Hierarchical Deep Reinforcement Learning based Dynamic RAN Slicing for 5G V2X

Umuralp Kaytaz*, Fikret Sivrikaya* and Sahin Albayrak*†
GT-ARC gGmbH, Berlin, Germany*
DAI-Labor, Technical University of Berlin, Berlin, Germany †
{umuralp.kaytaz@gt-arc.com, fikret.sivrikaya@gt-arc.com, sahin.albayrak@tu-berlin.de}

***Abstract*—Radio Access Network (RAN) slicing is getting increasing attention as a resource allocation technique for satisfying diverse Quality-of-Service (QoS) requirements in 5G vehicular networks. Hierarchical Reinforcement Learning (HRL), such as hierarchical-DQN (h-DQN), is a promising slice management approach that decomposes performance constraints into a subroutine hierarchy and uses Deep Reinforcement Learning (DRL) at different temporal scales for online-learning an optimal policy of bandwidth allocation. In this paper, we tackle RAN slicing problem in 5G vehicle-to-everything (V2X) communications and present h-DQN based Soft Slicing (HSS) method for model-free opportunistic slice management. HSS consists of a multi-controller learning framework where a high-level meta-controller takes state input for determining a subgoal and a low-level controller decides on the action based on the given subgoal and the state. We compare performance of HSS with model-free and model-based Reinforcement Learning (RL) methods in terms of Age of Information (AoI), service delay and network throughput. Our results show that proposed scheme improves sample-efficiency and outperforms traditional and RL-based V2X RAN slice management methods in terms of network utility maximization.**

***Index Terms*—5G, Quality-of-Service (QoS), RAN Slicing, Reinforcement Learning, Intelligent Transportation System (ITS).**

I. Introduction

Intelligent Transportation Systems (ITSs) are envisioned to be an integral part of 5G communications enabling advanced applications while providing social as well as economic benefits [1]. Diverse Quality-of-Service (QoS) requirements of vehicular connectivity motivated various standardization efforts such as Cellular Vehicular-to-Everything (C-V2X) by the *3rd Generation Partnership Project* (3GPP) and Dedicated Short Range Communications (DSRC) by the *Institute of Electrical and Electronics Engineering* (IEEE) and *European Telecommunications Standards Institute* (ETSI) [2]. Although V2X communications and related 5G use cases have been considered since Long Term Evaluation (LTE) release 14 by the 3GPP, enabling mobile broadband with vertical applications in future vehicular domain still remains a challenge [3].

Due to dynamic pace of the vehicular communication networks, 5G ITS applications require in-network content storage and ultra-reliable V2X communications [4]. Therefore, storing replica of content with the help of Roadside Units (RSUs) and facilitating V2X communications is a promising solution for improving efficiency and connectivity in cache-enabled vehicular networks [5]. Furthermore, diversity of V2X application categories causes multi-dimensional QoS demands [6]. While infotainment applications, such as peer-to-peer file transfer or gaming, rely on enhanced Mobile Broad Band (eMBB) with high bandwidth allocation, an autonomous driving service such as platooning focuses on cooperative adaptive cruise control and demands Ultra-Reliable and Low-Latency Communications (URLLC) [7]. Radio Access Network (RAN) slicing is a virtualization technique for creating multiple logical networks on a shared physical infrastructure and a key enabling technology for simultaneously accommodating these diverse set of applications in 5G networks [1]. Although, there have been previous efforts on vehicular network slicing using network state information, online-learning for multi-dimensional QoS provisioning remains an open issue.

Cellular network slicing approaches are not practical solutions for the management of heterogeneous vehicular resources due to the multiplicity of use cases and connectivity requirements [8]. A RAN slicing architecture for vehicular resource management based on High Altitude Platforms (HAPs) is presented in [9]. Another work based on RSU virtualization is proposed in [6] for hierarchical opportunistic slicing, namely soft slicing, in evenly distributed traffic density. [10] addresses latency-aware dynamic resource allocation by formulating RAN slicing as a maximum utility optimisation problem and uses hierarchical decomposition technique for reducing the complexity of the optimisation problem. These approaches require complete network state information for computation of a slice management policy and do not allow an online-learning approach. The method proposed in [11] uses application aware content placement framework for determining vehicular cache content. Considering the dynamics of the ever-changing vehicular environment proposed heuristic design is not practical for satisfying diverse QoS demands in 5G networks. [12] examines RAN slicing by constructing optimization subroutines with constrained Reinforcement Learning (RL) algorithms. Another work in [13] proposes a proactive algorithm for Radio Resource Management (RRM) using Long Short-Term Memory (LSTM) and addresses partial network state observability with Deep Reinforcement Learning (DRL). [7] uses RL in a two-slice system consisting of eMBB and V2X slices for RAN slicing in heterogeneous networks. These

works use traditional RL-based methods that require training of unique policies for different tasks, therefore, they are not feasible solutions for complex real-world problems with multiple subgoals. On the other hand, Hierarchical Reinforcement Learning (HRL) accomplishes behaviour planing in a modular fashion by pursuing multiple subgoal policies that are reused for subsequent tasks. This allows sample-efficient computation and online-learning a policy using spatio-temporal abstractions [14].

In this paper, we propose HRL based dynamic RAN slicing for satisfying diverse demands in 5G vehicular communication networks. First, we develop a HRL formulation as a modular architecture using subgoal-based task decomposition for on-road content services. Secondly, we propose h-DQN based Soft Slicing (HSS) method as a novel multi-controller learning framework for dynamic RAN slicing in 5G V2X communications. Third, we integrate hierarchically prioritized experience replay [14] and a hybrid reward function strategy [6] to the proposed HSS algorithm for evaluating diverse QoS requirements. Fourth, we investigate bandwidth utilization and slice management performance of HSS using Age of Information (AoI), network utility and service delay metrics.

The remainder of the paper is organized as follows. Section II describes the system model. Section III presents the HRL formulation and multi-controller learning framework. Details of the HSS-based dynamic RAN slicing algorithm is provided in Section IV. Section V details simulation setup and analyzes experimental outcomes. Finally, we present our concluding remarks in Section VI.

II. System Model

Application-specific RAN slice management and orchestration framework in Fig. 1 is considered for the 5G V2X networking scenario. Automation in the slicing architecture is managed by the Service Orchestrator (SO), which receives requirements of vertical applications through a Service Level Agreement (SLA). SO creates slice and/or service instances and enforces slicing decisions through Network Functions Virtualization (NFV) Management platform by broking Virtual Machines (VMs) at the network edge. NFV Management & orchestration platform coordinates network resources and lifecycle management of Virtual Network Functions (VNF) while also taking responsibility for operating the network content of on-road applications. RSUs store the published content and distribute content to vehicles using infrastructure-to-vehicle (I2V) links. On the receiving end, vehicles store content for requested services and share their cache content using V2V communications for reducing workload of RSUs.

We employ four distinct RAN slices for possible V2X application categories, namely autonomous driving, intelligent assistance, infotainment, remote diagnostics and management [3]. For autonomous driving slice we consider platooning and remote driving applications that require URLLC service configurations. Similarly, intelligent assistance is a time-sensitive service aiming to increase driving comfort providing information related to parking lots, safety and traffic efficiency [3]. Contrarily, infotainment and remote diagnostics applications demand high-bandwidth consumption and local content availability rather than low-latency communications.

Each RAN slice contains multiple functions realized over RSU virtualization and local content sharing. RSU content update function is used for renewal of cached content at the infrastructure. Infrastructure-to-Vehicle (I2V) broadcast function enables proactive content delivery to multiple vehicles. On the other hand, I2V unicast function is for delivering requested content to a single node. Local breakout of requested content is realized through Vehicle-to-Vehicle (V2V) breakout function. Each function of RSU virtualisation operates through a Virtual Machine (VM) for the dedicated slice.

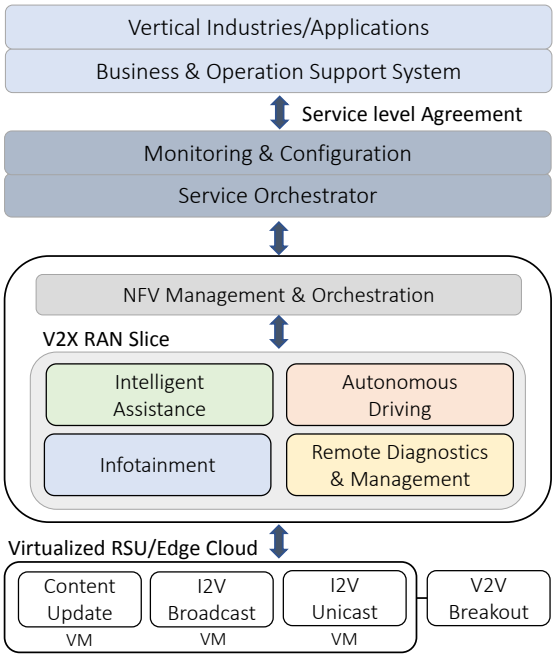

Figure 1: RAN slice management & orchestration framework for 5G V2X functionalities

III. Preliminaries

In this section, we provide preliminary information related to the HRL based dynamic RAN slicing approach. Initially, we present Markov Decision Process (MDP) formulation with respect to the dynamic RAN slicing problem. Afterwards, we elaborate on RL definitions, such as *value functions*, for representing the expected utility. We discuss Deep Q-Learning (DQL) for approximation of an optimal policy using Deep Neural Networks (DNNs). Furthermore, we explain DQL using Double Deep Q-Network (DDQN) approach for parameter optimization and loss function minimization. Finally, we introduce HRL as a multi-controller DQL framework based on temporal abstraction.

A. Markov Decision Process & Value Functions

MDP is a formulation technique for discrete-time stochastic control problems. MDP formulation uses one-step dynamics of the environmental information alongside with agent's state and action definitions [15]. Our state-space $\mathcal{S}$ represents possible slice bandwidth allocation ratios among edge VMs. Action space $\mathcal{A}$ is the set of actions where each action is a possible rescheduling of the available bandwidth. Environmental dynamics of the MDP consist of a discount factor γ for contribution of possible future rewards, a state-transition probability distribution $\mathcal{P} : \mathcal{S}\text{x}\mathcal{A}\text{x}\mathcal{S} \rightarrow [0, 1]$ for representing probability of upcoming bandwidth allocation ratios and a reward function $\mathcal{R} : \mathcal{S}\text{x}\mathcal{A}\text{x}\mathcal{S} \rightarrow \mathbb{R}$ for mapping immediate reward based on the given state, upcoming state and given action. We can define MDP as a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$ where $\mathcal{P}$ and $\mathcal{R}$ compose the agent's environmental model.

RL is a commonly used way of solving MDP formulations that require finding a policy π for maximizing expected future rewards [15]. In RL, *value-function* $V^{\pi}(s)$ estimates utility and/or expected discounted future rewards of state s while the agent behaves under a policy π. Similarly, *action-value function* (or Q-value) $Q^{\pi}(s, a)$ represents the expected return of taking an action a at state s while following a policy π for the agent objective. Value functions, $V^{\pi}(s)$ and $Q^{\pi}(s, a)$, at time t can be calculated with the following equations where γ^k is used for decreasing importance of future rewards and $r_{t+1} = \mathcal{R}(s_t, a_t, s_{t+1})$ is the immediate reward

$$V^{\pi}(s) = \mathbb{E}^{\pi}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s \right] \tag{1}$$

$$Q^{\pi}(s, a) = \mathbb{E}^{\pi}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s, a_t = a \right] \tag{2}$$

Optimal *state-value* $V^*(s)$ and *action-value* $Q^*(s, a)$ functions represent the utility under a policy π^* that maximizes the expected rewards

$$V^*(s) = \max_{\pi} V^{\pi}(s) \tag{3}$$

$$Q^*(s, a) = \max_{\pi} Q^{\pi}(s, a) \tag{4}$$

B. Deep Q-Learning & Double Deep Q-Network

In the absence of environmental model definitions, solving an MDP formulation requires approximation of a deterministic stationary policy. As an online-learning approach, Q-learning recursively learns optimal policy π^* by iteratively approximating optimal *action-value* function $Q^*(s, a)$ that stores highest utility values. Considering high-dimensional problems where *action-value* functions are approximated using DNNs, Q-value can have a parametrized representation $Q(s, a|\theta)$. DQL process using DNN approximation can be expressed as follows

Table I: Hierarchical Reinforcement Learning Notation

Notation	Description
γ	Discount factor
π	Policy
π^*	Optimal policy
π^*_g	Optimal policy over subgoals
$\pi^*_{a,g}$	Optimal policy over actions for subgoal g
r_t	Reward observed at time t
s_t	State at time t
a_t	Action taken at time t
g'	Forthcoming subgoal
$V^{\pi}(s)$	*State-value* function under policy π
$Q^{\pi}(s, a)$	*Action-value* function under policy π
$V^*(s)$	Optimal *state-value* function
$Q^*(s, a)$	Optimal *action-value* function
$Q(s, a\|\theta)$	Parametrized *action-value* function
$Q(s, a\|\theta, g)$	Parametrized *action-value* function given subgoal g
$\mathcal{L}(\theta)$	Loss function for parameter set θ

$$Q^*(s_t, a_t) = \mathbb{E}[r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}|\theta_t)] \tag{5}$$

DDQN is a widely used parameter optimization method for loss function $\mathcal{L}(\theta)$ minimization and weight update in Q-learning procedure. DDQN uses two DNNs where at each iteration a *target-value* $Y_t = Q^*(s_t, a_t)$ and a *prediction-value* $Q_t = Q(s_t, a_t|\theta_t)$ are calculated with separate DQL processes. Using *Huber loss* as the loss function $\mathcal{L}(\theta)$ with a constant c and *learning-rate* α, DDQN weight update procedure can be expressed as

$$\mathcal{L}(\theta_t) = \begin{cases} \frac{1}{2}(Y_t - Q_t)^2 & |Y_t - Q_t| \leq c \\ c|Y_t - Q_t| - \frac{1}{2}c^2 & o/w \end{cases} \tag{6}$$

$$\theta_{t+1} = \theta_t + \alpha \frac{\delta \mathcal{L}(\theta)}{\delta \theta} \tag{7}$$

C. Hierarchical Deep Reinforcement Learning

HRL uses DQL in a two-stage hierarchy incorporating a *meta-controller* and a *controller*. High-level *meta-controller* chooses a subgoal $g_t \in \mathcal{G}$ given state s_t and estimates optimal *action-value function* $Q^*_1(s_t, g_t)$ for an optimal policy π^*_g over subgoals. On the other hand, low-level *controller* takes in state s_t and subgoal g_t and estimates optimal *action-value function* $Q^*_2(s_t, a_t|g_t)$ for an optimal policy $\pi^*_{a,g}$ over actions [14]. Using neural networks as non-linear function approximators, *value-functions* can be represented as $Q^*_1(s_t, g_t) = Q_1(s_t, g_t|\theta_1)$ and $Q^*_2(s_t, a_t|g_t) = Q_2(s_t, a_t|\theta_2, g_t)$. Resulting hierarchical learning process with temporal abstraction can be expressed with the following Q-value functions

$$Q_1(s_t, g_t|\theta_1) = \mathbb{E}\left[\sum_{t'=t+1}^{t+1+N} r_{t'} + \gamma \max_{g'} Q_1(s_{t+1+N}, g'|\theta_1) \right] \tag{8}$$

$$Q_2(s_t, a_t|\theta_2, g_t) = \mathbb{E}[r_{t+1} + \gamma \max_{a_{t+1}} Q_2(s_{t+1}, a_{t+1}|\theta_2, g_t)] \tag{9}$$

where g' is the forthcoming subgoal that is activated at s_{t+1+N} and N is the number of time steps that takes *controller* to achieve current subgoal g.

IV. Approach

In this section, we describe the HSS based opportunistic RAN slicing architecture. Initially, we present a multi-controller learning structure for modular behaviour planning and RAN slicing in vehicular environment. Later on, we explain the HSS algorithm based on hierarchically prioritized experience replay and hybrid reward function techniques. Followingly, we elaborate on a hybrid reward mechanism for network utility and QoS performance analysis of the HSS method during dynamic slice allocation.

A. Multi-Controller Learning for RAN Slicing

Proposed architecture consists of an HRL agent that uses h-DQN algorithm for coordinating separate DQL policy controllers, namely *meta-controller* and *controller*, as shown in Fig. 2. Each controller uses a DDQN for the policy approximation process. High-level *meta-controller* consists of an *Option-Value Network* (OVN) that computes a policy over *options*. *Option* set assigned to the *meta-controller* represents possible subgoals/*options* of slice allocation, where each subgoal is a V2X RAN slice of the NFV service. *Meta-controller* periodically decides on a RAN slicing subgoal based on the observed environmental state. On the other hand, an *Action-Value Network* (AVN) operated at the low-level *controller* computes a policy over *actions*. Each *action* represents an inter-slice or intra-slice bandwidth allocation decision for opportunistically managing resources among RSUs and/or edge cloud VMs that are responsible for V2X functions. *Controller* uses given subgoal alongside with environmental state and response for iteratively computing a spectrum utilization strategy and selecting an *action*. Using environmental response for the *controller* actions, agent calculates reward signals based on the QoS performance.

B. H-DQN based Soft Slicing (HSS) Algorithm

Algorithm 1 is executed for learning a policy of optimal bandwidth allocation between edge VMs during network slice orchestration. At the beginning, possible V2X RAN slice *options* are taken as the input by the multi-controller process. Agent initializes DDQNs for *option* and *action* policy calculation (Line 1). Training procedure is iteratively carried out for the predetermined number of episodes (Lines 2-24). At each episode, DQNs iterate over state observations and optimize parameters using stochastic gradient descent at different temporal scales (Lines 3-23). *Meta-controller* uses OVN for selecting the utility maximizing slice type at every subgoal selection period (Lines 4-8). However, with ϵ-greedy policy, a random *option* is selected with the exploration probability (ϵ_1) for higher expected utility detection (Line 8). AVN receives subgoal and calculates the bandwidth allocation *action* for expected utility maximization (Line 9). Similarly, an ϵ-greedy procedure is followed by the

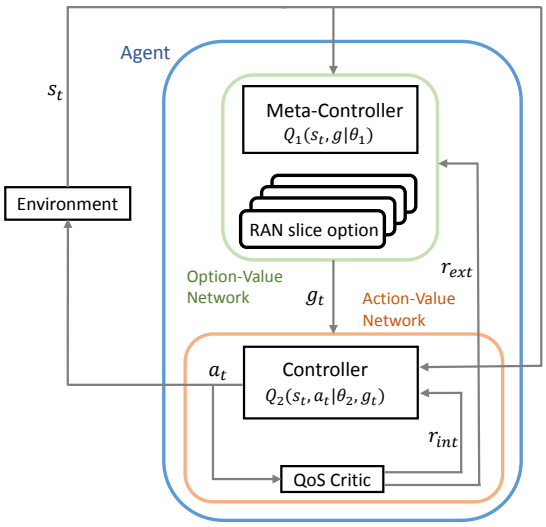

Figure 2: Multi-controller hierarchical RL agent architecture for option-based dynamic RAN slicing

controller (Line 10). Agent calculates received reward signals for the controllers using environmental response (Lines 11-12). Upon receiving the upcoming state, at every time-step, a controller *experience* e_A is stored in the *controller* replay memory $\mathcal{D}_2$ (Lines 13-14). In case of sufficient *experiences*, greater than a predetermined threshold Th_A^{sample}, controller samples *experiences* from the replay memory $\mathcal{D}_2$ for training on decorrelated samples (Lines 15-16). Followingly, *meta-controller* calculates cumulative extrinsic reward for the current subgoal and environmental state gets updated (Lines 17-18). *Experience* replay procedure is also executed by the *meta-controller* in case of subgoal termination. *Meta-controller* stores and samples *experiences* at a different temporal scale based on the subgoal completion frequency (Lines 19-22). At the end of each episode, trained weights of agent prediction networks $\{\mathcal{Q}_O^{pred}, \mathcal{Q}_A^{pred}\}$ are transferred to $\{\mathcal{Q}_O^{trgt}, \mathcal{Q}_A^{trgt}\}$ for target network updates (Lines 23-24).

C. Hybrid Reward Mechanism for HSS

In order to satisfy diverse QoS requirements of 5G applications and assess the performance of controller resource allocation strategies, we develop a hybrid reward mechanism using AoI, service delay and throughput metrics. Reward function of the policy controller for a given subgoal and/or slice i is calculated with the following

$$\mathcal{R}^i(s_t, a_t, s_{t+1}) = \sum_{f=1}^{F} (\beta_1^i AoI_f^i + \beta_2^i \frac{1}{\bar{\mu_f}^i - \bar{\lambda_f}^i}) + \beta_3^i Th_t^i \tag{10}$$

where β_j^i represents weight of the metric j at slice i, AoI_f^i is the time elapsed since generation of the slice i content f, Th_t^i is the slice i throughput at time t and $\frac{1}{\bar{\mu_f}^i - \bar{\lambda_f}^i}$ is the service delay where $\bar{\mu_f}^i$ is the average service rate to the vehicles and $\bar{\lambda_f}^i$ is the average arrival rate of content file

Algorithm 1: h-DQN based Soft Slicing (HSS)

Input: $\mathcal{G} \leftarrow$ set of *subgoals(V2X RAN slices)*
Data: $\{\mathcal{D}_1, \mathcal{D}_2\}$: experience replay memories,
$\{\theta_1, \theta_2\}$: controller parameters

Initialize prediction and target networks for *option* and *action* calculation $\{\ \mathcal{Q}_O^{pred}, \mathcal{Q}_O^{trgt}, \mathcal{Q}_A^{pred}, \mathcal{Q}_A^{trgt}\}$
for *each episode* $\mathcal{E}_i \mid i$ = {*1,...,num_episodes*} **do**
  for *each state* $s_t \in \mathcal{E}_i$ **do**
    if $\{t == kN \mid k = \{0, ..., K\}\}$ **then**
      Initialize $\mathrm{R}_{ext} \leftarrow 0$
      Initialize $\mathrm{s}_{init} \leftarrow s_t$
      Compute $g = \mathcal{Q}_O^{pred}.get\_subgoal(s_t)$
      $g \leftarrow random(\epsilon_1)$
    Compute $a_t = \mathcal{Q}_A^{pred}.get\_action(s_t, g)$
    $a_t \leftarrow random(\epsilon_{2,g})$
    Compute $r_{ext} = \mathcal{Q}_O^{pred}.get\_reward(s_t, a_t, g)$
    Compute $r_{int} = \mathcal{Q}_A^{pred}.get\_reward(s_t, a_t, g)$
    Get next state s_{t+1}
    Store $\mathcal{D}_2 \leftarrow e_A = \langle s_t, a_t, r_{int}, s_{t+1}, g \rangle$
    if $|\mathcal{D}_2| > Th_A^{sample}$ **then**
      Compute $\mathcal{Q}_A^{pred}.updateWithReplay(\mathcal{L}_2(\theta_2, i), \mathcal{D}_2)$
    Update $\mathrm{R}_{ext} += r_{ext}$
    Update $s_t \leftarrow s_{t+1}$
    if *g is terminated* **then**
      Store $\mathcal{D}_1 \leftarrow e_O = \langle s_{init}, \mathcal{R}_{ext}, s_{t+1}, g \rangle$
      if $|\mathcal{D}_1| > Th_O^{sample}$ **then**
        Compute $\mathcal{Q}_O^{pred}.updateWithReplay(\mathcal{L}_1(\theta_1, i), \mathcal{D}_1)$
  Update $\mathcal{Q}_O^{trgt} \xleftarrow{weights} \mathcal{Q}_O^{pred}$
  Update $\mathcal{Q}_A^{trgt} \xleftarrow{weights} \mathcal{Q}_A^{pred}$

f to the RSU cache. Overall, reward value for OVN can be calculated as the average of reward signals from all RAN slice *options* using

$$\mathcal{R}_t^{OVN} = \frac{1}{N_{slice}} \sum_{i=1}^{N_{slice}} \mathcal{R}^i(s_t, a_t, s_{t+1}) \tag{11}$$

V. Simulations

A. Simulation Setup

Simulation results of spectrum decision methods have been obtained using numerical computation library TensorFlow, open-source neural-network library Keras and numeric computing environment MATLAB. HSS uses two-level policy controller hierarchy consisting of different Feed-Forward Network (FFN) architectures with varying hidden layer sizes, as presented in Table II. Similarly, non-hierarchical actor-critic DRL architecture, a DDQN, has been implemented for performance comparison. As a traditional bandwidth utilization approach, slotted-Aloha protocol have been used. On the other hand, myopic bandwidth allocation based Whittle index policy was preferred as a model-based RL approach for optimal policy computation [16].

Table II: Hyperparameters for DRL

Hyperparameter	Value
Meta-Controller FFN	20-12-5
Controller FFN	20-15-15-12-12-10-8-5
Non-hierarchical DRL FFN	20-15-15-10
Number of episodes	100
Batch size B	10
Memory size	2000
Learning rate α	0.001
Discount factor γ	0.95
Exploration rate ϵ	0.7 $\rightarrow$ 0.01
Epsilon decay	0.997
Loss function	Huber
Optimizer	Adam

For the road traffic simulation, a one-dimensional road consisting of unidirectional and/or bidirectional lanes have been considered similar to the work in [6]. Distribution of cache-enabled vehicles has been designed as a Poisson Point Process (PPP) whereas RSU has been modeled as a queuing system with a shared processor and limited cache availability. Available bandwidth of the RSU is uniformly distributed among VMs that are responsible for serving each slice instance at the network edge. Each file f from slice i has been randomly assigned a request probability q_f^i, where $\sum_{i=1}^{N_{slice}} \sum_{f=1}^{F} q_f^i = 1$ and a PPP was used for modeling vehicle requests.

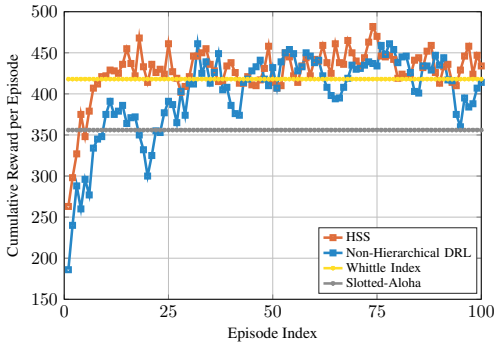

Figure 3: HSS versus other slice management methods

B. Simulation Results

We first present the simulation results obtained under various slice management algorithms. Fig. 3 displays *cumulative reward per episode* obtained by a single slice allocation agent operating under HSS, non-hierarchical DQL, Whittle index and slotted-aloha algorithms. We observe that HSS provides sample-efficient convergence compared to non-hierarchical DQL approach due to its modular behaviour

planning architecture that optimizes pursuing subsequent tasks. Furthermore, better stability of bandwidth utilization decisions are observed. Therefore, higher utility values are perserved in consequent episodes. On the other hand, both model-agnostic online-learning methods, actor-critic DQL and HSS, outperform model-based RL, namely Whittle index policy, and slotted-aloha protocol based slicing decisions.

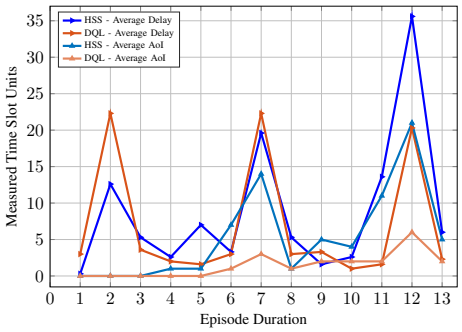

Figure 4: Delay and AoI performance of DRL-based methods

Fig. 4 shows the comparison of model-free DRL approaches, namely HSS and non-hierarchical actor-critic DQL, in terms of average delay and AoI performance throughout the continuing time-slot iterations in an episode. As the number of time slots and transmissions increases, average delay of traditional DQL-based approach oscillates with a stable period while delay obtained from HSS increases. Similarly, AoI observed from transmitted packets under HSS regime tends to increase with the episode duration. Contrarily, non-hierarchical policy controller reaches stabilized AoI statistics at data transmissions. Multiple controllers utilized in HSS method operate at different temporal scales. While the *meta-controller* that is responsible for slice and subgoal selection has a less frequent decision making strategy, non-hierarchical DQL operates with a flat architecture that is capable of slot-based decision making. Therefore, transmission decisions under HSS result in larger delays and increasing AoI observations.

VI. Conclusion and Future Work

In this paper, we present hierarchical deep reinforcement learning based dynamic RAN slicing policy control as a model-agnostic online-learning method for 5G V2X communication systems. Initially, h-DQN based slicing approach HSS has been proposed based on the hybrid reward mechanism for satisfying diverse QoS demands. Later on, we compared performance of HSS with slotted-aloha and traditional model-free and model-based RL algorithms. We observe that pursuing task-decomposition based subgoal policies using hierarchical HSS architecture resulted in better utilization of bandwidth resources with increased stability and sample-efficiency. On the other hand, non-hierarchical DQL had a more stable delay and AoI performance due to its time-slot based decision making approach. In our future work we aim to focus on stabilizing delay performance of HSS using multi-agent cooperative learning techniques.

VII. Acknowledgment

This work is a part of the 5G-MOBIX project funded by the European Union's Horizon 2020 research and innovation programme under grant agreement number 825496.

References

[1] H. Khan, P. Luoto, M. Bennis, and M. Latva-aho, "On the application of network slicing for 5g-v2x," in *European Wireless 2018; 24th European Wireless Conference*. VDE, 2018, pp. 1–6.

[2] S. Zeadally, M. A. Javed, and E. B. Hamida, "Vehicular communications for its: standardization and challenges," *IEEE Communications Standards Magazine*, vol. 4, no. 1, pp. 11–17, 2020.

[3] C. Campolo, A. Molinaro, A. Iera, and F. Menichella, "5g network slicing for vehicle-to-everything services," *IEEE Wireless Communications*, vol. 24, no. 6, pp. 38–45, 2017.

[4] Z. MacHardy, A. Khan, K. Obana, and S. Iwashina, "V2x access technologies: Regulation, research, and remaining challenges," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 1858–1877, 2018.

[5] I. U. Din, B.-S. Kim, S. Hassan, M. Guizani, M. Atiquzzaman, and J. J. Rodrigues, "Information-centric network-based vehicular communications: Overview and research opportunities," *Sensors*, vol. 18, no. 11, p. 3957, 2018.

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

[7] H. D. R. Albonda and J. Pérez Romero, "Analysis of ran slicing for cellular v2x and mobile broadband services based on reinforcement learning," *EAI Endorsed Transactions on Wireless Spectrum*, vol. 4, no. 13, pp. 1–11, 2020.

[8] C. Campolo, A. Molinaro, A. Iera, R. R. Fontes, and C. E. Rothenberg, "Towards 5g network slicing for the v2x ecosystem," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*. IEEE, 2018, pp. 400–405.

[9] S. Zhang, W. Quan, J. Li, W. Shi, P. Yang, and X. Shen, "Air-ground integrated vehicular network slicing with content pushing and caching," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 9, pp. 2114–2127, 2018.

[10] S. O. Oladejo and O. E. Falowo, "Latency-aware dynamic resource allocation scheme for multi-tier 5g network: A network slicing-multitenancy scenario," *IEEE Access*, vol. 8, pp. 74 834–74 852, 2020.

[11] D. D. Van, Q. Ai, Q. Liu, and D.-T. Huynh, "Efficient caching strategy in content-centric networking for vehicular ad-hoc network applications," *IET Intelligent Transport Systems*, vol. 12, no. 7, pp. 703–711, 2018.

[12] W. Wu, N. Chen, C. Zhou, M. Li, X. Shen, W. Zhuang, and X. Li, "Dynamic ran slicing for service-oriented vehicular networks via constrained learning," *IEEE Journal on Selected Areas in Communications*, 2020.

[13] X. Chen, C. Wu, T. Chen, H. Zhang, Z. Liu, Y. Zhang, and M. Bennis, "Age of information aware radio resource management in vehicular networks: A proactive deep reinforcement learning perspective," *IEEE Transactions on Wireless Communications*, vol. 19, no. 4, pp. 2268–2281, 2020.

[14] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum, "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation," *Advances in neural information processing systems*, vol. 29, pp. 3675–3683, 2016.

[15] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[16] K. Liu and Q. Zhao, "Indexability of restless bandit problems and optimality of whittle index for dynamic multichannel access," *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5547–5567, Nov 2010.