Tripathi, S., Puligheddu, C. & Chiasserini, C. F. (9-11 March 2021). *An RL Approach to Radio Resource Management in Heterogeneous Virtual RANs* [proceedings]. 2021 16th Annual Conference on Wireless On-demand Network Systems and Services Conference (WONS), Klosters, Switzerland.

DOI: 10.23919/WONS51326.2021.9415591

An RL Approach to Radio Resource Management in Heterogeneous Virtual RANs

Sharda Tripathi
Politecnico di Torino
Turin, Italy
sharda2309@gmail.com

Corrado Puligheddu
Politecnico di Torino
Turin, Italy
corrado.puligheddu@polito.it

Carla Fabiana Chiasserini
Politecnico di Torino
Turin, Italy
carla.chiasserini@polito.it

***Abstract*—5G networks are primarily designed to support a wide range of services characterized by diverse key performance indicators (KPIs). A fundamental component of 5G networks, and a pivotal factor to the fulfillment of the services KPIs, is the virtual radio access network (RAN), which allows high flexibility on the control of the radio link. However, to fully exploit the potentiality of virtual RANs in non-stationary environments, an efficient mapping of the rapidly varying context to radio control decisions is not only essential, but also challenging owing to the non-trivial interdependence of network and channel conditions. In this paper, we propose CAREM, an RL framework for dynamic radio resource allocation, which selects the best link and modulation and coding scheme (MCS) for packet transmission, so as to meet the KPI requirements in heterogeneous virtual RANs. To show its effectiveness in real-world conditions, we provide a proof-of-concept through actual testbed implementation. Experimental results demonstrate that CAREM enables an efficient radio resource allocation, for any of the considered time periodicity of the decision-making process.**

***Index Terms*—5G technology, reinforcement learning, radio access network virtualization, radio resource allocation, heterogeneous networks.**

I. Introduction

The envisaged paradigm of 5G mobile technologies is aimed to serve a broad spectrum of applications having diverse requirements on various key performance indicators (KPIs), ranging from high reliability and low latency to large-scale connectivity and massive data rates [1]. To accommodate such ambitious vision of 5G, new generation wireless access networks are required not only to integrate various flexible multi-access technologies such as mmWave and massive MIMO, but also to provide a versatile radio resource management (RRM) system that can ensure efficient spectrum utilization and seamless interoperability [2].

A powerful concept addressing such needs is the virtualization of the radio access network (RAN), wherein the legacy communication system is decoupled by centralizing the softwarized radio access through virtual machines or containers running on servers at the edge of the cellular network [3]. While this makes the network more agile and minimizes the requirement of expensive dedicated hardware, the edge may host several applications competing for resources, thereby limiting the efficiency of radio functions [4]. Further, the unification of hybrid technologies under the 5G umbrella adds to the complexity of the problem, thereby making the use of conventional communication theoretic approaches often inadequate to achieve optimum traffic and resource management, owing to intricate mathematical modeling and complex dependencies between network and channel variables. It has therefore become indispensable the design of innovative solutions that can swiftly and effectively deal with the system complexity thanks to a fully automated, data driven approach.

Recently, learning-based techniques including supervised, unsupervised, reinforcement learning (RL), and deep learning have shown to hold an enormous potential in addressing the challenges of applying standard mathematical optimization frameworks to resource allocation problems in virtual RANs and in allowing an automatic system control [5]. However, it is worth noting that, while deep learning approaches are computationally intensive, the primary challenge associated with simpler ones such as supervised/unsupervised learning is the creation of an exhaustive dataset for training the model. Besides, in case of rapidly changing environment, frequent retraining of the model is required to achieve the desired accuracy, which can be expensive when there are stringent latency constraints. To this end, it is required to devise a framework that is easy to train in non-stationary environments, yet effective in making intelligent choices in an autonomous fashion using near real-time feedback on channel conditions and temporal variation of user demand so as to improve performance and reliability of the network.

In this work, we leverage the advantages offered by machine learning and develop a context-aware, RL-based solution to radio resource management in heterogeneous virtual RANs. Our scheme, named CAREM (Context-Aware Radio rEsource Management), can effectively cope with time-varying operating conditions thanks to a persistent interaction between the learning agent and its environment. The key contributions of this work are as follows:

1) We define CAREM, a novel framework using differential semi-gradient State-Action-Reward-State-Action (SARSA) for periodic RRM in virtual RANs. CAREM efficiently allocates radio resources in terms of link and modulation and coding scheme (MCS) for packet transmissions while meeting two of the main KPI requirements identified by 3GPP [6], namely, packet loss and latency.

This work has been supported by the EC H2020 5GPPP 5GROWTH project (Grant No. 856709.)

2) To provide an adaptive and self-sustaining solution, learning in CAREM is governed by a reward signal which acts as an evaluative feedback from the network to asses the KPI satisfaction. A snapshot of the environment in terms of Signal to Noise Ratio (SNR) and buffer state is provided as input, along with the reward signal, at the decision-making instant to make a smart and context-aware choice. High dimensionality of context variables is addressed using tile coding.
3) A proof-of-concept is provided in the context of vehicle-to-infrastructure (V2I) communications, by designing a testbed for heterogeneous radio access network and implementing CAREM over 3GPP LTE and IEEE 802.11p links using software defined radios.

Unlike Q-learning [7], which is a popular off-policy RL approach particularly useful for episodic tasks, SARSA has low per-sample variance, thereby making it less susceptible to convergence problems. Also, in a continuous task setting such as RRM where it is required to care for agent's performance during the exploration phase, online learning using SARSA is preferred as it avoids high risk actions that generate large negative reward from the environment. To the best of our knowledge, no existing work has presented such comprehensive and dynamic framework for RRM, keen on fast and reliable data transmission in heterogeneous virtual RANs.

II. Related Work

RL is a popular approach in the recent literature for radio resource provisioning problems, especially if the action corresponds to a decision-making scenario with discrete choices. RL-based schemes have been proposed for selecting the radio access technology in heterogeneous networks using network-centric [8], and user-centric approaches [9]. In [10], a policy gradient actor-critic algorithm is studied for user scheduling and resource allocation in energy-efficient heterogeneous networks. The works in [11] and [12] instead investigate dynamic spectrum access in cognitive radio networks using the RL framework, with the aim to achieve high controllability in spectrum sharing and to minimize the sensing duration.

Owing to delay-sensitivity and massive volume of data traffic in 5G access networks, a RL-based scheduling scheme is introduced in [13], [14] to minimize the packet delay and drop rate. The study in [4], instead, proposes a deep deterministic policy gradient algorithm based on actor-critic neural network and a classifier for resource control decisions. This is the most relevant work to ours, as it specifically addresses a virtualized access network and presents an implemented solution in a full-fledged testbed.

Advanced machine learning such as deep learning techniques are of interest for resource allocation problems when the size of state-action space is large, leading to slow convergence of RL approaches. A deep Q-network for channel selection is proposed in [15], [16] to adaptively learn in time-varying scenarios subject to improvement in accuracy of channel selection and maximization of network utility. The study in [17] envisions an adaptive deep actor-critic, RL-based framework for channel access in dynamic environment for single user as well as multi-user scenarios. Deep RL is explored for selection of suitable MCS for primary transmissions in cognitive radio networks in [18]. In a similar setting, the study in [19] investigates a deep learning dynamic power control method for a secondary user to coexist with the primary user. A distributed dynamic power allocation using multi-agent deep RL is developed in [20], which utilizes the channel state and quality of service information as feedback to maximize a sum-rate utility function.

In this work, by RRM we broadly refer to the action of link and MCS selection such that the learning objective, i.e., meeting the target values of the packet loss rate and latency KPIs, is achieved. In terms of actions, we find [9], [18], and [4] somewhat aligned to our work. In particular, [9] addresses the problem of link selection by modeling it as a repeated game wherein the players (mobile users) aim to maximize their throughput over long run using network assisted feedback through RL. A deep RL agent in [18] is trained at the primary receiver in a cognitive radio network to infer interference from secondary transmissions in future frames and adaptively select MCS to minimize the packet loss. We observe, however, that, although in these works the actions might be similar to ours, their learning objectives and KPIs are very different.

The resource allocation problem in [4] focuses on allocating the computation resources and maximum eligible MCS to the point of access. It is important to note that, unlike our work, none of these studies have considered the connectivity between a radio point of access and users over heterogeneous links. Besides, the radio policy selection in [4] is based on a supervised neural network classifier, which is required to be pre-trained offline using an extensive dataset. On the contrary, in MCS selection using CAREM, the policy is spontaneously learned and updated over time by its continuous interaction with the environment, thus being able to adapt continuously to time-varying channel and network dynamics.

III. System Architecture

In this section, we present the system model considered for provisioning of radio resources via CAREM. Although our approach and methodology are general and can apply to any number and type of virtual RAN technologies, we focus on a vehicle-to-infrastructure (V2I) communication environment where a cellular and a IEEE 802.11p link are available.

As shown in Fig. 1, the system architecture is composed of two interconnected blocks: the edge host (left block) and the mobile terminal (right block). The purpose of the edge host is to provide computational resources and mobile connectivity for services offered by the edge applications, which are then consumed by the mobile applications running on the mobile terminal. Connectivity between the edge host and the mobile terminal is provided through a heterogeneous RAN integrating the 3GPP LTE (bottom link in Fig. 1) and IEEE 802.11p (top link) technologies, both implemented with SDR solutions. The LTE RAN is based on srsLTE [21], an open-source SDR LTE stack implementation that offers EPC, eNB, and UE

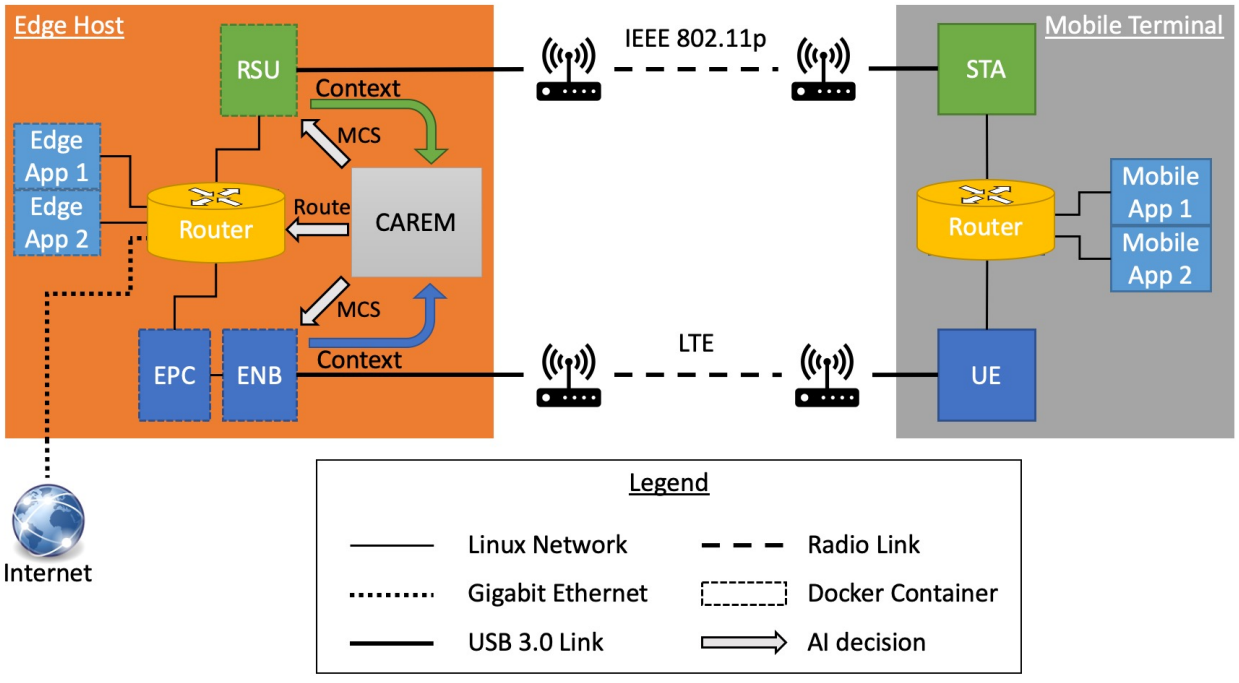


Fig. 1: System architecture.

applications. It is compliant with LTE Release 9 and supports up to 20 MHz bandwidth channels as well as transmission modes from 1 to 4, all using the FDD configuration. The 802.11p transceiver is implemented through a GNU Radio flowgraph, released by the WiME project [22], and it is interoperable with commercial IEEE 802.11p devices.

The core component of the edge host is the proposed CAREM framework, which controls the operation of the heterogeneous RAN. The algorithm periodically selects the appropriate link and the MCS to be used by the selected link for downlink packet transmission. To interact with the host operating system network stack, both the SDR solutions expose a tun/tap interface to which an IP address is assigned. A router is connected to those interfaces to steer traffic over the radio links, the host applications, and the internet, according to the link selected by CAREM. The link selection is enforced with dynamic modification to the Linux kernel routing table.

The srsLTE eNB application has been patched to run a dedicated thread that listens to and applies the MCS selected by the CAREM framework to the downlink and uplink transmission of a specific UE. The 802.11p GNU Radio flowgraph has been modified for the same purpose by adding an XMLRPC server block, which exposes a remote procedure call interface to dynamically set the MCS to be used. Furthermore, both the SDR applications have been modified to collect environment variables such as mean and variance of SNR, and buffer occupancy status at the MAC layer.

IV. The CAREM Framework

The joint impact of channel and network dynamics on RRM in wireless networks is non-trivial. To comprehensively investigate this within a machine-learning framework, it is essential to continuously map the variations in transmission channel and traffic load into a context, and learn to decide on the best link and MCS for a given context using a reward signal. The reward we use here is basically a feedback that quantifies the goodness of the decision taken. In the sequel, we discuss the components and RL algorithm used in CAREM.

A. Components of CAREM

The agent comprises a policy and a RL algorithm. The policy continuously maps observation of a context from the environment to a decision in the form of an action, while the learning algorithm updates the policy parameters based on actions, context, and reward values. The goal of the RL model is to train the agent to find an optimal policy that eventually maximizes the cumulative reward from an uncertain environment. The single components are detailed below.

Context Space. At every monitoring slot $n \in \mathbb{N}$, the agent observes a context vector $s^{(n)} \in \mathcal{S}$, takes an action $a^{(n)} \in \mathcal{A}$, which has been chosen with periodicity equal to N slots, and receives a reward value $r(s^{(n)}, a^{(n)})$ as feedback. The environment variables, namely, SNR and buffer state, influence the choice of the link and MCS in the provisioning of radio resources. Let $\gamma^{(n)}$ and $\sigma^{(n)}$ denote, respectively, the SNR and

the buffer state reported by the UE during the n-th monitoring slot. Then we define the context space $\mathcal{S} \in \mathbb{R}$ comprising context vector $s^{(n)} := \{\gamma^{(n)}, \sigma^{(n)}\}, \forall n \in \mathbb{N}$.

Action Space. The action space comprises choices for the selection of the appropriate link and MCS. Given that the network supports heterogeneous connectivity, namely, IEEE 802.11p and LTE, and several MCSs can be supported over each link, we map a link-MCS pair $\{\zeta^{(n)}, m^{(n)}\}$ for the n-th monitoring slot into a single action denoted by $a^{(n)}$. Note that an action is selected at the beginning of every decision period of duration N slots, and it is applicable to all subsequent N monitoring slots. Let the number of MCS supported over the two available links be i and j respectively, then the action space is given by $\mathcal{A} := \{a^{(n)} \in [0, i+j-1]\}$, such that $a^{(n)} = \{0, 1, \cdots, i-1\}$ when the first (e.g., IEEE 802.11p) link is selected with MCS varying from 0 to $i-1$, and $a^{(n)} = \{i, i+1, \cdots, i+j-1\}$ when the second (e.g., cellular) link is selected with MCS varying from 0 to $j-1$. The advantage of such definition of an action is that it limits the action space to a subset of discrete positive integers with low cardinality, and facilitates simultaneous selection of link as well as MCS with a single action.

Reward. Given a traffic flow, we consider as KPIs the packet loss rate at the MAC layer and the latency observed during a packet transmission within the system. To meet the KPI requirements at the UE, it is required to provide the traffic flow with radio resources such that the observed KPIs do not exceed their target values (hereinafter also referred to as thresholds). Besides meeting the KPI thresholds, it is essential to keep the observed KPIs as close as possible to the respective KPI thresholds for optimum utilization of network resources: substantially better values than the target ones would indeed translate into a waste of resources. To this end, the choice of reward function should be such that it equally accounts for both the KPIs and its value increases as the observed KPIs approach the KPI thresholds and vice versa.

Let the observed packet loss rate, target packet loss rate, observed latency, and target latency be denoted with x_o, x_{th}, l_o, and l_{th}, respectively. We define the reward value r as the sum of two reward components, namely, packet loss $r_x(\cdot)$ and latency $r_l(\cdot)$. Thus, at the n-th monitoring slot, we have:

$$r(s^{(n)}, a^{(n)}) = r_x(s^{(n)}, a^{(n)}) + r_l(s^{(n)}, a^{(n)}) \tag{1}$$

where the packet loss and latency components are given by:

$$\begin{aligned} r_x(s^{(n)}, a^{(n)}) &= 1 - \text{erf}(x_{th} - x_o), \\ r_l(s^{(n)}, a^{(n)}) &= 1 - \text{erf}(l_{th} - l_o) \end{aligned}$$

if the target KPIs are met, and by:

$$\begin{aligned} r_x(s^{(n)}, a^{(n)}) &= \text{erf}(x_{th} - x_o), \\ r_l(s^{(n)}, a^{(n)}) &= \text{erf}(l_{th} - l_o) \end{aligned}$$

otherwise.

Since the maximum and minimum value of the erf function lies between $+1$ and -1, we have: $-2 \leq r(s^{(n)}, a^{(n)}) \leq 2$. Our choice of erf for estimating individual reward components is motivated by its shape, which takes 0 value at the origin, and gradually increases (decreases) and saturates to the maximum (minimum) value in the positive (negative) direction. Consequently, for the individual reward components, in the positive region of operation, i.e., when the KPI threshold is met, the reward value is positive and it further increases to saturate to $+1$ as the observed KPI approaches its target KPI value. Likewise, in the negative region of operation, i.e., when the KPI threshold is not met, the value of the individual reward components is negative, which further reduces and saturates to -1 as the observed KPI moves away from the KPI threshold.

It may be recalled that the agent's goal in RL is to eventually maximize the cumulative reward measured as the sum of immediate reward and future rewards in the long run. Here, we adopt our definition of cumulative reward observed during slot n as the differential return $G^{(n)}$ defined in [23], i.e.,

$$G^{(n)} = r^{(n+1)} - r(\pi) + r^{(n+2)} - r(\pi) + r^{(n+3)} - r(\pi) \cdots \tag{2}$$

where, $\pi(s) : \mathcal{S} \rightarrow \mathcal{A}$, denotes the radio policy that maps the context space into actions, $r(\pi)$ being the average reward conditioned on initial state $s^{(0)}$, and subsequent actions $a^{(0)}, a^{(1)}, \cdots, a^{(n-1)}$ taken according to π. Assuming that agent's interaction with the environment since $n = 0$ has been over k slots, then the average reward is given by [23],

$$r(\pi) = \lim_{k \rightarrow \infty} \frac{1}{k} \sum_{n=1}^{k} \mathbb{E}[r^{(n)} | s^{(0)}, a^{(0:n-1)} \sim \pi]\,. \tag{3}$$

Thus, differential return essentially represents the gain in the reward value compared to average reward of the policy.

B. RL in CAREM using differential semi-gradient SARSA

In the absence of any prior knowledge of the environment, here we exploit the concept of experience-based learning using sample sequences of context, actions, and rewards observed from the actual interaction of RL agent with the environment. SARSA, an acronym for quintuple $(S_t, A_t, R_t, S_{t+1}, A_{t+1})$, is an on-policy algorithm where learning of the RL agent at time t is governed by its current state S_t, choice of action A_t, reward R_t received on taking action A_t, state S_{t+1} that the RL agent enters after taking action A_t, and finally the next action A_{t+1} that the agent chooses in new state S_{t+1} [23]. Given a context vector, the key steps involved in the learning of the SARSA approach are: (i) estimation of action values, (ii) selection of best action, and (iii) update of the action-value estimates. These are further elaborated as follows.

Action value estimation. The goodness of taking an action in a given context is quantified using action values. If action a is taken in state s under policy π, then its action value $q_\pi(s, a)$ is defined as expected differential return conditioned on state s, and action a following policy π. Mathematically,

$$q_\pi(s, a) = \mathbb{E}_\pi[G^{(n)} | s^{(n)} = s, a^{(n)} = a]\,. \tag{4}$$

Apparently, a policy π can be better than any other policy π' if $q_\pi(s, a) \geq q_{\pi'}(s, a)$. Since the context vector comprises SNR

and buffer state, context space $\mathcal{S}$ is real and an uncountable number of states are possible. Consequently, tracking action values corresponding to different contexts is not scalable. To overcome this problem, we use a practical method for action value estimation using function approximation. This yields a parametric approximation of action value function $\hat{q}_\pi(s^{(n)}, a^{(n)}, w) = \sum_{i=1}^{m} w_i x_i(s^{(n)}, a^{(n)})$, where $w \in \mathbb{R}^m$ and $x(s^{(n)}, a^{(n)})$ denote the weight and feature vector, respectively. Here, feature vector $x_i(s^{(n)}, a^{(n)})$ is generated using tile coding [24], which converts a point in the 2-dimensional context vector into a binary feature vector such that vectors of neighboring points have a high number of common elements.

Action selection. The estimation of the action values is followed by an ϵ-greedy action selection policy [23], which selects the best action so as to maximize the cumulative reward over infinite time horizon. We consider an ϵ-greedy action selection with $\epsilon = 0.1$ and ϵ-decay factor $= 0.99$. Thus, if the context at the beginning of a decision-making period is $s^{(n)}$, and the action value estimates for all possible actions $a_n = 0 \cdots |\mathcal{A}|$ in $s^{(n)}$ are obtained as $\hat{q}_\pi(s^{(n)}, a_n, w)$, the greedy action $a^{*(n)}$ is chosen with probability $1 - \epsilon$ such that $a^{*(n)} = \text{argmax}_a\, \hat{q}_\pi(s^{(n)}, a_n, w)$. The ϵ parameter decays by a factor of 0.99 in the subsequent decision period. This favors higher exploration while the environment is still unfamiliar; with progression of time, instead, it allows for further exploitation of the environment knowledge gained during the exploration, so as to maximize the expected return.

Action value update. Action values satisfy the recursive Bellman equations given as,

$$q_\pi(s, a) = \sum_{r,s'} p(s', r|s, a)[r - r(\pi) + \sum_{a'} \pi(a'|s') q_\pi(s', a')] \tag{5}$$

where $p(s', r|s, a) = \Pr\{s^{(n)} = s', r^{(n)} = r | s^{(n-1)} = s, a^{(n-1)} = a\}$, with $\pi(a'|s')$ being the probability of taking action a' in state s' under policy π. This fundamental property forms the basis of the update of the action values of the present context, based on an error term defined as the difference between a target action value and the current action value. Details on Bellman equation and the derivation of the update rule can be found in [23]. Here we consider the temporal difference learning in which the target action value for the present context is the bootstrapping estimate of the action values of the immediate next context, given by, $r(s^{(n)}, a^{(n)}) - r(\pi) + \hat{q}_\pi(s^{(n+1)}, a^{(n+1)}, w)$. Since the difference in action value estimates of successive contexts drives the learning procedure, the error is termed as temporal difference error δ. Subsequently, δ updates the average reward $r(\pi)$ and weight vector w using gradient descent. Note, however, that the bootstrapping target itself depends on the weight vector. Consequently, it is biased and does not produce a true gradient descent, hence this is referred to as a semi-gradient method.

The workflow of the CAREM RL algorithm is presented in Algorithm 1. Parameters including decision-making period N, step size α, learning rate β, weight vector for learning of action values w, and the average reward estimate $r(\pi)$ are initialized at the start of the algorithm. After observing the context vector, reinforcement learning takes place using differential semi-gradient SARSA, as discussed above. For periodic-decision making (i.e., $N > 1$), the mean reward and weighted mean context observed over the last decision period are used for learning the action values in the subsequent decision period. The weights y_n in the weighted mean context are assigned such that the latest context has the highest weight. Although they can be arbitrarily set, in our experiments, we fix them as $1, 2, \ldots, N$, in accordance with the temporal sequence of the monitoring slots.

Algorithm 1 Workflow in CAREM framework

Define N, Initialize $\alpha, \beta \in (0, 1]$
Initialize $w \in \mathbb{R}^m$ arbitrarily, $w \geq 0$, $r(\pi) = 0$
Initialize context s_0, and action a_0
for the h-th decision period, $h = 1, 2, \cdots$ **do**
  for $n = 1, 2, \cdots, N$ **do**
    if $h = 1$ **then**
      if $n = 1$ **then**
        $s^{(n)} = s_0, a^{(n)} = a_0$
      else
        Observe $s^{(n)}, a^{(n)} = a_0$
    else
      if $n = 1$ **then**
        $s^{(n)} = s^{(h)}, a^{(n)} = a^{(h)}$
      else
        Observe $s^{(n)}, a^{(n)} = a^{(h)}$
    Take action $a^{(n)}$, and evaluate reward $r(s^{(n)}, a^{(n)})$
  $r(s^{(h)}, a^{(h)}) = \sum_{n=1}^{N} r(s^{(n)}, a^{(n)})/N$ ▷ Find mean reward over the h-th decision period
  $s^{(h+1)} = \sum_{n=1}^{N} y_n s^{(n)} / \sum_{n=1}^{N} y_n$, such that $y_n > 0$ and $y_N > y_{N-1} > \cdots y_1$ ▷ Find weighted mean of context observed over the h-th decision period
  Compute action values $\hat{q}_\pi(s^{(h+1)}, , w)$ for all possible actions in $s^{(h+1)}$
  Choose $a^{(h+1)}$ as a function of $s^{(h+1)}$ using the ϵ-greedy policy
  $\delta \leftarrow r(s^{(h)}, a^{(h)}) - r(\pi) + \hat{q}_\pi(s^{(h+1)}, a^{(h+1)}, w) - \hat{q}_\pi(s^{(h)}, a^{(h)}, w)$ ▷ Evaluate temporal difference error
  $r(\pi) \leftarrow r(\pi) + \beta\delta$ ▷ Update average reward estimate
  $w \leftarrow w + \alpha\delta\nabla\hat{q}(s^{(h)}, a^{(h)}, w)$ ▷ Update weights
  $s^{(h)} \leftarrow s^{(h+1)}$
  $a^{(h)} \leftarrow a^{(h+1)}$

V. Performance Evaluation

In this section, we evaluate the performance of the CAREM framework using our testbed implementation. We consider two cases: (a) $N = 1$, which corresponds to per-slot (i.e., 100 ms) decision making, and $N = 10$, where a decision is periodically made every second. We first discuss the variation of the KPI values observed in our testbed implementation of CAREM. Since this is a limited scenario, we also execute CAREM over a longer time horizon using Matlab simulations in a heterogeneous vRAN environment and analyze its average

performance. The simulations are carried out for a duration of about 40 hours. Based on the obtained results, we discuss the convergence of reward values for each of the two above operational settings and the variation of the KPI values with respect to time and the context variables.

A. Testbed-based KPIs assessment

Using our testbed implementation of the CAREM framework, we measured the variation of the KPI values over time, when per-slot and periodic decision making are executed. According to the 3GPP specifications for 5G [6], the KPI thresholds are set at 0.1 s for latency and 0.01 for packet loss. The results, presented in Figures 2a and 2b, show that, except for an initial exploration period, the observed KPI values remain below their respective thresholds, thereby satisfying the performance requirements. Compared to per-slot decision making, packet losses and latency are higher in the case of $N = 10$ (note the different y-axis scale in the two plots), as the action executed by the CAREM framework during a decision making interval may not be the optimum choice for all the slots in that interval. Also, owing to the higher values of observed packet losses and latency, the low reward values lead to larger exploration time for $N = 10$.

B. Convergence of the CAREM RL algorithm

We further study the performance of CAREM using vRAN simulations designed in Matlab. First, we evaluate the performance of CAREM in terms of convergence of reward values on time-sequenced context. The variation of reward values as a function of time, under both the per-slot and periodic decision-making operational settings, is depicted in Fig. 3. Although the variation in reward values is higher for periodic decision making, it converges faster with respect to per-slot decision case. This may seem to contradict the observation we made based on the results obtained through the testbed implementation where periodic decision making has a higher exploration time. However, it is important to note here that the limited scenario of the testbed implementation cannot represent a wide range of variations in the network and channel conditions. Consequently, given a quasi-stationary scenario, the periodic decision setting may spend a longer time exploring the solution space, but it converges faster in the presence of a non-stationary environment over a longer time horizon. This is primarily due to the averaged values of context variables and reward that are used in periodic decision making, which tends to smoothen sharp variations thereby expediting the learning process. It follows that, in comparison to per-slot decisions, periodic ones not only reduce the computational efforts in the system, but they also lead to faster convergence of reward values, and hence rapid learning.

C. Variation of KPI values with SNR

Figures 4a and 4b depict the variation of the observed latency and packet loss averaged over context variable SNR, for per-slot and periodic decision making, respectively. In both the cases, latency is almost constant with respect to

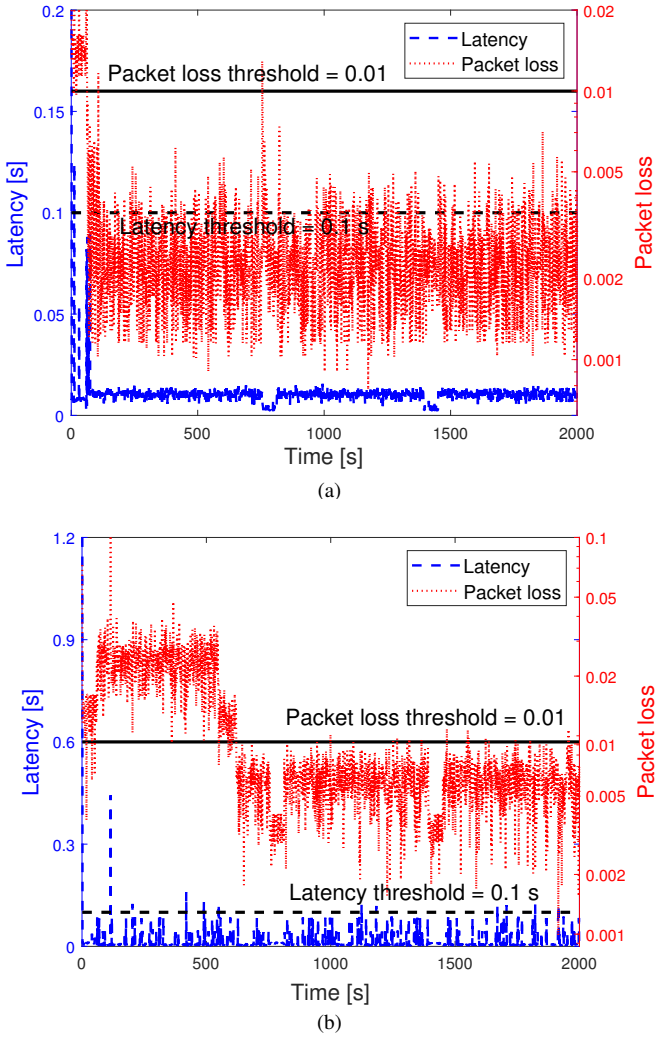


Fig. 2: Variation of KPIs observed over time using the testbed implementation: (a) $N = 1$, (b) $N = 10$.

SNR. This may be surprising, as one would expect that, with the increase in SNR, the number of possible retransmissions required for successful packet reception at the UE, and hence the observed latency, should reduce. However, in this case the retransmission delay in the network is much less as compared to the queueing delay at the buffer; consequently, the impact of the SNR on the observed latency is negligible. Unlike latency, we observe that an increase in SNR causes the packet loss to reduce for both $N = 1$ and $N = 10$, which is as expected. Additionally, note that the values of packet loss in Figures 2a and 2b may exceed the threshold during the learning phase, but when averaged over SNR, they fall below the threshold, thereby meeting the KPI requirements.

D. Variation of KPI values with buffer state

Similar to SNR, next we average the KPI values over the other context variable. The variation of packet loss and latency with respect to buffer state for different decision-making

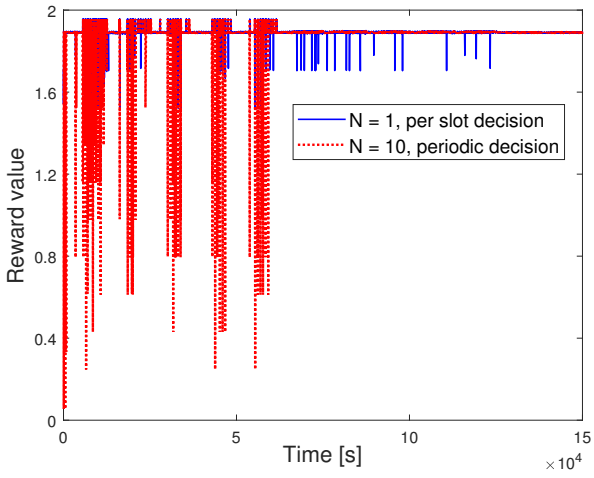
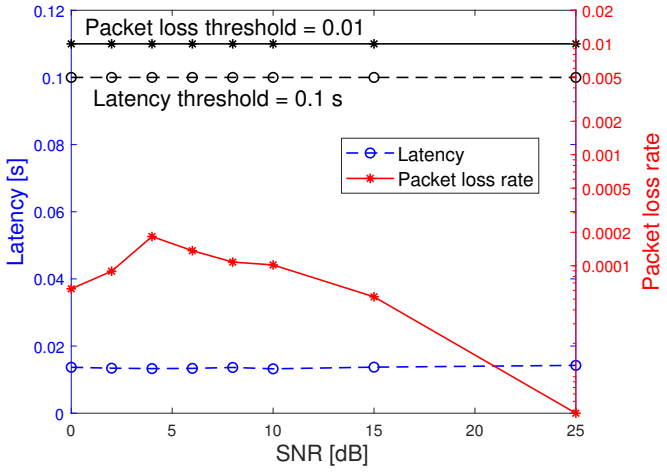
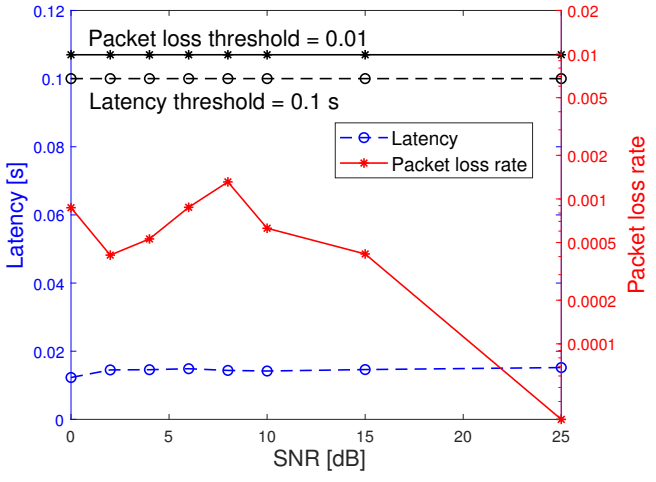


Fig. 3: Convergence of the reward for per-slot decision making ($N = 1$), and periodic decision making ($N = 10$).



(a)



(b)

Fig. 4: Variation of KPIs with SNR: (a) $N = 1$, (b) $N = 10$.

periodicity are presented in Figures 5a and 5b, respectively. As discussed earlier, since the latency is largely governed by the buffer state, we observe a linear increase in latency as the number of packets in the buffer increases. It may be noted that, since we have considered a single-UE scenario in our implementation, an increase in the vRPA buffer state values does not significantly add to the packet losses. Also, the packet loss measured in case of $N = 10$ in Figures 4b and 5b is always higher in comparison to that obtained for $N = 1$ in Figures 4a and 5a, because in the former case, the corresponding actions are computed based on averaged context and reward values and they need not necessarily be optimal for all slots in a decision period. Nevertheless, the attained KPI values always meet the required threshold once learning is completed.

VI. Conclusions and Future Work

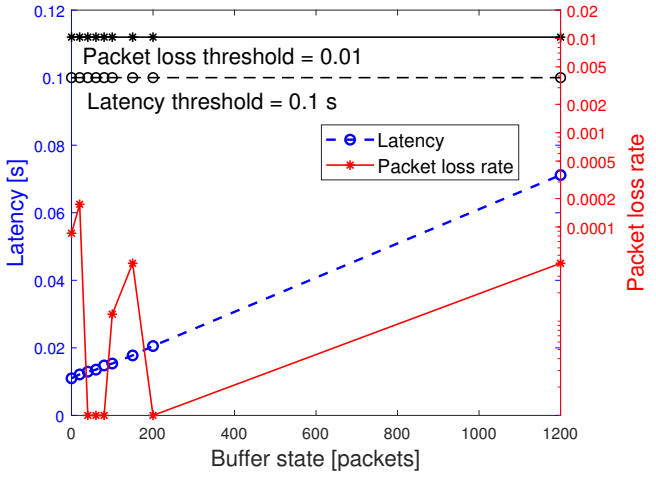
We have proposed CAREM, a novel RL-based framework that efficiently allocates radio resources in terms of link and MCS for packet transmissions in heterogeneous virtual RANs. The choice of the RL algorithm, actions, and reward functions have been made so that the resource utilization is optimized with respect to dynamic and non-stationary environment, with minimum computation efforts. We have also provided a proof-of-concept of our solution, by developing a testbed that leverages an LTE and an IEEE 802.11p SDR implementation. We have evaluated CAREM under two operational settings, with different decision-making periodicity. Furthermore, through large-scale simulations, we demonstrated that the RL algorithm converges faster when a longer decision-making periodicity is adopted, although the packet loss observed in this case is slightly higher than in the case of a per-slot decision-making process. Nevertheless, as the learning process of the model saturates, actions are chosen such that both the observed KPIs, latency and packet loss, always satisfy their target values. Finally, we remark that CAREM is a promising starting point to the development of 5G heterogeneous networks, where the advantages of different radio technologies can be fully exploited to maximize the performance and the robustness of the network. Additionally, it effectively addresses the need for a solution that can swiftly adapt to the underlying channel-network dynamics for context-aware radio resource allocation in heterogeneous virtual RANs.

Future work will focus on extending the framework implementation and performance evaluation in the case of additional SDR technologies and multiple UEs connected to the virtual RAN.

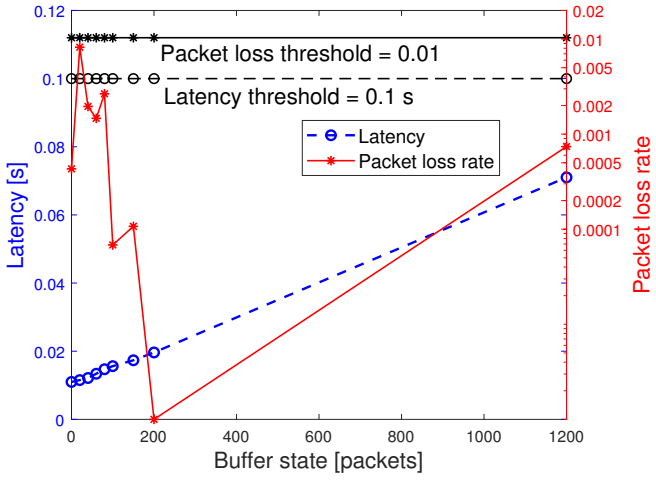
References

[1] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, "What will 5G be?" *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.

[2] T. O. Olwal, K. Djouani, and A. M. Kurien, "A survey of resource management toward 5G radio access networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 1656–1686, 2016.



(a)



(b)

Fig. 5: Variation of KPIs with buffer state: (a) $N = 1$, (b) $N = 10$.

[3] A. Gopalasingham, D. G. Herculea, C. S. Chen, and L. Roullet, "Virtualization of radio access network by virtual machine and docker: Practice and performance analysis," in *Proc. IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, Jul. 2017, pp. 680–685.

[4] J. A. Ayala-Romero, A. Garcia-Saavedra, M. Gramaglia, X. Costa-Perez, A. Banchs, and J. J. Alcaraz, "Vrain: A deep learning approach tailoring computing and radio resources in virtualized rans," in *The 25th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '19. New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: https://doi.org/10.1145/3300061.3345431

[5] Y. Fu, S. Wang, C. Wang, X. Hong, and S. McLaughlin, "Artificial intelligence to manage network traffic of 5G wireless networks," *IEEE Netw.*, vol. 32, no. 6, pp. 58–64, Dec. 2018.

[6] *3GPP TS 23.501 V16.3.0 Technical Specification Group Services and System Aspects; System architecture for the 5G System (5GS); Stage 2, (Release 16)*, 3GPP, 12 2019.

[7] B. Jang, M. Kim, G. Harerimana, and J. W. Kim, "Q-learning algorithms: A comprehensive classification and applications," *IEEE Access*, vol. 7, pp. 133 653–133 667, Sep. 2019.

[8] M. El Helou, M. Ibrahim, S. Lahoud, K. Khawam, D. Mezher, and B. Cousin, "A network-assisted approach for RAT selection in heterogeneous cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 6, pp. 1055–1067, Jun. 2015.

[9] D. D. Nguyen, H. X. Nguyen, and L. B. White, "Reinforcement learning with network-assisted feedback for heterogeneous RAT selection," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 6062–6076, Sep. 2017.

[10] Y. Wei, F. R. Yu, M. Song, and Z. Han, "User scheduling and resource allocation in hetnets with hybrid energy supply: An actor-critic reinforcement learning approach," *IEEE Trans. Wireless Commun.*, vol. 17, no. 1, pp. 680–692, Jan. 2018.

[11] N. Morozs, T. Clarke, and D. Grace, "Heuristically accelerated reinforcement learning for dynamic secondary spectrum sharing," *IEEE Access*, vol. 3, pp. 2771–2783, Dec. 2015.

[12] V. Raj, I. Dias, T. Tholeti, and S. Kalyani, "Spectrum access in cognitive radio using a two-stage reinforcement learning approach," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 20–34, Feb. 2018.

[13] I. Comşa, S. Zhang, M. E. Aydin, P. Kuonen, Y. Lu, R. Trestian, and G. Ghinea, "Towards 5G: A reinforcement learning-based scheduling solution for data traffic management," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 4, pp. 1661–1675, Dec. 2018.

[14] I. Comşa, R. Trestian, G. Muntean, and G. Ghinea, "5mart: A 5G SMART scheduling framework for optimizing QoS through reinforcement learning," *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 2, pp. 1110–1124, June 2020.

[15] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access in wireless networks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 2, pp. 257–265, Jun. 2018.

[16] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for distributed dynamic spectrum access," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 310–323, Nov. 2019.

[17] C. Zhong, Z. Lu, M. C. Gursoy, and S. Velipasalar, "A deep actor-critic reinforcement learning framework for dynamic multichannel access," *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 4, pp. 1125–1139, Nov. 2019.

[18] L. Zhang, J. Tan, Y. Liang, G. Feng, and D. Niyato, "Deep reinforcement learning-based modulation and coding scheme selection in cognitive heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 6, pp. 3281–3294, Apr. 2019.

[19] X. Li, J. Fang, W. Cheng, H. Duan, Z. Chen, and H. Li, "Intelligent power control for spectrum sharing in cognitive radios: A deep reinforcement learning approach," *IEEE Access*, vol. 6, pp. 25 463–25 473, Apr. 2018.

[20] Y. S. Nasir and D. Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2239–2250, Aug. 2019.

[21] I. Gomez-Miguelez, A. Garcia-Saavedra, P. D. Sutton, P. Serrano, C. Cano, and D. J. Leith, "Srslte: An open-source platform for lte evolution and experimentation," in *Proceedings of the Tenth ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation, and Characterization*, ser. WiNTECH '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 25–32. [Online]. Available: https://doi.org/10.1145/2980159.2980163

[22] B. Bloessl, M. Segata, C. Sommer, and F. Dressler, "Performance Assessment of IEEE 802.11p with an Open Source SDR-based Prototype," *IEEE Trans. Mobile Comput.*, vol. 17, no. 5, pp. 1162–1175, May 2018.

[23] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[24] A. A. Sherstov and P. Stone, "Function approximation via tile coding: Automating parameter choice," in *Abstraction, Reformulation and Approximation*, J.-D. Zucker and L. Saitta, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 194–205.