F}$, where f_c represents the f^{th} content of class c. Moreover, let the content size be S bits.

Definition 3 (Content Features): Let the content in the c^{th} class have $G_c \in \mathbb{Z}^+$ features. Denote the feature set of content f_c by $\mathscr{G}_{f_c} = \{g_{f_c}\}_{g=1}^{G_c}$. Note that the content features are essentially the descriptive attributes of the content. For example, it can be the genre/type of the content, names of the directors, actors, actresses, geolocation information, timestamp, etc.

Definition 4 (Content Library): The content library is comprised of all contents from all classes. Let $\mathscr{F} = \bigcup_{c=1}^{C} \mathscr{F}_c$ be the content library.

Definition 5 (Duration of Interest (DoI)): The period for which the content library remains fixed is denoted as the duration of interest (DoI). Denote this period by Υ.

The content library $\mathscr{F}$ is fixed. While the CVs may request contents in each slot t, the edge servers can update its cached content only in each $t = n\Upsilon$ time slot, where $n \in \mathbb{Z}^+$ defines the cache (re)placement counter. Note that this assumption is made as it may not be practical to update the cache storage in each slot t due to hardware limitations. Moreover, we assume that the content update/refresh process is independent of the content request arrival process. To this end, we focus on request arrival modeling from the CVs, followed by modeling their preferences, i.e., their requested $f_c \in \mathscr{F}$ in slot t.

B. User Request/Traffic Modeling

We assume that the CVs can make content requests in each slot t following *Bernoulli* distribution[2]. At slot t, let Θ_u^t denote a *Bernoulli* random variable - with success probability p_u, that defines whether u makes a content request or not. The total number of requests during a DoI Υ for an individual CV u follows *Binomial* distribution. Besides, at slot t, the total number of requests

[1] Although channel reciprocity does not hold in FDD, the edge server can use some feedback channels to estimate the CSI.

[2] Similar access modeling was also used in existing works [37], [38].

from all CVs, i.e., $\Psi_t = \sum_{u=1}^{U} \Theta_u^t$, follows a Poisson binomial distribution [39] with probabilities $\mathbf{p} = \{p_u\}_{u\in\mathscr{U}}$. Moreover, the probability of the distribution of Ψ_t can be bounded using Proposition 1.

Proposition 1: Let $\bar{\mu} = \mathbb{E}[\Psi_t] = \sum_{u=1}^{U} p_u$ and $\bar{p} = (1/U)\sum_{u=1}^{U} p_u$ be the average success probability. Then, at slot t, the probability that the distribution of the total number of requests of the CVs gets larger than some $\xi = \bar{\mu} + \delta$ and $0 < \delta < U - \bar{\mu}$, is bounded above as follows:

$$\Pr\{\Psi_t \geq \xi\} \leq \exp\left[-U D_{\bar{p}}(\chi)\right], \tag{11}$$

where $D_{\bar{p}}(\chi) = \chi \ln(\chi/\bar{p}) + (1-\chi)\ln((1-\chi)/(1-\bar{p}))$ is the relative entropy of χ to that of $\bar{p}$ and $\chi = \xi/U$.

Proof: Please see Appendix A. ■

C. Individual User Preference Modeling

Given that a CV makes a request, we now focus on the *which* question, i.e., given that $\Theta_u^t = 1$, which content shall that CV request at that slot? Let us express a particular content f_c requested by CV u during slot t by

$$\mathrm{I}_u^{f_c}(t) = \begin{cases} 1, & \text{if } \Theta_u^t = 1 \text{ and } u \text{ request } f_c, \\ 0, & \text{otherwise}, \end{cases} \tag{12}$$

Unlike legacy modeling[3], we consider that a CV's choice depends both on its personal preference and global popularity. Depending on the operational needs, a CV may prefer to consume a specific content related to its operation. Besides, it may also prefer consuming a content very similar to the one that it previously consumed. Moreover, it may also get influenced by the popularity of the contents. For example, at slot t, a CV may request operation-related content f_c specific to that particular timestamp. At slot $t+1$, it may then need another operation-related content f_c' that is very similar to f_c. Similarly, for purely entertainment-related content, the request can get influenced by the user's previous experience. The user may also choose to consume the most popular content at that time. As such, we model the user's content request as the *exploitation-exploration* tradeoff between personal preference and global popularity of the contents. We present this by the ϵ_u-policy, i.e., a CV exploits with probability ϵ_u and explores with probability $(1-\epsilon_u)$.

1) Content Selection during Exploitation: In this case, the CV exploits its preferred contents from the same class it previously consumed a content. Given that CV has requested f_c in slot t, it will request the most similar content to f_c in class c with probability ϵ_u if $\Theta_u^{t+1} = 1$. Note that similarity between f_c and f_c', where $f_c \neq f_c'$ is calculated as

$$\Omega_{f_c}^{f_c'} = \Big(\sum_{g\in\mathscr{G}_{f_c}} g_{f_c}\, g_{f_c'}\Big) \Big/ \Big(\sqrt{\sum_{g\in\mathscr{G}_{f_c}} g_{f_c}^2}\sqrt{\sum_{g\in\mathscr{G}_{f_c'}} g_{f_c'}^2}\Big). \tag{13}$$

2) Content Selection during Exploration: Given that the CV requested content from class c previously, it will explore new content from a different class $c' \neq c$. Denote CV u's class selection probability by p_c^u, which follows a *categorical* distribution. Once the CV chooses the new content class c, it randomly selects a content from this class based on global popularity. Denote the global popularity of contents in class c by $\mathbf{p}_c^f = \{p_c^{f_c}\}_{f_c\in\mathscr{F}_c}$, where $p_c^{f_c}$ is the popularity of content f_c.

Note that our design boils down to a solely popularity-based model [11] when there is only a single content class and $\epsilon_u = 0$.

D. Content (Re)placement in the Cache

Recall that only the edge server has limited cache storage in the proposed VEN. Let the cache storage of the edge server be Λ. Denote the cache placement indicator by the binary indicator function $\mathrm{I}_{f_c}(n)$ during cache placement counter n. The edge server obeys the following rules for cache placement:

$$S \cdot \sum_{c=1}^{C}\sum_{f=1}^{F} \mathrm{I}_{f_c}(n) \leq \Lambda, \tag{14}$$

$$S \cdot \sum_{f=1}^{F} \mathrm{I}_{f_c}(n) = \Lambda^c, \tag{15}$$

where Λ^c is the total storage taken by the cached content from class $c \in \mathscr{C}$. Moreover, (14) ensures that the size of the total cached contents must not be larger than the storage capacity. At each $t = n\Upsilon$, the software-defined controller pushes the updated contents into the cache storage. These contents remain at the edge servers' cache storage till the next DoI update.

IV. Delay Modeling and Problem Formulation

A. Content Delivery Delay Modeling

For higher automation (and uninterrupted entertainment of the onboard users), the CVs may need to continuously access diverse contents within a tolerable delay to avoid fatalities (and quality of experiences (QoEs)). This motivates us to introduce a hard deadline requirement for the edge server to deliver the requested contents. We consider that content requests arrive continuously following Θ_u^ts. Each CV can make at most a single content request based on its preference if $\Theta_u^t = 1$. The requester has an associated hard deadline requirement $d_f^{\max}$, within which it needs the entire payload. The edge server, on the other hand, can have a shorter deadline, denoted by $\hat{\mathrm{d}}_u^{f_c}(t)$, associated with the requests as it replaces the content at the end of each DoI. Formally, the deadline for the edge server to fully offload a requested content is

$$\hat{\mathrm{d}}_u^{f_c}(t) = \min\left\{d_f^{\max}, (n+1)\Upsilon - t\right\}, \quad \forall t \in [n\Upsilon, (n+1)\Upsilon]. \tag{16}$$

Essentially, (16) ensures that the edge server cannot exceed the minimum of the maximum allowable delay threshold $d_f^{\max}$ and remaining time till the next cache replacement slot $(n+1)\Upsilon$.

To that end, we calculate the associated delay of delivering the requested contents from all CVs $u \in \mathscr{U}$. This delay depends on whether the requested content has been prefetched and the underlying wireless communication infrastructure. Particularly, for the cache miss event[4], the requested content is extracted from the cloud. This extraction causes an additional delay and involves the upper layers of the network[5]. Denote the delay for extracting

[3]Legacy model assumes that content requests follow Zipf distribution [22], [23], [28], which does not capture the personal preferences of the users[8].

[4]When the edge server needs to serve content request $\mathrm{I}_u^{f_c}(t)$ but it does not have content f_c in its local cache storage is known as the cache miss.

[5]We assume that each miss event needs to be handled by the upper layers.

content f_c, requested by CV u during slot t, from the cloud by $d^{m,t}_{u,f_c}$. Moreover, we consider two more additional delays. The first one is the wait time of a request before being scheduled for transmission by the edge server, given that the edge server has either prefetched the content during the cache placement slot or the upper layers have already processed the requested content from the cloud. The second one is the transmission delay. Denote these two delays, i.e., wait time and transmission delays, for $\mathrm{I}^{f_c}_u(t)$ by $d^{q,t}_{u,f_c}$ and $d^{s,t}_{u,f_c}$, respectively. Therefore, the total delay of delivering the entire content is calculated as

$$d^{f_c}_u(t) = \left[1 - \mathrm{I}_{f_c}(n)\right] d^{m,t}_{u,f_c} + d^{q,t}_{u,f_c} + d^{s,t}_{u,f_c}. \tag{17}$$

Thus, the average content delivery delay for all CVs is

$$\bar{\mathrm{d}}(t) = (1/U) \sum\nolimits_{u=1}^{U} \sum\nolimits_{c=1}^{C} \sum\nolimits_{f=1}^{F} \mathrm{I}^{f_c}_u(t) \cdot d^{f_c}_u(t). \tag{18}$$

B. Joint Problem Formulation

We aim to find joint cache placement $\mathrm{I}_{f_c}(n)$, user scheduling $\mathrm{I}_u(t)$, total $W(t)$ VC to form, the VC configuration $\mathscr{B}^a_{vc}(W(t))$, VC association $\mathrm{I}^{i,a}_u(t)$ and radio resource allocation for the serving APs in the selected VCs, i.e., $\mathrm{I}^{b,u}_z(t)$s to minimize long-term expected average content delivery delay for the CVs. As such, we pose our joint optimization problem as

$$\underset{\mathrm{I}_{f_c}(n), \mathrm{I}_u(t), W(t), \mathscr{B}^a_{vc}(W(t)), \mathrm{I}^{i,a}_u(t), \mathrm{I}^{b,u}_z(t)}{\text{minimize}} \quad \mathrm{d} = \limsup_{T\to\infty} \mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} \bar{\mathrm{d}}(t)\right] \tag{19}$$

$$\text{s.t.} \quad C_1: \sum\nolimits_{c=1}^{C} \sum\nolimits_{f=1}^{F} \mathrm{I}^{f_c}_u(t) \le 1, \quad \forall u, t \tag{19a}$$

$$C_2: (14), (15), \quad \forall n, \tag{19b}$$

$$C_3: \sum\nolimits_{u=1}^{U} \mathrm{I}_u(t) \le W(t), \quad \forall t, \tag{19c}$$

$$C_4: \sum\nolimits_{i=1}^{W(t)} \mathrm{I}^{i,a}_u(t) = 1, \quad \forall u, \tag{19d}$$

$$C_5: \sum\nolimits_{u=1}^{U} \mathrm{I}^{i,a}_u(t) = 1, \quad \forall i, \tag{19e}$$

$$C_6: \sum\nolimits_{u=1}^{U} \sum\nolimits_{i=1}^{W(t)} \mathrm{I}^{i,a}_u(t) = W(t), \tag{19f}$$

$$C_7: W(t) = \min\left\{\sum\nolimits_{u=1}^{U} \mathrm{I}_u(t), \mathrm{W}_{\max}\right\}, \tag{19g}$$

$$C_8: \sum\nolimits_{b=1}^{B} \mathrm{I}^{b,u}_z(t) = 1, \quad \forall z, u, \tag{19h}$$

$$C_9: \sum\nolimits_{z=1}^{Z} \mathrm{I}^{b,u}_z(t) = 1, \quad \forall b, u, \tag{19i}$$

$$C_{10}: \sum\nolimits_{u=1}^{U} \mathrm{I}^{b,u}_z(t) = 1, \quad \forall z, b, \tag{19j}$$

$$C_{11}: \sum\nolimits_{z=1}^{Z} \sum\nolimits_{b=1}^{B} \sum\nolimits_{u=1}^{U} \mathrm{I}^{b,u}_z(t) = Z, \tag{19k}$$

$$C_{12}: d^{f_c}_u(t) \le \hat{\mathrm{d}}^{f_c}_u(t), \tag{19l}$$

$$C_{13}: \mathrm{I}_{f_c}(n), \mathrm{I}_u(t), \mathrm{I}^{i,a}_u(t), \mathrm{I}^{b,u}_z(t) \in \{0, 1\}, \tag{19m}$$

where $\mathrm{W}_{\max}$ is the maximum allowable number of VCs in the system. Constraint C_1 ensures that each CV can request at most one content in each slot t. The constraints (14) and (15) in C_2 are due to physical storage limitations. Constraint C_3 in (19c) restricts the total number of scheduled CVs to at max the total number of created VCs $W(t)$. Constraints (19d), (19e), and (19f) make sure that each CV can get at max one VC, each VC is assigned to at max one CV and summation of all assigned VCs is at max the total number of available VCs, respectively. Besides, constraints C_7 in (19g) restricts the total VCs $W(t)$ to be at max the minimum of the total scheduled CVs and $\mathrm{W}_{\max}$. Furthermore, constraints (19h), (19i) and (19j) ensure that each AP can get at max one pRB, each pRB is assigned to at max one AP and each CV gets non-overlapping resources, respectively. Constraint C_{11} in (19k) ensures that all available radio resources are utilized. C_{12} is introduced to satisfy the entire payload delivery delay of a requested content to be within the edge server's hard deadline $\hat{\mathrm{d}}^{f_c}_u(t)$. Finally, constraints in (19m) are the feasibility space.

Remark 1 (Intuitions behind the constraints): Constraint C_1 incorporates CV's content request, while C_2 is for the cache placement at the edge server. Constraint C_3 is for CV scheduling. Constraints C_4 - C_7 are for the user-centric RAT's VC formation and associations. Besides, constraints C_8 - C_{11} are for radio resource allocation. Moreover, C_{12} is introduced to satisfy the hard deadline for delivering the CVs' requested contents, which holds if $\sum_{\bar{t}=t}^{\hat{\mathrm{d}}^{f_c}_u(t)} \mathrm{I}_u(\bar{t}) \cdot \kappa \cdot R_u(\bar{t}) \ge S$, where κ is the transmission time interval (TTI).

Remark 2: The total delay associated with each content request, calculated in (17), depends on both cache placement and the RAT. More specifically, an efficient cache placement solution can minimize cache miss events, i.e., minimize $d^{m,t}_{u,f_c}$s. On the other hand, $d^{q,t}_{u,f_c}$ and $d^{s,t}_{u,f_c}$ depend on the CV scheduling, and total VC $W(t)$, VC configuration $\mathscr{B}^a_{vc}(W(t))$, CV-VC association $\mathrm{I}^{i,a}_u(t)$ and radio resource allocation $\mathrm{I}^{b,u}_z(t)$.

Note that the optimization problem in (19) is an average Markov decision process (MDP) over an infinite time horizon with different combinatorial optimization variables. Recall that the CSI varies in each slot t. Besides, in slot t, neither the CV's to-be-requested contents nor the CSI in the future time slots are known beforehand. As such, without knowing these details, the optimal decision variables may not be known. In the subsequent section, we will prove that even the reduced problems of this complex joint optimization are NP-hard. Moreover, the decision variables are different in different time slots. As such, we decompose the original problem into two sub-problems. The first sub-problem transforms the cache placement problem, which will be solved using a learning solution. The second sub-problem introduces a joint CV scheduling, total $W(t)$ VC formation, association and resource allocation optimization problem for minimizing the average content delivery delay, given that the edge server knows the cache placement decisions. The learning solution for the cache placement depends on the following preliminaries of DRL.

C. Preliminary of Deep Reinforcement Learning

An MDP contains a set of states $\mathscr{X} = \{x\}_{x=1}^{|\mathscr{X}|}$, a set of possible actions $\mathscr{M} = \{m\}_{m=1}^{|\mathscr{M}|}$, a transition probability $P_{tt'}(m)$ from the current state $x_t \in \mathscr{X}$ to the next state $x_{t'} \in \mathscr{X}$ when action m is taken, and an immediate reward $R_t(m)$ for this state

transition [40]. RL perceives the best way of choosing actions in an unknown environment through repeated observations and is widely used for solving MDP. The RL agent learns policy $\pi : \mathscr{M} \times \mathscr{X} \rightarrow [0, 1]$, where $\pi(x_t, m) = \Pr\{m|x_t\}$ denotes the probability of taking action m given the agent is at state x_t. Following π, given the agent is at state x_t, the expected return from that state onward denotes how good it is to be at that state and is measured by the following state-value function:

$$V_\pi(x_t) = \mathbb{E}[R|x_t, \pi] = \mathbb{E}\left[\sum\nolimits_{t'=t}^{\mathrm{T_{end}}} \gamma^{t'-t} R_{t'}(m)|x_t, \pi\right], \quad (20)$$

where $\gamma \in [0, 1]$ is the discount factor, $\mathrm{T_{end}}$ is the time step at which the episode ends, and $R_{t'}(m)$ is the reward at step t'. Moreover, the quality of an action taken at a state is ascertained by the following action-value function [40]:

$$Q(x_t, m) = R_t(m) + \gamma \sum\nolimits_{x_{t'} \in \mathscr{X}} P_{tt'}(m) V_\pi(x_{t'}). \quad (21)$$

The agent's goal is to find optimal policy π^* to maximize

$$Q^*(x_t, m) = R_t(m) + \gamma \sum\nolimits_{x_{t'} \in \mathscr{X}} P_{tt'}(m) V_{\pi^*}(x_{t'}), \quad (22a)$$

where $V_{\pi^*}(x_{t'}) = \max\limits_{\check{m} \in \mathscr{M}} Q^*(x_{t'}, \check{m})$. This $Q(x_t, m)$ value is updated as [41]

$$Q\left(x_t, m\right) \leftarrow (1 - \alpha) Q\left(x_t, m\right) + \alpha \bar{y}_t, \quad (23)$$

where α is the learning rate and $\bar{y}_t = R_t(m) + \gamma \max\limits_{\check{m} \in \mathscr{M}} Q(x_{t'}, \check{m}))$ is commonly known as the temporal target. Usually, a deep neural network (DNN), parameterized by its weight $\boldsymbol{\theta}$, is used to approximate $Q^*(x, m) \approx Q(x, m; \boldsymbol{\theta})$, which is known as the so-called DRL [42]. The agent is trained by randomly sampling S_b batches from a memory buffer $\mathscr{D}$, which sores of the agent's experiences $\{x_t, m, R_t, x_{t'}\}$, and performing stochastic gradient descent (SGD) to minimize the following loss function [42]:

$$\mathrm{L}(\boldsymbol{\theta}) = \left[\bar{y}_t(\boldsymbol{\theta}) - Q\left(x_t, m; \boldsymbol{\theta}\right)\right]^2, \quad (24)$$

where $\bar{y}_t(\boldsymbol{\theta}) = R_t(m) + \gamma_{\check{m} \in \mathscr{M}} \max Q(x_{t'}, \check{m}; \boldsymbol{\theta})$. While the same DNN $\boldsymbol{\theta}$ can be used to predict both $Q(x_t, m; \boldsymbol{\theta})$ and the target $\bar{y}_t(\boldsymbol{\theta})$, to increase learning stability, a separate target DNN, parameterized by $\boldsymbol{\theta}^-$, is used to predict $\bar{y}_t(\boldsymbol{\theta}^-)$ [42].

Here we emphasize that since the Q value functions are estimated using the DNN $\boldsymbol{\theta}$, i.e., $Q^*(x, m) \approx Q(x, m; \boldsymbol{\theta})$, unlike classical tabular Q-learning, the DRL solution may not be optimal [40]. To that end, we first introduce our problem transformation in the next section, followed by more pertinent information on a DRL-based solution in Section VI.

V. Problem Transformations

Since the original problem is hard to solve and the decision variables are not the same in different time slots, we decompose the original problem by first devising a learning solution for cache placement policy (CPP) for the cache placement slot $t = n\Upsilon$. Then, we use this learned CPP to re-design the delay minimization problem from the RAT perspective. Intuitively, given that the best CPP for the slot $t = n\Upsilon$ is known, in order to ensure minimized content delivery delay, one should optimize the RAT parameters jointly.

A. Cache Placement Policy (CPP) Optimization Sub-Problem

We want to learn the CPP π_{ca} that provides the optimal cache placement decision $\mathrm{I}_{f_c}(n)$ in the cache placement time slots $t = n\Upsilon$, $\forall n$. We have total $\mathrm{m}_{ca} = \prod_{c=1}^{C} \mathrm{m}_{ca}^c = \prod_{c=1}^{C} \binom{F}{\Lambda^c}$ ways for content placement as Λ^c and Λ are of the unit of content size S based on constraints (14) and (15). Moreover, the CPP π_{ca} is a mapping between the system state x_{ca}^n and an action m_{ca} in the joint action space with $\mathrm{m}_{ca}(n)$ possible actions. To this end, let us define a cache hit event by

$$\mathbb{1}_{\mathrm{I}_u^{f_c}}(t) = \begin{cases} 1, & \text{if } \mathrm{I}_u^{f_c}(t) = 1 \text{ and } \mathrm{I}_{f_c}(n) = 1, \\ 0, & \text{otherwise.} \end{cases} \quad (25)$$

Besides, the total cache hit at the edge server is calculated as the summation of the locally served requests and is calculated as $h(t) = \sum_{u=1}^{U} \mathbb{1}_{\mathrm{I}_u^{f_c}}(t)$. Thus, we calculate the CHR as

$$\mathrm{CHR}(t) = h(t) / \left(\sum\nolimits_{u=1}^{U} \mathrm{I}_u^{f_c}(t)\right). \quad (26)$$

Next, we devise the CPP of the edge server that ensures a long-term CHR while satisfying the cache storage constraints. Formally, we pose the optimization problem as follows:

$$\underset{\pi_{ca}}{\text{maximize}} \quad \mathrm{CHR}(\pi_{ca}) = \lim_{T \rightarrow \infty} \mathbb{E}\left[(1/T) \sum\nolimits_{t=1}^{T} \mathrm{CHR}(t)\right], \quad (27)$$

$$\text{s. t.} \quad C_1, C_2, \mathrm{I}_{f_c}(n) \in \{0, 1\}, \quad (27a)$$

where C_1 and C_2 are introduced in (19).

Theorem 1: The CHR maximization problem (27) is NP-hard.

Proof: Please see Appendix B. ■

B. Joint Optimization Problem for the User-Centric RAT

Note that as the delay of extracting a content from the cloud during a cache miss event is fixed, the first term in (17) will be minimized if the CPP π_{ca} ensures maximized CHR. In this sub-problem, we focus on the other two delays $d_{u,f_c}^{q,t}$ and $d_{u,f_c}^{s,t}$ in (17) by jointly optimizing scheduling, VC formation, VC association and radio resource allocation of the proposed user-centric RAT solution assuming the cache placement is known at the edge server. Therefore, we pose the following modified content delivery delay minimization problem.

$$\underset{\mathrm{I}_u(t), W(t), \mathscr{B}_{vc}^a(W(t)), \mathrm{I}_u^{i,a}(t), \mathrm{I}_z^{b,u}(t)}{\text{minimize}} \quad \mathrm{d} = \limsup_{T \rightarrow \infty} \mathbb{E}\left[(1/T) \sum\nolimits_{t=1}^{T} \bar{\mathrm{d}}(t)\right] \quad (28)$$

$$\text{s. t.} \quad C_3, C_4, C_5, C_6, C_7, C_8, C_9, C_{10}, C_{11}, C_{12}, \quad (28a)$$

$$\mathrm{I}_u(t), \mathrm{I}_u^{i,a}(t), \mathrm{I}_z^{b,u}(t) \in \{0, 1\}, \quad (28b)$$

where the constraints in (28a) and (28b) are taken for the same reasons as in the original problem in (19).

Sub-problem (28) contains combinatorial optimization variables and, thus, is NP-hard. An exhaustive search for optimal parameters is also infeasible due to the large search space as well as sequential dependencies for the deadline constraints in

C_{12}. Besides, as each content request arrives with a deadline constraint and wireless links vary in each slot, we consider that the edge server adopts priority-based scheduling. Intuitively, given the fact that the edge server does not know the transmission delay $d^{s,t}_{u,f_c}$ due to channel uncertainty and it needs to satisfy constraint C_{12} for all $\mathrm{I}^{f_c}_u(t)$s, it should schedule the CVs with earliest-deadline-first (EDF)[6] followed by optimal VC formation, association and radio resource allocation. Note that EDF is widely used for scheduling in real-time operating systems [45]. If EDF cannot guarantee zero deadline violation for the tasks, no other algorithm can [44]. In our case, scheduling also depends on the availability of the requested content at the edge server. In cache miss event, the edge server must wait for $d^{m,t}_{u,f_c}$ so that the upper layers can extract the content from the cloud.

Upon receiving a content request $\mathrm{I}^{f_c}_u(t)$, the edge server checks $\mathrm{I}_{f_c}(n)$. If $\mathrm{I}_{f_c}(n)=0$, the request is forwarded to the upper layers. The upper layer initiates the extraction process from the cloud. At each slot t, before making the scheduling and VC formation decisions, the edge server considers previous $\mathscr{T}^t_{\text{SoI}}$ slots information. These $\mathscr{T}^t_{\text{SoI}}$ slots are termed as our slots of interest (SoI) and are calculated in (29).

$$\mathscr{T}^t_{\text{SoI}}=\left\{\min\{0,t-d^{\max}_f+\zeta\}\right\}_{\zeta=1}^{d^{\max}_f}. \tag{29}$$

This SoI captures the previous slots that may still have undelivered payloads with some remaining time to the deadlines at the current slot t. Denote the remaining time to the deadline and payload for $\mathrm{I}^{f_c}_u(t-d^{\max}_f+\zeta)$ in current slot t by $\mathrm{T}^{t-d^{\max}_f+\zeta}_{u,\text{rem}}$ and $\mathrm{P}^{t-d^{\max}_f+\zeta}_{u,\text{rem}}$, respectively. Particularly, for all $\mathrm{I}^{f_c}_u(t-d^{\max}_f+\zeta)$, the edge server first checks whether the content is available at the edge server's local cache storage or by the upper layers. If it is available, the edge server calculates the remaining time to the deadlines and payloads for the requests in all slots of $\mathscr{T}^t_{\text{SoI}}$. The edge server finds a set of candidate requester CVs $\mathscr{U}^t_{\text{val}}\subseteq\mathscr{U}$, their minimum remaining time to the deadline set $\mathscr{T}^t_{\text{rem}}$ and corresponding left-over payload set $\mathscr{P}^t_{\text{rem}}$. This procedure is summarized in Algorithm 1. Note that the time complexity of Algorithm 1 is $\mathscr{O}(4U|\mathscr{T}^t_{\text{SoI}}|+3U+5)$.

After extracting the valid CV set $\mathscr{U}^t_{\text{val}}$, the edge server can formulate total $W(t)$ VCs based on the following equation:

$$W(t)=\min\left\{\left|\mathscr{U}^t_{\text{val}}\right|,\mathrm{W}_{\max}\right\}, \tag{30}$$

where $|\mathscr{U}^t_{\text{val}}|$ is the cardinality of the set $\mathscr{U}^t_{\text{val}}$. This essentially means that the server creates the minimum of the total valid CVs in the set $\mathscr{U}^t_{\text{val}}$ and the maximum allowable number of VCs. Besides, the edge server calculates the priorities of the valid CVs set based on their remaining time to the deadlines using the following equation:

$$\phi_u(t)=\bar{\phi}_u(t)/\left(\sum\nolimits_{u\in\mathscr{U}^t_{\text{val}}}\bar{\phi}_u(t)\right), \tag{31}$$

where $\bar{\phi}_u(t)=(\sum_{u\in\mathscr{U}^t_{\text{val}}}\mathscr{T}^t_{\text{val}}[u])/\mathscr{T}^t_{\text{val}}[u]$. Note that (31) sets the highest priority to the CV that has the least remaining time to the deadline. The edge server then picks the top-$W(t)$ CVs for scheduling based on the priorities $\phi_u(t)$s. Denote the scheduled CV set during slot t by $\mathscr{U}^t_{\text{sch}}\subseteq\mathscr{U}^t_{\text{val}}$. Given that the edge server makes scheduling decisions based on top-$W(t)$ priorities of (31), to satisfy the hard deadline constraint in C_{12}, we aim to maximize a WSR, which is calculated as

$$\breve{R}(t)=\sum\nolimits_{u\in\mathscr{U}^t_{\text{sch}}}\mathrm{I}_u(t)\cdot R_u(t)\cdot\phi_u(t), \tag{32}$$

where the weights are set based on the CV's priority $\phi_u(t)$. Again, the intuition for this is that with the underlying RAT solution, due to channel uncertainty, the edge server expects to satisfy constraint C_{12} by prioritizing the CVs based on (31) and follow optimal VC configuration, their association and radio resource allocation. As such, we pose the following WSR maximization problem for the edge server:

$$\underset{\mathscr{B}^a_{vc}(W(t)),\mathrm{I}^{i,a}_u(t),\mathrm{I}^{b,u}_z(t)}{\text{maximize}} \quad \breve{R}(t), \tag{33}$$

$$\text{subject to} \quad C_4,C_5,C_6,C_8,C_9,C_{10},C_{11}, \tag{33a}$$

$$\mathrm{I}^{i,a}_u(t)\in\{0,1\},\mathrm{I}^{b,u}_z(t)\in\{0,1\}, \tag{33b}$$

where the constraints in (33a) and (33b) are taken for the same reasons as in the original problem in (19).

Remark 3: The edge server finds $W(t)$ VCs and $\mathrm{I}_u(t)$s using (30) and (31), respectively. Given that the contents are placed following π_{ca} during slot $t=n\Upsilon$, and the edge server knows $W(t)$ and $\mathrm{I}_u(t)$s, the joint optimization problem in (28) is simplified to a joint VC configuration, CV-VC association and radio resource allocation problem in (33).

VI. Problem Solution

The edge server uses a DRL agent to solve the transformed CHR maximization problem (27). Since the CVs request contents based on the preference-popularity tradeoff and their future demands are unknown to the edge server, DRL is adopted as a sub-optimal learning solution for (27). Moreover, we optimally solve the joint optimization problem (33). In order to do so, first, we leverage graph theory to find optimal pRB allocation based on a given VC configuration $\mathscr{B}^a_{vc}(W(t))\in\mathscr{B}_{vc}(W(t))$. Then we perform a simple linear search to find the best VC configuration $\mathscr{B}^{a^*}_{vc}(W(t))$.

A. Learning Solution for the CPP

To find the CPP π_{ca}, the edge server uses some key information from the environment and learns the underlying environment dynamics. Recall that the CVs requests are modeled by the exploration and exploitation manner. At the beginning of each DoI, the edge server determines top-Λ^c popular contents in each class and also calculates top-Λ^c similar contents for each of these popular contents as

$$\mathbf{F}^{top}(n)[c,f_c]=\begin{cases}1, & \text{if } f_c \text{ is top}-\Lambda^c \text{ similar content of } f^{top}_c,\\ 0, & \text{otherwise,}\end{cases}$$

where f^{top}_c is in top-Λ^c popular content list of class c. Besides, the edge server also keeps track of the content requests coming from each CV and corresponding cache hit based on the stored content during the previous DoI. Let the edge server store

[6]Similar scheduling is also widely used in literature [43], [44].

Algorithm 1: Get Eligible CV Set for Scheduling.

Input: $\mathscr{T}^t_{\text{SoI}}$, $\{\mathrm{I}^{f_c}_u(k)\}_{k\in\mathscr{T}^t_{\text{SoI}}}$, $\{\mathrm{T}^k_{u,\text{rem}}\}_{k\in\mathscr{T}^t_{\text{SoI}}}$, $\{\mathrm{P}^k_{u,\text{rem}}\}_{k\in\mathscr{T}^t_{\text{SoI}}}$

1 Initiate empty valid CV set $\mathscr{U}^t_{\text{val}} = [\cdot]$, valid minimum time to the remaining deadline set $\mathscr{T}^t_{\text{rem}} = [\cdot]$ and valid remaining payload set $\mathscr{P}^t_{\text{rem}} = [\cdot]$;
2 Initiate an initial deadline matrix $\mathbf{T}^t_{rem} \leftarrow Ones(U \times d^{max}_f) \times 100$;
3 Initiate an initial payload matrix $\mathbf{P}^t_{rem} \leftarrow Zeros(U \times d^{max}_f)$;
4 **for** *all k slots in SoI set* $\mathscr{T}^t_{SoI}$ **do**
5 **for** $u \in \mathscr{U}$ **do**
6 **if** $\mathrm{I}^{f_c}_u(k)$ *is available at the edge server in this slot t and* $\mathrm{T}^k_{u,rem} > 0$ and $\mathrm{P}^k_{u,rem} > 0$ **then**
7 $\mathbf{T}^t_{rem}[u,k] \leftarrow \mathrm{T}^k_{u,\text{rem}}$;
8 $\mathbf{P}^t_{rem}[u,k] \leftarrow \mathrm{P}^k_{u,\text{rem}}$;
9 **end**
10 **end**
11 **for** $u \in \mathscr{U}$ **do**
12 Find the maximum remaining payload for CV u in all slots inside the SoI $\mathscr{T}^t_{\text{SoI}}$ as $P^u_{max} = \max\{\mathbf{P}^t_{\text{rem}}[u,:]\}$;
13 **if** $P^u_{max} > 0$ **then**
14 Find the minimum valid remaining time to the deadline $k^{val}_{min} = \min\{\mathbf{T}^t_{rem}[u,:]\}$ and corresponding slot index k^{idx}_{min} ;
15 $\mathscr{U}^t_{\text{val}}$.append$(u)$;
16 $\mathscr{T}^t_{\text{rem}}$.append$(k^{val}_{min})$;
17 $\mathscr{P}^t_{\text{rem}}$.append $(\mathbf{P}^t_{\text{rem}}[u, k^{idx}_{min}])$;
18 **end**

Output: $\mathscr{U}^t_{\text{val}}$, $\mathscr{T}^t_{\text{rem}}$ and $\mathscr{P}^t_{\text{rem}}$

the content-specific request from CV u into a $\mathbb{R}^{C\times F}$ matrix $\mathbf{P}^u_{req}(n)$ during all slots of the DoI. Similarly, let there be a matrix $\mathbf{P}^u_{hit}(n) \in \mathbb{R}^{C\times F}$ that captures content-specific cache hit $\mathbb{1}_{\mathrm{I}^{f_c}_u}(t)$s during all t within the DoI. Furthermore, we also provide the measured popularity matrix $\mathbf{P}_f(n)$ during the current DoI based on the CVs requests in the previous DoI change interval $(n-1)$. As such, the edge server designs **state** x^n_{ca} as the following tuple:

$$x^n_{ca} = \left\{\{\mathbf{P}^u_{req}(n)\}^U_{u=1}, \{\mathbf{P}^u_{hit}(n)\}^U_{u=1}, \mathbf{F}^{top}(n), \mathbf{P}_f(n)\right\}. \tag{34}$$

The intuition behind this state design is to provide the edge server some context on how individual CVs' preferences and global content popularity may affect the overall system reward.

At each $t = n\Upsilon$, the edge server takes a cache placement action m_{ca} to prefetch the contents in its local storage. At the end of the DoI, it gets the following **reward** r^n_{ca}

$$r^n_{ca} = (1/\Upsilon)\sum\nolimits_{\tilde{t}=t}^{(n+1)\Upsilon} r_{ca}(\tilde{t}), \tag{35}$$

where $r_{ca}(\tilde{t}) = \sum_{c=1}^{C}\sum_{f_c=1}^{F_c} r_{ca}[c,f_c]$. Moreover, $r_{ca}[c,f_c]$ is calculated in (36), where δ^{sim}_{pop} and δ_{hit} are two hyper-parameters. Note that these hyper-parameters balance the cache hit for the top-Λ^c contents and the other stored contents in the edge server's cache storage. Empirically, we have observed $\delta^{sim}_{pop} > \delta_{hit}$ works well.

$$r_{ca}[c,f_c](\tilde{t}) = \begin{cases} \delta^{sim}_{pop}\cdot\sum_{u=1}^{U}\mathbb{1}_{\mathrm{I}^{f_c}_u}(\tilde{t}), & \text{if } \mathbf{F}^{\text{top}}(\mathrm{n})\ [\mathrm{c},\ \mathrm{f_c}] = 1 \\ & \text{and } \sum_{\mathrm{u}=1}^{\mathrm{U}} \mathrm{I}^{\mathrm{f_c}}_{\mathrm{u}}(\tilde{\mathrm{t}}) > 0, \\ \delta_{hit}\cdot\sum_{u=1}^{U}\mathbb{1}_{\mathrm{I}^{f_c}_u}(\tilde{t}), & \text{if } \mathbf{F}^{\text{top}}(\mathrm{n})\ [\mathrm{c},\ \mathrm{f_c}] \neq 1 \\ & \text{and } \sum_{\mathrm{u}=1}^{\mathrm{U}} \mathrm{I}^{\mathrm{f_c}}_{\mathrm{u}}(\tilde{\mathrm{t}}) > 0, \\ -\sum_{f_c=1}^{F}\sum_{u=1}^{U}\mathrm{I}^{f_c}_u(\tilde{t}), & \text{otherwise,} \end{cases} \tag{36}$$

We consider that the edge server learns the CPP π_{ca} offline. It uses two DNNs - $\boldsymbol{\theta}_{ca}$ and $\boldsymbol{\theta}^-_{ca}$, and learns π_{ca} following the basic principles described in Section IV-C. Algorithm 2 summarizes the CPP learning process. While the training episode is not terminated, in line 6, the CVs make content requests. During the cache placement slots $t = n\Upsilon$, line 7, the edge server observes its state x^n_{ca} in line 8. Based on the observed state, the agent takes action m_{ca} following the ϵ-greedy policy [40] using $\boldsymbol{\theta}_{ca}$ in line 9. During the last time slot of the current DoI, in line 11, the environment returns the reward r^n_{ca} and transits to the next state $x^{n'}_{ca}$ in line 12. Moreover, in line 13, the edge server stores its experiences tuple $\{x^n_{ca}, m_{ca}, r^n_{ca}, x^{n'}_{ca}\}$ into its memory buffer mem_{ca}, which can hold $mem^{\max}_{ca}$ number of samples. In line 15, the edge server randomly sample S_{ca} batches from mem_{ca} and uses the $\boldsymbol{\theta}_{ca}$ and $\boldsymbol{\theta}^-_{ca}$ to get $Q(x^n_{ca}, m_{ca}; \boldsymbol{\theta}_{ca})$ and the target value $\bar{y}_t(\boldsymbol{\theta}^-)$, respectively. In line 16, it then trains the DNN $\boldsymbol{\theta}_{ca}$ by minimizing the loss function shown in (24) using SGD. Moreover, after $\breve{\eta}_{ca}$ steps, the offline DNN $\boldsymbol{\theta}^-_{ca}$ gets updated by $\boldsymbol{\theta}_{ca}$ in line 20.

B. WSR Maximization

Recall that once the edge server determines $W(t)$ based on (30), all possible VC configurations $\mathscr{B}_{vc}(W(t)) = \{\mathscr{B}^a_{vc}(W(t))\}^{A_{W(t)}}_{a=1}$ can be generated following the VC formation rules defined in (1a)-(1c). Besides, each VC configuration $\mathscr{B}^a_{vc}(W(t))$ has exactly $W(t)$ number of VCs. Moreover, the edge server schedules $|\mathscr{U}^t_{\text{sch}}| = W(t)$ CVs in each slot t based on the priority $\phi_u(t)$. Let the i^{th} CV in $\mathscr{U}^t_{\text{sch}}$ be assigned to the i^{th} VC in $\mathscr{B}^a_{vc}(W(t))$. This assigns each CV to exactly one VC and all VCs are assigned to all scheduled CVs. Therefore, essentially, for a selected VC configuration $\mathscr{B}^a_{vc}(W(t))$, by assigning the VCs in the above mentioned way, the edge server can satisfy constraints C_3, C_4, C_5 and C_6. To this end, given that the selected VC configuration $\mathscr{B}^a_{vc}(W(t))$ and $\mathrm{I}^{i,a}_u(t)$ are known at the edge server, we can rewrite (33) as follows:

$$\underset{\mathrm{I}^{b,u}_z(t)}{\text{maximize}} \quad \breve{R}(t), \tag{37}$$

$$\text{subject to} \quad C_8, C_9, C_{10}, C_{11}, \mathrm{I}^{b,u}_z(t) \in \{0,1\}. \tag{37a}$$

As the CSI is perfectly known at the edge server, it can choose maximal ratio transmission to design the precoding vector $\mathbf{w}^{u,z}_b$. In other words, given $\mathrm{I}^{b,u}_z(t) = 1$, the edge server chooses $\mathbf{w}^{u,z}_b(t) = \mathbf{h}^{u,z}_b(t)/\|\mathbf{h}^{u,z}_b(t)\|$. Besides, the received SNR at the CV u, calculated in (9), is the summation over all APs of the CV's assigned VC divided by total noise power. As such, we can stack the weighted data rate at the CV from the APs that are in its serving VC over all pRBs into a matrix - denoted by $\mathbf{R}_t \in \mathbb{R}^{B\times Z}$ matrix. This weighted data rate matrix extraction process is summarized in Algorithm 3. In this algorithm, we initiate a matrix of zeros of $\mathbb{R}^{B\times Z}$ in line 1. Recall that all VCs are assigned to the scheduled CVs and all APs are assigned to form the VCs based on the rules defined in Section II-A. As such, for each $u \in \mathscr{U}^t_{sch}$, we get the assigned VC in line 3. Then, for all APs and all pRBs, we calculate the spectral efficiency in line 6. Moreover, we update the respective (b,z) element of the

Algorithm 2: CPP Learning Algorithm.

Input: S_{ca}, Mem_{ca}, $\check{\eta}_{ca}$, ε_{max}, ε_{min}, ν, T_{epoch}, Υ, $\boldsymbol{\theta}_{ca}$, $\boldsymbol{\theta}^{-}_{ca}$

Calculate ε decaying rate as $decay_{\varepsilon} = \frac{\varepsilon_{max}-\varepsilon_{min}}{\nu \times T_{epoch}}$;
for *e in* T_{epoch} **do**
  $\varepsilon \leftarrow max\{\varepsilon_{min}, \varepsilon_{max} - (e \times decay_{\varepsilon})\}$;
  Set $t = 0$, $n = 0$, *done*=False ;
  while *not done* **do**
    Get all CVs content requests using Section III-C ;
    if $t == 0$ *or* $((t+1) \mod \Upsilon) == 0$ **then**
      Get state x^{n}_{ca} from the environment ;
      Edge server takes cache placement action m_{ca} based on observation x^{n}_{ca} using its action selection policy ;
      $n \mathrel{+}= 1$;
    else if $(t+1) == (n\Upsilon - 1)$ **then**
      Get r^{n}_{ca}, $x^{n'}_{ca}$ and *done* flag from environment ;
      Store $\{x^{n}_{ca}, m_{ca}, r^{n}_{ca}, x^{n'}_{ca}, done\}$ into Mem_{ca} ;
      if $len(Mem_{ca}) \geq S_{ca}$ **then** // train $\boldsymbol{\theta}_{ca}$
        Uniformly sample S_{ca} samples from Mem_{ca} ;
        Use the sampled samples to train $\boldsymbol{\theta}_{ca}$;
      $x^{n}_{ca} \leftarrow x^{n'}_{ca}$;
    $t \mathrel{+}= 1$;
    if $t \mod \check{\eta}_{ca} == 0$ **then** // Update $\boldsymbol{\theta}^{-}_{ca}$
      $\boldsymbol{\theta}^{-}_{ca} \leftarrow \boldsymbol{\theta}_{ca}$
  end
end
Output: $\boldsymbol{\theta}_{ca}$

$\mathbf{R}_t$ matrix in line 7. Note that Algorithm 3's time complexity is $\mathcal{O}(W(t)[2Z|VC^{i}_{a}| + 1] + 1)$.

Algorithm 3: Get Weighted Data Rate Matrix.

Input: $\mathscr{U}^{t}_{\text{sch}}$, $\mathscr{B}^{a}_{vc}(W(t))$, $\{\phi_u(t)\}_{u \in \mathscr{U}^{t}_{\text{sch}}}$, $\mathbf{H}^{b}_{u}(t)$

Initiate matrix $\mathbf{R}_t = zeros(B \times Z)$;
for $u \in \mathscr{U}^{t}_{sch}$ **do**
  Get assigned VC^{i}_{a} using $\mathrm{I}^{i,a}_{u}(t)$;
  for $b \in VC^{i}_{a}$ **do**
    for $z \in \mathscr{Z}$ **do**
      Calculate $r_t(u,b,z) = \log_2\left(1 + \frac{P_b \left|\mathbf{h}^{u,z}_{b}(t)^{H} \mathbf{w}^{u,z}_{b}(t)\right|^2}{\sigma^2_b}\right)$ received at CV u from AP b over pRB z during slot t ;
      $\mathbf{R}_t[b,z] = r_t(u,b,z) \times \phi_u(t)$;
    end
  end
end
Output: $\mathbf{R}_t$

Algorithm 4: Optimal VC Configuration and pRB Allocation.

Input: $W(t)$, $\mathscr{U}^{t}_{\text{sch}}$, $\{\phi_u(t)\}_{u \in \mathscr{U}^{t}_{\text{sch}}}$, $\mathbf{H}^{b}_{u}(t)$, $\mathscr{B}_{vc}(W(t))$

Initiate WSR vector $\check{\mathbf{r}}_t = zeros(A_{W(t)})$ and empty pRB allocation set $\mathbf{I}_z(t) = [\cdot]$;
for $\mathscr{B}^{a}_{vc}(W(t)) \in \mathscr{B}_{vc}(W(t))$ **do**
  Get $\mathbf{R}_t$ matrix from Algorithm 3 for this VC configuration ;
  Solve the MWBM problem using Hungarian algorithm [46] to get optimal pRB allocation set $\boldsymbol{I}^{*}_{z} = \{\mathrm{I}^{b,u}_{z}(t)\}^{Z}_{z=1}$ and get the optimal sum-weights r^{*}_{t} from the optimal edges $e^{*}(b,z)$;
  $\check{\mathbf{r}}_t[a] = r^{*}_{t}$;
  $\mathbf{I}_z(t).append(\boldsymbol{I}^{*}_{z})$;
end
Find the max$(\check{\mathbf{r}}_t)$ and corresponding index a^{*} ;
Take best VC configuration $\mathscr{B}^{a^*}_{vc}(W(t))$ and corresponding optimal pRB allocation set $\mathbf{I}^{b,u^*}_{z}(t) = \mathbf{I}_z(t)[a^{*}]$;
Output: $\mathscr{B}^{a^*}_{vc}(W(t))$ and $\mathbf{I}^{b,u^*}_{z}(t)$

Upon receiving $\mathbf{R}_t$, the edge server leverages graph theory to get the optimal assignment as follows. It forms a bipartite graph $G = (\mathscr{B} \times \mathscr{Z}, \mathscr{E})$, where $\mathscr{B}$ and $\mathscr{Z}$ are the set of vertices, and $\mathscr{E}$ is the set of edges that can connect the vertices [47]. Moreover, $R_t[b,z]$ are the weights of edge $e(b,z)$ that connects vertex $b \in \mathscr{B}$ and $z \in \mathscr{Z}$. Note that, for the graph G, a matching is a set of pair-wise non-adjacent edges where no two edges can share a common vertex. This is commonly known as the maximum weighted bipartite matching (MWBM) problem [47]. The edge server needs to find the set of edges $e^{*}(b,z) \in \mathscr{E}$ that maximizes the summation of the weights of the edges. Moreover, the edge server uses well-known Hungarian algorithm [46] to get the optimal edges $e^{*}(b,z)$, i.e., pRB allocations $\mathrm{I}^{b,u}_{z}(t)$s in polynomial time. This pRB allocation is, however, optimal only for the selected VC configuration $\mathscr{B}^{a}_{vc}(W(t))$. In order to find the best VC configuration $\mathscr{B}^{a^*}_{vc}(W(t))$, the edge server performs a simple linear search over all $A_{W(t)}$s VC configurations. As such, we can solve problem (33) optimally using the above techniques. Algorithm 4 summarizes the steps. Note that Algorithm 4 has a time complexity of $\mathcal{O}(A_{W(t)}[W(t)(2Z|VC^{i}_{a}| + 1) + Z^3 + 4] + 3)$.

Algorithm 5: Content Delivery Model.

Input: $\mathrm{I}^{f_c}_{u}(t)$'s of all CVs in content delivery slot t

Check if the requested contents are in the cache storage, if any requested content is not available, forward the request to upper layer for extraction from cloud;
Calculate SoI $\mathscr{T}^{t}_{\text{SoI}}$ using (29);
Find eligible CV set $\mathscr{U}^{t}_{\text{val}}$ using Algorithm 1;
Find total number of VC to formulate, i.e., $W(t)$ using (30);
Calculate eligible CVs', i.e., $u \in \mathscr{U}^{t}_{\text{val}}$, priorities using (31);
Get the CV set $\mathscr{U}^{t}_{\text{sch}}$ to schedule by picking the top-$W(t)$ $\phi_u(t)$s;
Find optimal VC configuration $\mathscr{B}^{a^*}_{vc}(W(t))$ and optimal pRB allocations $\mathbf{I}^{b,u^*}_{z}(t)$ by running Algorithm 4;
Based on VC configuration $\mathscr{B}^{a^*}_{vc}(W(t))$ and $\mathbf{I}^{b,u^*}_{z}(t)$ calculate CVs SNRs $\boldsymbol{\Gamma}_t = \{\Gamma^{z}_{u}(t)\}_{u \in \mathscr{U}^{t}_{\text{sch}}}$ using (9);
Calculate $R^{bit}_{u}(t)$ using (38) for all $u \in \mathscr{U}^{t}_{\text{sch}}$;
Offload $R^{bit}_{u}(t)$ bits from the remaining payloads of all CVs $u \in \mathscr{U}^{t}_{\text{sch}}$ orderly from the requests made in the SoIs $\mathscr{T}^{t}_{\text{SoI}}$;
Update all $u \in \mathscr{U}$ remaining payload and deadline;

C. Content Delivery Process

Contents are placed using the trained CPP π_{ca} during each cache placement slot $t = n\Upsilon$, while the CVs make content requests in each t following Section III-C. Please note that, during $t = n\Upsilon$, the edge server only requires to perform one forward pass[7] on the trained $\boldsymbol{\theta}_{ca}$. Upon receiving the $\mathrm{I}^{f_c}_{u}(t)$s, the edge server checks whether $\mathrm{I}_{f_c}(n) = 1$ or $\mathrm{I}_{f_c}(n) = 0$. If $\mathrm{I}_{f_c}(n) = 1$, f_c can be delivered locally. All cache miss events are forwarded to the VEN's upper layers. The upper layer extracts each cache missed content from the cloud with an additional delay of $d^{m,t}_{u,f_c}$. In all t, the edge server calculates the SoI $\mathscr{T}^{t}_{\text{SoI}}$ using (29). It then finds the eligible CV set $\mathscr{U}^{t}_{\text{val}}$ and forms total $W(t)$ VCs using Algorithm 3 and (30), respectively. To that end, the edge server calculates the priorities $\phi_u(t)$s using (31) and selects top-$W(t)$ CVs to schedule. Once the edge server knows $W(t)$, $\phi_u(t)$s and $\mathscr{U}^{t}_{\text{sch}}$, it runs Algorithm 4 to get the VC configuration and pRB allocations that maximizes the WSR of (33). Algorithm 4 returns the $\mathscr{B}^{a^*}_{vc}$ and $\mathbf{I}^{b,u^*}_{z}(t)$ which then can be

[7]The time complexity of the forward pass depends on the input/output size and DNN architecture.

TABLE II
SYSTEM PARAMETERS

Item/Description	Value
Total number of APs B	6
Maximum possible VC per slot W_{max}	5
TTI κ	1 ms
DoI update interval Υ	$50 \times \kappa$
Carrier frequency	2 GHz
pRB size ω	180 KHz
Noise power σ^2	-174 dBm/Hz
AP coverage radius	250 m
Antenna/AP L	4
AP antenna height	25 m
CV antenna height	1.5 m
Transmission power P_b	30 dBm
AP transmitter antenna gain G^b_{TX}	8 dBi
CV receiver antenna gain G^u_{RX}	3 dBi
CV receiver noise figure L^u_{RX}	9 dB
Total content class c	3
Contents per class $\lvert\mathscr{F}_c\rvert$	5
Feature per content G_c	10
AN cache size Λ	$\{3,6,9,12\} \times S$
Max allowable delay d_f^{max}	$10 \times \kappa$
Content extraction delay $d^{m,t}_{u,fc}$	$5 \times \kappa$
CV active probability p_u	Uniform$(0.1, 1)$
CV's inclination to similarity/popularity ε_u	Uniform$(0, 1)$

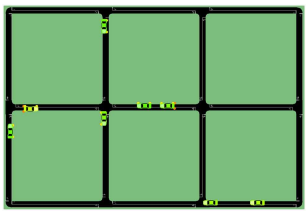

Fig. 2. Simulated RoI.

used to get the SNRs $\Gamma^z_u(t)$s from (9). Upon receiving the SNRs $\Gamma^z_u(t)$s, the edge server can calculate the possible transmitted bits for the CVs as follows:

$$R^{bit}_u(t) = \kappa \cdot R_u(t). \tag{38}$$

The edge server then delivers the remaining $\mathrm{P}^{t-d^{\max}_f+\zeta}_{u,\mathrm{rem}}$s sequentially. This entire process is summarized in Algorithm 5. The time complexity of running Algorithm 5 is $\mathscr{O}(U[\Lambda/S + 4|\mathscr{T}^t_{\mathrm{SoI}}| + 4] + W(t)[\log(|\mathscr{U}^t_{\mathrm{val}}|) + A_{W(t)}(2Z|VC^i_a| + 1) + 3] + A_{W(t)}[Z^3 + 4] + |\mathscr{U}^t_{\mathrm{val}}| + |\mathscr{T}^t_{\mathrm{SoI}}| + 10)$.

VII. PERFORMANCE EVALUATION

A. Simulation Setting

We consider U CVs roam over a region of interest (RoI) and deploy $B = 6$ APs alongside the road to cover the entire RoI. Table II shows other key simulation parameters used in this article. We consider a 300 meters by 200 meters Manhattan grid model [34] with two-way roads as shown in Fig. 2. For realistic microscopic CV mobility modeling, we use the widely known simulation of urban mobility (SUMO) [48]. The CVs are deployed with some initial routes with a maximum speed of 45 miles/hour and later randomly rerouted from the intersections on this RoI. In SUMO, we have used car-following mobility model [49] and extracted the CVs' locations using the Traffic Control Interface [50] application programming interface.

To design our simulation episode, we consider 1000κ milliseconds of CVs activities. For the CPP learning, the edge server uses DNN $\boldsymbol{\theta}_{ca}$ that has the following architecture: 2D convolution (*Conv2d*) $\rightarrow$ *Conv2d* $\rightarrow$ *Linear* $\rightarrow$ *Linear*. We train $\boldsymbol{\theta}_{ca}$ in each cache placement slots with a batch size $S_{ca} = 512$. Besides, we choose $\gamma = 0.995$, $\epsilon_{\max} = 1$, $\epsilon_{\min} = 0.005$, $\nu = 0.6$, $Mem^{\max}_{ca} = 15000$, $T_{epoch} = 15000$, $\breve{\eta}_{ca} = 4\Upsilon$. For training, we use $Adam$ as the optimizer with a learning rate of 0.001. Using our simulation setup, the edge server first learns π_{ca} using Algorithm 2 for T_{epoch} episodes. The average per state returns during this learning is shown in Fig. 3. As the training progresses, we observe that the edge server learns to tune its policy to maximize the expected return. After sufficient exploration, the edge server is expected to learn the CPP that gives the maximized expected return. As a result, it is expected that the reward will increase as the learning proceeds. Fig. 3 also validates this and shows the convergence of Algorithm 2. As such, we use this trained CPP π_{ca} for performance evaluations in what follows.

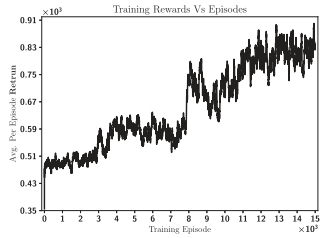

Fig. 3. CPP learning: average return during training.

B. Performance Study

We first show the performance comparisons of the learned CPP with the following baselines without any RAT solution.

Genie-Aided cache replacement (Genie): The to-be requested contents are known beforehand during the start of the DoI provided by a Genie. In this best case, we then store the top-Λ^c requested contents from all $c \in \mathscr{C}$ in all n.

Random cache replacement (RCR): In this case, contents from each class are selected randomly for cache placement.

K-Popular (K-PoP) replacement [51]: In this popularity-based caching mechanism, we store the most popular $K = \Lambda^c$ contents during the past DoI for each content class $c \in \mathscr{C}$.

Modified K-PoP+LRU (K-LRU) replacement: We modify the popularity-driven K-PoP with classical least recently used (LRU) [52] cache replacement. The least popular contents in the K-PoP contents are replaced by the most recently used but not in K-PoP contents to prioritize recently used contents.

To this end, we vary the cache size of the VEN and show the average CHR during an episode in Fig. 4(a). The general intuition is that when we increase the cache size Λ, more contents can be placed locally. Therefore, by increasing Λ, the average CHR is expected to increase. K-PoP and K-LRU do not capture the heterogeneous preferences of the CVs. Similarly, as contents are replaced randomly with the naive RCR baseline, it should perform poorly. However, when the cache size is relatively small, solely popularity-based K-PoP performs even worse than RCR. This means that popularity does not dominate the content demands of the CVs. Moreover, when the cache size becomes moderate, K-PoP and K-LRU outperform the naive

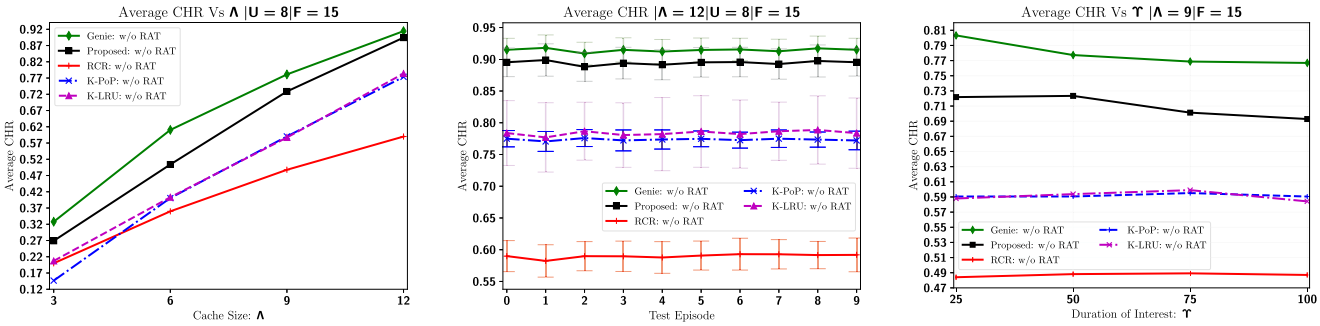

Fig. 4. (a) CHR comparison with baselines when $\Upsilon = 50 \times \kappa$ (without RAT). (b) CHR comparison with baselines for 10 test episodes when $\Upsilon = 50 \times \kappa$ (without RAT). (c) CHR analysis for different DoI when $U = 8$ (without RAT).

RCR baseline. On the other hand, the proposed CPP aims to optimize π_{ca} by capturing the underlying preference-popularity tradeoff of the CVs. Therefore, the average CHR is expected to be better than the baselines. Fig. 4(a) also reflects these analysis. Moreover, notice that the performance gap with the Genie-aided average CHR and our proposed CPP is lower. In the VEN, we do not know the future and CVs' content demands. Therefore, we can only predict the future and tune the CPP π_{ca} accordingly. Particularly, when the cache storage is reasonable, the performance gap of the proposed CPP is much lower. For example, at $\Lambda = 9$ and $\Lambda = 12$ the proposed CPP delivers around 93% and 98% of the Genie-aided solution. Moreover, the baselines perform poorly regardless of Λ. For example, at $\Lambda = 9$, the proposed CPP is around 49%, 23% and 24% better than RCR, K-PoP and K-LRU, respectively.

Fig. 4(b) shows the average CHR variation over 10 test episode in 100 simulation runs and corresponding standard deviations. As expected, the performance of the proposed CPP is very close to the Genie-aided performance in these test runs. Particularly, the proposed CPP delivers around 98% of the Genie-aided performance. Moreover, the other baselines' average CHRs largely deviate from the Genie-aided solution. We observe that the proposed CPP is around 52%, 16% and 14% better than RCR, K-PoP and K-LRU, respectively, even when Λ is 80% of the content catalog $\mathcal{F}$, which validate the effectiveness of the proposed method.

To this end, we study the impact of different DoI Υ on the CHR. Recall that the DoI is the period for which the contents in the library remain fixed. A shorter DoI means that the content catalog can be refreshed quickly. Besides, based on our content request model, each CV's content choices change fewer times within this short interval. Hence, the edge server can quickly accommodate the CPP to capture the future demands of the CVs. This, thus, may yield better CHR. On the other hand, when this period is extended, performance is expected to deteriorate slightly. This is due to the fact that the cache storage cannot be replaced until this DoI period expires, while the CVs' requests vary in each slot. We also observe similar trends in our simulation results. Fig. 4(c) shows CHR for different DoI, where we observe that even the Genie-aided performance degrades from 80% to 76% when the DoI is increased from $25 \times \kappa$ to $100 \times \kappa$. We also observe that our proposed CPP experiences only about 4% performance degradation. Moreover, the performance improvement of our proposed solution is about 49%, 22% and 23% at $\Upsilon = 25 \times \kappa$ and about 42%, 17% and 18% at $\Upsilon = 100 \times \kappa$, respectively, over the RCR, K-Pop and K-LRU baselines. Note that we leave the choice of DoI as a design parameter chosen by the system administrator, which can be decided based on the practical hardware limitations and other associated overheads in the network. As such, we fix $\Upsilon = 50 \times \kappa$ for the rest of our analysis.

As content requests arrive following preference-popularity tradeoff, the CHR also gets affected by the total number of CVs in the VEN. Intuitively, as the CVs' preferences are heterogeneous, when the total number of CVs in the VEN increases, the content requests largely diversify. Therefore, even with the Genie-aided solution, the CHR may degrade when the number of CVs in the VEN increases. This is also reflected in our simulated results in Fig. 5(a). The performance of the proposed CPP algorithm is stable regardless of the number of CVs in the VEN. We observe a slight performance gap between the CPP and the Genie-aided solution. This gap gets smaller and smaller as the total number of CVs in the VEN increases. Particularly, we observe that the proposed CPP delivers an average 97% CHR for the considered CV numbers. Besides, it delivers around 47%, 21% and 22% better performance than RCR, K-PoP and K-LRU, respectively. Therefore, we will use this CPP π_{ca} to find $\mathrm{I}_{f_c}(n)$ for all n and show performance analysis of our proposed user-centric RAT solution.

To that end, we compare the performance of the proposed RAT solution with legacy network-centric RAT (NC-RAT). In the NC-RAT, a base station (BS) is located at a fixed suitable location which has $Z = 6$ pRBs and total transmission power of 46 dBm. We use the same scheduling and deadline-based priority modeling for the NC-RAT as the proposed user-centric case. Besides, we distributed the total transmission power proportionally to the scheduled CVs' priorities. Moreover, we performed the same WSR maximization problem for getting the pRB allocation using Hungarian algorithm [46]. In the following, this legacy RAT solution is termed NC-RAT and used with the cache placement baselines. On the other hand, the 'Proposed' method uses the proposed CPP and user-centric RAT solution.

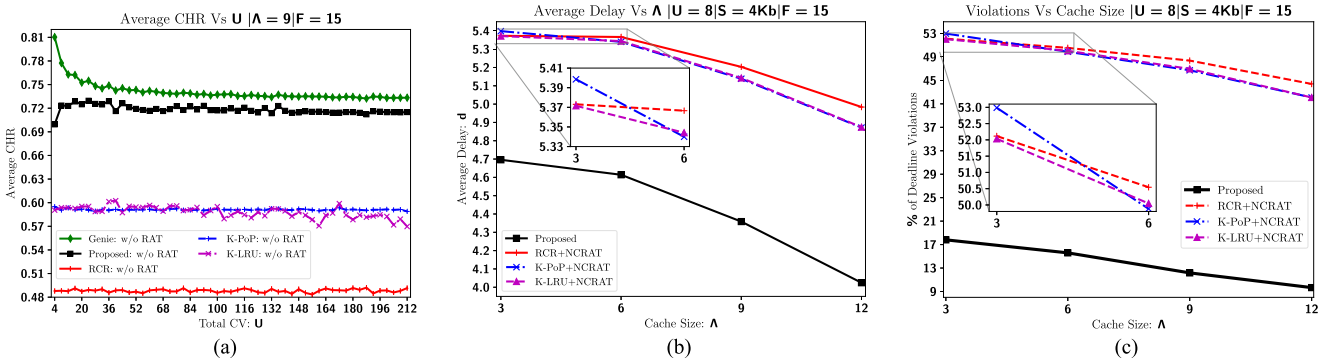

Fig. 5. (a) Deadline violation percentage comparisons for different content size S. (b) Average content delivery delay comparison with NC-RAT and caching baselines. (c) Deadline violation percentage comparison with legacy NC-RAT.

Intuitively, with an increased Λ, the edge server can store more contents locally which increases the total number of local delivery by assuring lower cache miss events. Therefore, with a proper RAT solution, the content delivery delay is expected to decrease if we increase the cache size of the edge server. We also observe this trend with both NC-RAT and our proposed user-centric RAT solution in Fig. 5(b). However, note that NC-RAT is inflexible, and depending on the location of the CVs, NC-RAT may not even have expected radio-link qualities. This can, therefore, cause link failure and may increase the content delivery delay for the CVs' requested content. On the other hand, the proposed user-centric RAT solution can design the appropriate VC configuration, VC associations and proper radio resource allocation to deliver the content timely. Therefore, we expect the user-centric RAT solution to outperform the traditional NC-RAT. Fig. 5(b) shows the average content delivery delay $\mathrm{d} = \frac{1}{T}\sum_{t=1}^{T} \bar{\mathrm{d}}(t)$, where $\bar{\mathrm{d}}(t)$ is calculated in (18) with $\mathrm{W}_{\max} = 5$. As we can see, the proposed solution outperforms the baselines. Particularly, the average gain of the proposed solution on content delivery delay is around 15% over the baselines.

The effectiveness of the proposed solution is more evident in Fig. 5(c), which shows the percentage of deadline violations in a test episode when the content size is $S = 4$ KB. As a general trend, the deadline violations decrease as Λ increases. Besides, among the cache placement baselines, as we have seen in the performance comparisons of the CPP, even RCR delivers lower deadline violations than solely popularity-based K-PoP when the cache size is small. Moreover, we observe around 28% higher deadline violations with the baseline NC-RAT over our proposed user-centric RAT solution. Recall that this deadline violation is essentially the violation of constraint C_{12}, which means the requester CVs have not received the requested content by their required deadlines. As such, these requester CVs may experience fatalities and degraded QoEs with the existing RAT and cache placement baselines.

Content size S also affects the delivery delays and corresponding deadline violations. Intuitively, content delivery delay shall increase if the payload increases when the network resources are unchanged. This also increases the likelihood of deadline violations. Fig. 6 shows how the delivery delay gets affected by content size S. Note that transmission delay is directly related

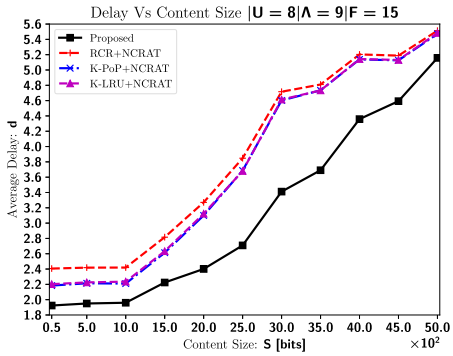

Fig. 6. Average content delivery delay comparisons for different content sze S.

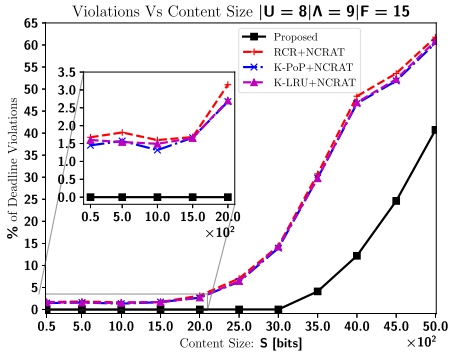

Fig. 7. Deadline violation percentage comparisons for different content size S.

to channel quality between the transmitter and receiver. This channel uncertainty can cause fluctuations in the content delivery delay. However, the general expectation is that the content delivery delay will increase when the payload size increases. We also observe these in Fig. 6. Particularly, when $S = 2.5$ KB, the performance gain of the proposed solution is around 30% over the RCR+NCRAT and around 27% over the K-PoP+NCRAT and K-LRU+NCRAT baselines.

Recall that delay cannot exceed the hard deadline. Therefore, higher content delivery delay leads to deadline violations. Fig. 7 shows how the payload size affects the deadline violations in

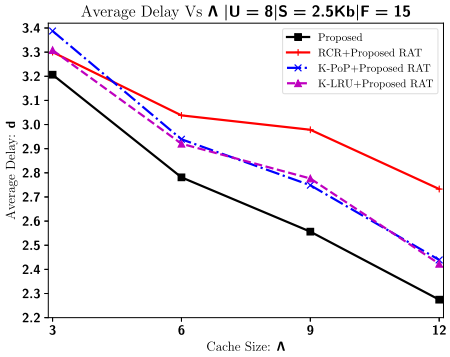

Fig. 8. Average content delivery delay for different Λ.

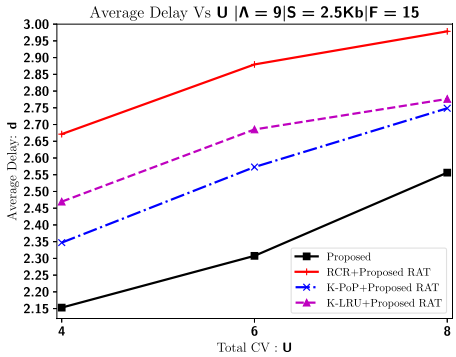

Fig. 9. Average delay for different number of CVs.

the proposed VEN. As expected, even when the payload size is small, we observe that the legacy NC-RAT solution cannot ensure guaranteed delivery within the deadline. On the contrary, our proposed solution can ensure 0% deadline violations till $S = 3$ KB. Moreover, when S increases, the deadline violation percentage of our proposed solution performs significantly better than the NCRAT-based baselines. For example, when $S = 4$ KB, the deadline violation percentage with our proposed solution is around 12%, whereas the NCRAT-based baselines have around 47% deadline violations. From Fig. 5(b)–7, we can clearly see that the traditional NC-RAT is not sufficient to deliver the demands of the CVs.

To that end, we show the efficacy of the proposed RAT solution by considering all cache placement baselines accompanied by the proposed RAT solution for delivering the requested contents of the CVs. Fig. 8 shows how the content delivery delay gets affected by different cache sizes. Particularly, the proposed solution delivers requested contents around 14%, 7% and 8% faster than the RCR+Proposed-RAT, K-PoP+Proposed-RAT and K-LRU+Proposed-RAT, respectively, when $\Lambda = 9$. Recall that the proposed CPP (without RAT) had a performance gain of around 49%, 23% and 24% over the RCR, K-LRU and K-PoP, respectively. The proposed RAT can, thus, significantly compensate for the cache miss events.

Moreover, Fig. 9 shows delay vs total number of CVs U in the VEN. Intuitively, if U increases, the edge server receives a larger number of content requests. Then, with the limited VCs, the edge server can at max schedule only $\mathrm{W}_{\max}$ number of CVs. Therefore, d is expected to increase if U increases, which is also reflected in Fig. 9. Notice that in both Figs. 8 and 9, while the proposed solution outperforms the other cache placement baselines, the performance gaps are small because all cache placement baselines now use our proposed user-centric RAT solution for delivering the requested contents.

Finally, our extensive simulation results suggest that the CHR from our proposed CPP is very close to the genie-aided solution, while the baseline RCR, K-Pop and K-LRU cache placements yield poor CHRs. Besides, when we use the proposed CPP with our user-centric RAT solution, the performance improvements, in terms of deadline violation percentage and content delivery delay, are significant compared to existing legacy NC-RAT with the above cache placement baseline solutions. Additionally, having a larger cache storage size increases the CHR, while having more CVs in the VEN leads to a slightly degraded CHR for all cache placement strategies. Moreover, with fixed limited radio resources, content delivery delays grow, which increases the deadline violation percentage.

VIII. Conclusion

Considering the higher automation demand on the road, in this article, we propose a user-centric RAT solution for delivering the CVs requested content with a learning solution for the cache placement. From the results and analysis, we can conclude that existing cache placement baselines may not be sufficient to capture the heterogeneous demands and preferences of the CVs. Moreover, the existing NC-RAT may cause severe fatalities on the road as it yields frequent deadline violations. Even for continuous deadline-constrained demand arrivals in each time slot, the proposed software-defined user-centric RAT solution has shown significant potential for offloading the payloads timely. The results suggest that our proposed cache placement policy delivers practical near-optimal cache hit ratio while the proposed user-centric RAT efficiently delivers the requested contents within the allowable deadline.

Appendix A
Proof of Proposition 1

Assuming $\iota > 0$, we write the following:

$$\begin{aligned}
&\Pr\{\Psi_t \geq \xi\} = \Pr\left\{e^{\iota\Psi_t} \geq e^{\iota\xi}\right\} \overset{(a)}{\leq} \left(\mathbb{E}\left[e^{\iota\Psi_t}\right]\right)/e^{\iota\xi},\\
&\overset{(b)}{=} e^{-\iota\xi}\prod_{u=1}^{U}\mathbb{E}\left[e^{\iota\Theta_u^t}\right] \overset{(c)}{=} e^{-\iota\xi}\prod_{u=1}^{U}\left(1-p_u+p_u e^{\iota}\right),\\
&\overset{(d)}{\leq} e^{-\iota\xi}\left[\frac{\sum_{u=1}^{U}\left(1-p_u+p_u e^{\iota}\right)}{U}\right]^{U} = e^{-\iota\xi}\left[1-\bar{p}+\bar{p}e^{\iota}\right]^{U},\\
&= \exp\left[-\iota\xi + U\ln\left(1-\bar{p}+\bar{p}e^{\iota}\right)\right], \qquad (39)
\end{aligned}$$

where (a) follows Markov inequality, (b) is true as Θ_u^ts are independent and identically distributed, (c) follows as $\mathbb{E}[e^{\iota\Theta_u^t}]$ is the moment generating function of Θ_u^t, and (d) is obtained following the inequality of arithmetic and geometric means.

To this end, we find $e^{\iota} = \frac{\xi(1-\bar{p})}{\bar{p}(U-\xi)}$ that minimizes (39). Plugging this value in (39), we obtain the bound as

$$\Pr\{\Psi_t \geq \xi\} \overset{(a)}{\leq} \exp\left[U\left\{\ln\left(\frac{(1-\bar{p})}{1-\chi}\right) - \chi\ln\left(\frac{\chi(1-\bar{p})}{\bar{p}(1-\chi)}\right)\right\}\right],$$
$$= \exp\left[-UD_{\bar{p}}(\chi)\right], \tag{40}$$

where $\chi = \frac{\xi}{U}$ in (a) and $D_{\bar{p}} = \chi\ln\left(\frac{\chi}{\bar{p}}\right) + (1-\chi)\ln\left(\frac{1-\chi}{1-\bar{p}}\right)$.

Appendix B Proof of Theorem 1

We show that an instance of our problem in (27) reduces to an instance of a well-known NP-hard problem. Particularly, we only consider a single cache placement step $t = n\Upsilon$ and assume that $\mathrm{I}_u^{f_c}(t)$s, $\forall t \in [n\Upsilon, (n+1)\Upsilon]$ are known at the edge server beforehand[8]. Then, we re-write our (27) instance as

$$\underset{\mathrm{I}_{f_c}(n);\forall f_c\in\mathscr{F}}{\text{maximize}} \sum\nolimits_{t\in[n\Upsilon,(n+1)\Upsilon]} \mathrm{CHR}(t), \tag{41}$$

$$\sum_{c=1}^{C}\sum_{f_c\in\mathscr{F}_c} S\cdot\mathrm{I}_{f_c}(n) \leq \Lambda, \sum_{f_c\in\mathscr{F}_c} S\cdot\mathrm{I}_{f_c}(n) = \Lambda^c, \tag{41a}$$

$$\mathrm{I}_{f_c}(n) \in \{0,1\}, \forall c = 1,\ldots,C; f_c \in \mathscr{F}_c, \tag{41b}$$

where the constraints are taken for the same reasons as in (27).

To that end, if $\Lambda^c = S\cdot 1$, we could rewrite the second constraint as $\sum_{f_c\in\mathscr{F}_c}\mathrm{I}_{f_c}(n) = 1$. Then, it is easy to recognize that an instance of the well-known multiple-choice knapsack problem (MCKP) [53] has reduced to this instance of our CHR maximization problem. As MCKP is a well-known NP-hard problem [53], we conclude that the cache placement problem for each $t = n\Upsilon$ is NP-hard even when the to-be requested contents are known beforehand. As such, the long-term policy optimization problem in (27) is NP-hard.

[8]This assumption is only for the sake of this proof. The edge server does not know the future.

References

[1] "Automated vehicles safety," National Highway Traffic Safety Administration. Accessed: Aug. 8, 2023. [Online]. Available: https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety

[2] "Connected and automated vehicles in the U.K.: 2020 information booklet," Centre for connected and autonomous vehicles, Department for Transportation U.K., London, U.K. Accessed: Aug. 8, 2023. [Online]. Available: https://www.gov.uk/government/publications/connected-and-automated-vehicles-in-the-uk-2020-information-booklet

[3] X. Li, L. Cheng, C. Sun, K.-Y. Lam, X. Wang, and F. Li, "Federated-learning-empowered collaborative data sharing for vehicular edge networks," *IEEE Netw.*, vol. 35, no. 3, pp. 116–124, May/Jun. 2021.

[4] M. Noor-A-Rahim et al., "6G for vehicle-to-everything (V2X) communications: Enabling technologies, challenges, and opportunities," *Proc. IEEE*, vol. 110, no. 6, pp. 712–734, Jun. 2022.

[5] Y. F. Payalan and M. A. Guvensan, "Towards next-generation vehicles featuring the vehicle intelligence," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 1, pp. 30–47, Jan. 2020.

[6] J. Liu and J. Liu, "Intelligent and connected vehicles: Current situation, future directions, and challenges," *IEEE Commun. Standards Mag.*, vol. 2, no. 3, pp. 59–65, Sep. 2018.

[7] S. Chandupatla, N. Yerram, and B. Achuthan, "Augmented reality projection for driver assistance in autonomous vehicles," SAE, Warrendale, PA, USA, Tech. Paper, 2020-01-1035, 2020.

[8] M. F. Pervej, L. T. Tan, and R. Q. Hu, "User preference learning aided collaborative edge caching for small cell networks," in *Proc. IEEE Glob. Commun. Conf.*, 2020, pp. 1–6.

[9] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, "A survey on low latency towards 5G: RAN, core network and caching solutions," *IEEE Commun. Surv. Tut.*, vol. 20, no. 4, pp. 3098–3130, Fourthquarter 2018.

[10] M. F. Pervej, L. T. Tan, and R. Q. Hu, "Artificial intelligence assisted collaborative edge caching in small cell networks," in *Proc. IEEE Glob. Commun. Conf.*, 2020, pp. 1–7.

[11] L. Zhao, H. Li, N. Lin, M. Lin, C. Fan, and J. Shi, "Intelligent content caching strategy in autonomous driving toward 6G," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 9786–9796, Jul. 2022.

[12] W. Qi, Q. Li, Q. Song, L. Guo, and A. Jamalipour, "Extensive edge intelligence for future vehicular networks in 6G," *IEEE Wireless Commun.*, vol. 28, no. 4, pp. 128–135, Aug. 2021.

[13] P. Wu, L. Ding, Y. Wang, L. Li, H. Zheng, and J. Zhang, "Performance analysis for uplink transmission in user-centric ultra-dense V2I networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 9342–9355, Sep. 2020.

[14] M. F. Pervej and S.-C. Lin, "Dynamic power allocation and virtual cell formation for Throughput-Optimal vehicular edge networks in highway transportation," in *Proc. IEEE Int. Conf. Commun. Workshops*, 2020, pp. 1–7.

[15] Y. Zhou and Y. Zhang, "User-centric data communication service strategy for 5G vehicular networks," *IET Commun.*, vol. 16, no. 10, pp. 1041–1056, Jul. 2022.

[16] Z. Cheng, D. Zhu, Y. Zhao, and C. Sun, "Flexible virtual cell design for ultradense networks: A machine learning approach," *IEEE Access*, vol. 9, pp. 91575–91583, 2021.

[17] H. Xiao, X. Zhang, A. T. Chronopoulos, Z. Zhang, H. Liu, and S. Ouyang, "Resource management for multi-user-centric V2X communication in dynamic virtual-cell-based ultra-dense networks," *IEEE Trans. Commun.*, vol. 68, no. 10, pp. 6346–6358, Oct. 2020.

[18] T. Sahin, M. Klugel, C. Zhou, and W. Kellerer, "Virtual cells for 5G V2X communications," *IEEE Commun. Standards Mag.*, vol. 2, no. 1, pp. 22–28, Mar. 2018.

[19] T. Şahin, M. Klügel, C. Zhou, and W. Kellerer, "Multi-user-centric virtual cell operation for V2X communications in 5G networks," in *Proc. IEEE Conf. Standards Commun. Netw.*, 2017, pp. 84–90.

[20] M. F. Pervej and S.-C. Lin, "Eco-Vehicular edge networks for connected transportation: A distributed multi-agent reinforcement learning approach," in *Proc. IEEE 92nd Veh. Technol. Conf.*, 2020, pp. 1–7.

[21] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[22] X. Huang, K. Xu, Q. Chen, and J. Zhang, "Delay-aware caching in Internet of Vehicles networks," *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10911–10921, Jul. 2021.

[23] Z. Nan, Y. Jia, Z. Ren, Z. Chen, and L. Liang, "Delay-aware content delivery with deep reinforcement learning in Internet of Vehicles," *IEEE Trans. Intel. Transport. Syst.*, vol. 23, no. 7, pp. 8918–8929, Jul. 2022.

[24] S. Fang, H. Chen, Z. Khan, and P. Fan, "On the content delivery efficiency of NOMA assisted vehicular communication networks with delay constraints," *IEEE Wireless Commun. Lett.*, vol. 9, no. 6, pp. 847–850, Jun. 2020.

[25] Y. Dai, D. Xu, K. Zhang, S. Maharjan, and Y. Zhang, "Deep reinforcement learning and permissioned blockchain for content caching in vehicular edge computing and networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4312–4324, Apr. 2020.

[26] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4298–4311, Apr. 2020.

[27] Z. Zhang, C.-H. Lung, M. St-Hilaire, and I. Lambadaris, "Smart proactive caching: Empower the video delivery for autonomous vehicles in ICN-based networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7955–7965, Jul. 2020.

[28] S. Fang, Z. Khan, and P. Fan, "A cooperative RSU caching policy for vehicular content delivery networks in two-way road with a T-junction," in *Proc. IEEE 91st Veh. Technol. Conf.*, 2020, pp. 1–5.

[29] W. Liu, H. Zhang, H. Ding, D. Li, and D. Yuan, "Mobility-aware coded edge caching in vehicular networks with dynamic content popularity," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2021, pp. 1–6.
[30] Y. Jiang, M. Ma, M. Bennis, F. Zheng, and X. You, "User preference learning-based edge caching for fog radio access network," *IEEE Trans. Commun.*, vol. 67, no. 2, pp. 1268–1283, Feb. 2019.
[31] A. Malik, J. Kim, K. S. Kim, and W.-Y. Shin, "A personalized preference learning framework for caching in mobile networks," *IEEE Trans. Mobile Comput.*, vol. 20, no. 6, pp. 2124–2139, Jun. 2021.
[32] Y. Lin, Z. Zhang, Y. Huang, J. Li, F. Shu, and L. Hanzo, "Heterogeneous user-centric cluster migration improves the connectivity-handover trade-off in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 16027–16043, Dec. 2020.
[33] S.-C. Lin, K.-C. Chen, and A. Karimoddini, "SDVEC: Software-defined vehicular edge computing with ultra-low latency," *IEEE Commun. Mag.*, vol. 59, no. 12, pp. 66–72, Dec. 2021.
[34] 3rd Generation Partnership Project "Technical specification group radio access network; V2X services based on NR; user equipment (UE) radio transmission and reception," 3GPP, Sophia Antipolis, France, Tech. Rep. 38.886 V0.5.0, Release 16, Feb. 2020.
[35] R. L. Graham, D. E. Knuth, O. Patashnik, and S. Liu, "Concrete mathematics: A foundation for computer science," *Comput. Phys.*, vol. 3, no. 5, pp. 106–107, 1989.
[36] 3rd Generation Partnership Project, "Technical specification group radio access network; study on channel model for frequencies from 0.5 to 100 GHz," 3GPP, Sophia Antipolis, France, Tech. Rep. 38.901 V16.1.0, Release 16, Dec. 2019.
[37] S. O. Somuyiwa, A. György, and D. Gündüz, "A reinforcement-learning approach to proactive caching in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1331–1344, Jun. 2018.
[38] B. N. Bharath, K. G. Nagananda, D. Gündüz, and H. V. Poor, "Caching with time-varying popularity profiles: A learning-theoretic perspective," *IEEE Trans. Commun.*, vol. 66, no. 9, pp. 3837–3847, Sep. 2018.
[39] Y. H. Wang, "On the number of successes in independent trials," *Statistica Sinica*, vol. 3, pp. 295–312, 1993.
[40] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[41] C. J. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3/4, pp. 279–292, 1992.
[42] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[43] T. Guo and A. Suárez, "Enabling 5G ran slicing with EDF slice scheduling," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2865–2877, Mar. 2019.
[44] P. Guan and X. Deng, "Maximize potential reserved task scheduling for URLLC transmission and edge computing," in *Proc. IEEE 92nd Veh. Technol. Conf.*, 2020, pp. 1–5.
[45] G. C. Buttazzo, *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*, vol. 24. Berlin, Germany: Springer Science & Business Media, 2011.
[46] H. W. Kuhn, "The Hungarian method for the assignment problem," *Nav. Res. Logistics Quart.*, vol. 2, no. 1/2, pp. 83–97, 1955.
[47] D. B. West et al. *Introduction to Graph Theory*, vol. 2. Upper Saddle River, NJ, USA: Prentice-Hall, 2001.
[48] P. A. Lopez et al., "Microscopic traffic simulation using sumo," in *Proc. 21st Int. Conf. Intell. Transp. Syst.*, 2018, pp. 2575–2582.
[49] R. R. Roy, *Handbook of Mobile Ad hoc Networks for Mobility Models*, vol. 170. Berlin, Germany: Springer, 2011.
[50] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrück, S. Fischer, and J.-P. Hubaux, "TraCI: An interface for coupling road traffic and network simulators," in *Proc. 11th Commun. Netw. Simul. Symp.*, 2008, pp. 155–163.
[51] Y. Zhang, R. Wang, M. S. Hossain, M. F. Alhamid, and M. Guizani, "Heterogeneous information network-based content caching in the Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 68, no. 10, pp. 10216–10226, Oct. 2019.
[52] S. Podlipnig and L. Böszörmenyi, "A survey of web cache replacement strategies," *ACM Comput. Surv.*, vol. 35, no. 4, pp. 374–398, Dec. 2003.
[53] H. Kellerer, U. Pferschy, and D. Pisinger, "Multidimensional knapsack problems," in *Knapsack Problems*. Berlin, Germany: Springer, 2004, pp. 235–283.

Md Ferdous Pervej (Graduate Student Member, IEEE) received the B.Sc. degree in electronics and telecommunication engineering from the Rajshahi University of Engineering and Technology, Rajshahi, Bangladesh, in 2014, and the M.S. degree in electrical engineering from Utah State University, Logan, UT, USA, in 2019. In 2023, he completed the requirements for his Ph.D. degree in electrical engineering with North Carolina State University, Raleigh, NC, USA.

In summer 2021, he was with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA. From May to December 2022, he was with Futurewei Wireless Research and Standards, Schaumburg, IL, USA. His research interests include wireless edge networks, vehicle-to-everything communication, edge caching/computing, and wireless networks for distributed and on-device ML.

Richeng Jin (Member, IEEE) received the B.S. degree in information and communication engineering from Zhejiang University, Hangzhou, China, in 2015, and the Ph.D. degree in electrical engineering from North Carolina State University, Raleigh, NC, USA, in 2020.

From 2021 to 2022, he was a Postdoctoral Researcher in electrical and computer engineering with North Carolina State University. He is currently a Faculty Member of the department of information and communication engineering with Zhejiang University. His research interests include the general area of wireless AI, game theory, and security and privacy in machine learning/artificial intelligence and wireless networks.

Shih-Chun Lin (Member, IEEE) received the Ph.D. degree from the Georgia Institute of Technology in 2017. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC, USA, leading the Intelligent Wireless Networking Laboratory. His research interests include 6G networks, software-defined infrastructure, machine learning techniques, mathematical optimization, and performance evaluation.

Huaiyu Dai (Fellow, IEEE) received the B.E. and M.S. degrees in electrical engineering from Tsinghua University, Beijing, China, in 1996 and 1998, respectively, and the Ph.D. degree in electrical engineering from Princeton University, Princeton, NJ, USA, in 2002.

He was with Bell Labs, Lucent Technologies, Holmdel, NJ, in summer 2000, and with AT&T Labs-Research, Middletown, NJ, in summer 2001. He is currently a Professor of Electrical and Computer Engineering with NC State University, Raleigh, NC, USA, holding the title of University Faculty Scholar. His research interests include the general areas of communications, signal processing, networking, and computing. His current research interests include machine learning and artificial intelligence for communications and networking, multilayer and interdependent networks, dynamic spectrum access and sharing, as well as security and privacy issues in the above systems.

He is the Area Editor of IEEE TRANSACTIONS ON COMMUNICATIONS, a Member of the Executive Editorial Committee for IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, and the Editor of IEEE TRANSACTIONS ON SIGNAL PROCESSING. he is currently the Area Editor of IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He was the co-recipient of the best paper awards at 2010 IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2010), 2016 IEEE INFOCOM BIGSECURITY Workshop, and 2017 IEEE International Conference on Communications (ICC 2017). He was also the recipient of Qualcomm Faculty Award in 2019.