On the Application of Q-learning for Mobility Load Balancing in Realistic Vehicular Scenarios

Martín Trullenque Ortiz[1], Oriol Sallent[2], Daniel Camps-Mur[1], Josep Escrig[1], Jad Nasreddine[1] and Jordi Pérez-Romero[2]
[1)] i2CAT Foundation, 08034, Barcelona, Spain
[2)] Signal Theory and Communications Dept., Universitat Politècnica de Catalunya (UPC), 08034, Barcelona, Spain
[martin.trullenque, daniel.camps, josep.escrig, jad.nasreddine]@i2cat.net, sallent@tsc.upc.edu,
jordi.perez-romero@upc.edu

2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring) | 979-8-3503-1114-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/VTC2023-Spring57618.2023.10201091

Abstract— **One of the applications where fifth generation (5G) networks are expected to have a greatest impact is vehicular-to-everything (V2X) communications. The exchange of data among different vehicles on the network will make roads a safer environment. However, the massive usage of V2X communications leads to an increase of data traffic and resources consumption. Moreover, the mobility associated to vehicles may drain the available resources in particular cells, compromising the required quality of service (QoS) of connected users. To avoid these situations, the introduction of machine learning algorithms to perform mobility load balancing between neighboring cells arises as a promising tool to ensure to all connected vehicles their demanded resources. In this paper, we address the cell overload problem by proposing an O-RAN compliant Q-learning algorithm that dynamically adapts the handover offset between two neighboring cells to mitigate a network overload situation. The algorithm performance is assessed using realistic vehicular traces. Results show that the network overload appearing during rush hour can be successfully mitigated and the load is fairly distributed between cells.**

Keywords—Mobility load balancing, V2X, Reinforcement Learning

I. Introduction

The most significant change from previous generation of wireless communications to 5G networks has been replacing the "one size fits all" paradigm to an ecosystem that simultaneously supports a wide variety of different requirements; from unprecedented high data rates to millisecond latencies. This will be a key enabler for a wide range of applications scenarios and will motivate vertical industries to move from dedicated solutions to a more cost efficient, interoperable, and open ecosystem enabled solution platform.

One of the areas where 5G is expected to have a greatest impact is the automotive industry. In this context, vehicular communications (V2X) will see their reliability and latency requirements now met, enabling vehicles to communicate with different elements such as other vehicles, road infrastructure, network, or pedestrians. This will suppose a massive roll out of safety-related services such as automated driving and non-safety-related services such as the update of high-definition (HD) maps.

To create a common framework for mobile network operators (MNO) to provide V2X services, the 3rd Generation Partnership Project (3GPP) has developed several extensions. Firstly, it created a set of requirements for V2X services in 3GPP TS 22.186 [1]. This included support for both safety and non-safety related V2X scenarios. Secondly, it provided a set of 5G architecture enhancements in 3GPP TS 23.287 [2], where among other features, two radio interfaces were defined: a PC5 interface for direct vehicle-to-vehicle (V2V) communications and a Uu interface for communication between vehicle and the network (V2N). Thirdly, it defined in 3GPP TS 23.286 [3] a set of application enablers to ease the integration of V2X application functions with 5G networks.

The inherent mobility associated to vehicles pose different challenges in vehicular networks. On the one hand, an increase of vehicular density taking place at certain hours of the day may drain the vehicular network resources, affecting users' required QoS. On the other hand, the distribution of vehicles may be uneven thus leading to uneven distribution of traffic over the different cells composing the 5G Radio Access Network (RAN).

When these two issues take place simultaneously, mobility load balancing (MLB) arises as a promising technique to ensure that all vehicles will be served with their required QoS. Typically, this is done by diverting users located at the edge of a congested cell to their neighboring cells. Specifically, the congested cell introduces a cell individual offset (CIO) [4] to favor these reconnections. In this respect, different solutions have been proposed in the scope of generic Internet mobile users, either implementing heuristic algorithms in [5] or fuzzy logic in [6].

The introduction of machine learning (ML) has constituted a big step for leveraging MLB algorithms, especially since traffic patterns on roads have shown to exhibit a strong periodicity [7]. These patterns can be exploited by data-driven techniques to maximize MLB efficiency. In particular, the high degree of flexibility of reinforcement learning (RL) algorithms makes them most suitable for MLB. Among the different RL techniques, Q-learning provides a robust performance in scenarios with a limited number of state variables and actions. Q-learning algorithms were implemented to relieve cell overloads in an LTE scenario, either simulating pedestrian users with a network simulator [8] or generic hexagonal cell deployments [9]. More extensively, Q-learning was applied in [10] to optimize both mobile robustness optimization and MLB in the framework of self-organizing networks (SON). Other works have focused on exploiting deep reinforcement learning as their state space was very large, either proposing centralized MLB solutions [11] or joint parameters optimization [12]. Moreover, algorithms such as double deep Q-networks or actor critic have been proposed, respectively, in [13] for jointly adapting the transmitting power and CIO, and in [12] for combining CIO and antenna tilt angle optimization. However, while the former considered a sub second timestep, which would introduce significant overhead on the amount of data traffic transmitted, the latter approximated all the cells to the same load regardless of the spectral efficiency.

979-8-3503-1114-3/23/$31.00 ©2023 IEEE

Despite the wide literature of MLB in cellular networks, the amount of works in the scope of vehicular communications is limited. The authors of [14] proposed an MLB scheme in a vehicular network where users connected to their desired cell depending on the available QoS. User association was approached in [15] to balance loads in a multi-cell scenario by means of online reinforcement learning. Despite obtaining promising results, computing a global match for connecting individual users with all the cells compromised the flexibility of the presented solution.

In this context, this paper considers the problem of cell overloads in vehicular scenarios due to traffic jams, leveraging the vehicular mobility analysis done in our previous work [16]. With the ultimate objective of providing a general solution on mitigating cell overloads caused by traffic congestions, the main contribution of this paper is to make a first step and bring together the flexibility and low complexity offered by both Q-learning and the O-RAN architecture to propose an MLB algorithm that adjusts CIOs between two neighboring cells. The algorithm is capable to accommodate steep increases in the demand of resources while maximizing the spectral efficiency if cells are not at risk of being overloaded. To the best of our knowledge, this is the first attempt to study the potentials of RL to adjust CIOs in vehicular networks. The work is enriched by the fact that the evaluation of the algorithm considers *realistic* vehicular traces, which allows to assess the impact of traffic congestions on the V2X traffic in a realistic 5G deployment scenario. Furthermore, for benchmarking purposes, the proposed approach is compared to an optimal agent, defined as an agent always taking the best possible action in every state.

The remainder of this paper is as follows. Section II presents the proposed algorithm. Section III describes the considered scenario, as well as the results obtained from applying the proposed algorithm to mitigate network overloads. Finally, the paper's conclusions are drawn in Section IV.

II. Mobility load Balancing Algorithm

This section is devoted to describing the MLB algorithm. To this end, we firstly provide the considered architecture to accommodate our algorithm. Secondly, we review the fundamentals of Q-learning, on which our algorithm is based. Finally, we present the proposed algorithm.

A. Considered architecture

MLB is one of the self-optimization functionalities that the SON framework addresses. The SON framework is a set of functionalities that attempt to automate the configuration, management, and optimization of cellular networks.

In this context, different frameworks have been developed to support the integration of different algorithms to optimize network functionalities. In this work we focus on the opportunities offered by the O-RAN architecture, which constitutes a versatile framework for building the next generation RAN as well as embedding support for AI/ML technologies. To this end, O-RAN defines the radio controllers where control functions can be executed and the interfaces enabling the communication between them.

Among the different controllers, we focus on the Near-RealTime RAN Intelligent Controller (near-RT RIC), which allows to integrate custom control plane applications known as *xApps*. With respect to the standardized interfaces, the E2 interface manages the interaction between the near-RT RIC and the different E2 Nodes, considered to be gNodeBs (*gNB*). Particularly, the different *xApps* hosted in the near-RT RIC can subscribe to different E2 Nodes to obtain information such as RAN performance metrics through the E2 report service. Moreover, to apply the algorithms in the *gNB*, the *xApps* can use the E2 policy service to notify a given *gNB* the radio resource management operation that it should follow. In this context, O-RAN has combined the different services to standardize different service models to enable a correct network operation.

To efficiently perform MLB, we place our focus on two of the standardized service models: the Key Performance Matrix (KPM) service model [17] and the RAN control service model [18]. The KPM service model accommodates the collection of radio resource utilization parameters as defined in subsection 5.1.1.2 of 3GPP TS 28.552 [19]. Note that this is necessary to know if a cell can safely accommodate load from an overloaded cell. The RAN control service model provides support for control functionalities from the near-RT RIC to the different E2 Nodes. Particularly, subsection 6.6.3 describes the RAN policy service to adjust the CIO between two cells and subsection 8.5.4.1 provides the available parameters to ensure a coordinated CIO adjustment.

In this respect, Fig. 1 provides an O-RAN compliant architecture that supports our MLB algorithm. We assume a RL agent embedded in a *xApp* and executing the MLB algorithm to adjust the CIO between two neighboring cells that are subscribed to the *xApp*. The communication between the agent and the cells is done through the E2 interface as supported by the KPM and RAN control service models. In these models, the radio resource utilization information is sent through the E2 report service and the algorithm adjusts the CIO through the E2 policy service.

B. Q-learning fundamentals

Among the different techniques that can be used to implement MLB, reinforcement learning (RL) is an ML algorithm in which an agent interacts with an environment in certain state *s* and takes the action *a* that will provide a reward r. The way actions are taken at each state is the so-called policy $\pi(s)$. Typically, the RL problem is modelled as a Markov decision process (s, a, π, r). In this process the

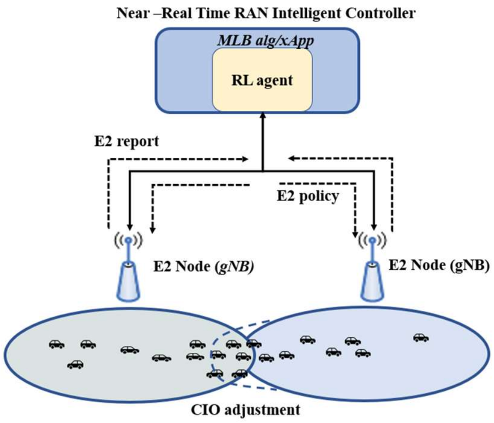

Fig. 1:Mobility load balancing architecture considered.

environment reports to the agent its current state (s_t), and once the agent has performed action (a_t) towards the environment following policy $\pi(s_t)$, it observes a reward (r_{t+1}) and the next state (s_{t+1}).

RL algorithms can be classified in two groups: model-based and model-free algorithms. In this paper we focus on model-free algorithms, which outperform the former in dynamic environments such as mobility load balancing [20]. Among the different model-free algorithms, we focus on Q-learning as it offers a high degree of robustness in situations in which a narrow range of decisions can be taken [21]. Q-learning solves RL problems by estimating a value function, denoted as $\boldsymbol{Q}$. This function estimates the overall expected reward of choosing action a_t in state s_t following a fixed policy $\pi(s_t)$. This can be expressed as $Q(s_t, a_t) = E[\sum_{t=0} \gamma^t r_t]$, where γ is bounded between 0 and 1 and is defined as the discounted reward factor, which tunes the importance of immediate reward against long term reward. Moreover, defining the dimensionality of the state space as $|S|$ and the number of different actions as $|A|$, $\boldsymbol{Q}$ can be expressed as a matrix of order $|S| \times |A|$.

Each value of $Q(s_t, a_t)$ is updated using the temporal difference method [22]. This tunes the value of $Q(s_t, a_t)$ according to the immediate reward observed and the estimate of the optimal future value, defined as $\max_{a_{t+1}} Q(s_{t+1}, a_{t+1})$ with respect to $Q(s_t, a_t)$. The optimal policy (thus, the best agent's performance) is determined by iteratively updating $\boldsymbol{Q}$ for each state and action pair until convergence is reached. More precisely, the $Q(s_t, a_t)$ update equation is as follows:

$$Q(s_t, a_t) \longleftarrow Q(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)), \tag{1}$$

where α is the learning rate, which tunes the impact a certain action a_t has on the value of $Q(s_t, a_t)$. This parameter is bounded between 0 and 1. While learning rates of 0 indicate that $Q(s_t, a_t)$ is never updated, α equals 1 indicates that $Q(s_t, a_t)$ is not considered anymore.

In order not to be caught in sub-optimal policies when updating (1), a commonly used method is the ε -greedy algorithm. Using this tool, the agent selects a_{t+1} randomly in each state with probability ε and $\max_{a_{t+1}} Q(s_{t+1}, a_{t+1})$ with probability $1 - \varepsilon$. This ensures that although the agent is in s_t and has found an action reporting a high reward, it will eventually check others to explore if there is any better. This allows analyzing all the actions in s_t.

C. Proposed algorithm

To describe our algorithm, we define as BS_i and BS_j the neighboring cells or base stations (BS). We also define the downlink received signal reference power (RSRP) by BS_i from user (u) of a set of users U to BS_i at time t as $RSRP_t^{u,BS_i}$.

In the absence of an MLB strategy, each user will connect at time t to the cell with higher RSRP. Thus, the subset of users connected to BS_i at the instant t is defined as $U_t^{BS_i}$ and composed by the users satisfying:

$$U_t^{BS_i} = \{\mathrm{u} \in \mathrm{U} \mid RSRP_t^{u,BS_i} = \max_{BS} RSRP_t^{u,BS}\}. \tag{2}$$

Denoting the capacity of BS_i as c^{BS_i}, where c^{BS_i} is defined as the number of resource blocks (RB) available for data communications, we also define its requested load $\rho_t^{BS_i}$ as follows:

$$\rho_t^{BS_i} = \frac{\sum_{u=1}^{|U_t^{BS_i}|} \mathrm{P}_t^{u,BS_i}}{c^{BS_i}}, \tag{3}$$

where $|U_t^{BS_i}|$ is the size of $U_t^{BS_i}$ and P_t^{u,BS_i} denotes the number of resource blocks (RB) demanded by the u^{th} user connected to BS_i during *t*. Note that if $\rho_t^{BS_i} \leq 1$ BS_i will safely accommodate the required traffic. Instead, if $\rho_t^{BS_i} > 1$ BS_i will be overloaded and connected users will experience QoS degradation.

To prevent this overload, BS_i negotiates with its neighboring cell BS_j to adjust a CIO, denoted as β (dB). In our solution, the CIO is adjusted in a step basis, defined as β_{step}. Thus, the value of the CIO at time *t* is updated from its previous value at time *t*-1 as $\beta_t = \beta_{t-1} + \beta_{step}$, aligned with the handover control policy defined in subsection 8.5.4.1.2 of [18]. Moreover, we bound the value of β_t in the range [β_{min}, β_{max}]. By adjusting the CIO, a user connected to cell BS_i will now be forced to reconnect to BS_j if meeting the following inequality:

$$RSRP_t^{u,BS_j} > RSRP_t^{u,BS_i} - \beta_t. \tag{4}$$

Similarly, users connected to BS_j will not connect to BS_i until the following inequality is fulfilled:

$$RSRP_t^{u,BS_j} < RSRP_t^{u,BS_i} - \beta_t\,. \tag{5}$$

However, incremental increases of β_{step} might not meet the steep increase in the resources demand due to the fast time changing dynamics of vehicular mobility. In this respect, activating the load transfer before reaching the cell's capacity enables mitigating the cell overload. To this end, we define the normalized capacity usage threshold at which BS_i activates transferring load to BS_j as ρ^{thre}, with $0 < \rho^{thre} < 1$. Thus, the closer ρ^{thre} gets to 1, the shorter the margin to balance load will be.

The definition of ρ^{thre} drives to the consideration of two operational modes for cells in the MLB algorithm: idle and active. In the event of having a cell in idle mode we attempt to set the CIO value to 0 (i.e. $\beta_t \rightarrow 0$) to maximize spectral efficiency. Instead, if BS_i's requested load is higher than ρ^{thre}, it will start to balance load to BS_j unless the latter is overloaded ($\rho_t^{BS_j} > 1$).

Having provided the operation of the algorithm we proceed to describe the state, action, and reward considered.

1) State: five variables are considered in the state space, expressing the state vector as follows:

$$\underline{s_t} = \left[\rho_t^{BS_i}, \rho_t^{BS_j}, \rho_t^{BS_i,edge}, \rho_t^{BS_j,edge}, \beta_t\right], \tag{6}$$

where $\rho_t^{BS_i}$ and $\rho_t^{BS_j}$ denote both BS_i's and BS_j's requested load. Moreover, we consider the requested load by users located at the cell edge. We define as $\rho_t^{BS_i,edge}$ the requested load of connected users to BS_i that would be transferred to BS_j if β_t was increased by β_{step}. Formally, it is the requested load of connected users to BS_i meeting:

$$RSRP_t^{BS_i} - RSRP_t^{BS_j} < \beta_{step} + \beta_t. \quad (7)$$

Similarly, we define as $\rho_t^{BS_j,edge}$ the requested load of connected users to BS_j that would be transferred to BS_i if β_t was increased by β_{step}. In this case, it is the requested load of connected users to BS_j satisfying the following criteria:

$$RSRP_t^{BS_i} - RSRP_t^{BS_j} > \beta_t - \beta_{step}. \quad (8)$$

Note that the set users fulfilling (7) and (8) can be determined from the cells performing measurement reports and assessing their RSRP with the neighboring cells. The last parameter of the state vector is the current CIO, β_t.

2) *Action:* determines the modification of the CIO between BS_i and BS_j. To this end, three different actions can be taken with β_t: increase its value by β_{step}, decrease it by β_{step} or keeping it the same. Note that favouring the reconnection of users from BS_i to BS_j by increasing the CIO by β_{step} leads BS_j to observe a decrease of its CIO value. Then, the action space, defined as A, is given by:

$$A = [-\beta_{step},\ 0,\ \beta_{step}]. \quad (9)$$

3) *Reward:* the aim of the reward is to incentivize an increase of the CIO value when a cell is in active mode ($\rho_t^{BS_i} > \rho^{thre}$) and to motivate a CIO value of 0 if there is no risk of having an overload, maximizing the spectral efficiency. In this respect, when $\rho_t^{BS_i} < \rho^{thre}$, the reward is defined as follows.

$$r_t = \begin{cases} 1 & \beta_t = 0 \; or \; |\beta_t| < |\beta_{t-1}| \\ -1 & otherwise \end{cases} \quad (10)$$

According to this expression, high rewards are given in two different situations: if no offset is in place and if the CIO absolute value is decreased from the previous timestep. Note that while the first case mimics a normal operation, the second case replicates a situation that might happen when vehicular density decreases after a traffic congestion. In turn, increasing the absolute value of the CIO (i.e., $|\beta_t| > |\beta_{t-1}|$) is penalized.

Instead, if $\rho_t^{BS_i} \geq \rho^{\text{thre}}$, the reward is:

$$r_t = min\left(\rho^{thre} - \rho_t^{BS_i}, \rho^{thre} - \rho_t^{BS_j}\right), \quad (11)$$

where the algorithm penalizes the excess of load over ρ^{thre} in any cell, motivating the increase of the CIO value in the overloaded cell. Although it compromises the average spectral efficiency (if BS_i is overloaded several users will reconnect from BS_i to BS_j having lower RSRP thus needing more resource blocks to transmit the same data), we deliver the connected vehicles the best possible average QoS.

III. Performance Evaluation

This section assesses the performance of the proposed algorithm in a realistic scenario. To this end, we have used the same simulation tool we implemented for our previous works [16] [23], which allows to analyze large amount of data while being compliant with the vehicular traces timescale. In this section, we start by providing a detailed description of the considered scenario. Then, we assess our algorithm convergence rate and its overload mitigation performance.

A. Scenario description

The evaluation scenario consists of a 12 km^2 urban area placed in the city of Cologne (Germany). The considered radio access network is composed by a set of 16 cell sites as depicted in Fig. 2. The location of the cell sites is as defined in the Telekom network available in [24]. The area of study is the central part of Fig. 2 [25], while cell sites at the edge (e.g., BS12, BS14, BS15) are taken into consideration to limit the border effects.

1) RAN deployment

We consider that each cell site operates a single cell centered at 2.1 GHz band, using frequency-division duplexing, and with 20 MHz channel bandwidth for each link direction. To this end, the selected subcarrier spacing is 15 kHz [26]. With these values, the transmission bandwidth includes 106 RBs, each one consisting of 12 consecutive subcarriers in the frequency domain. For normal cyclic prefix, 14 OFDM symbols can be transmitted per slot and per subcarrier.

The path loss model chosen is the urban area model as defined by 3GPP in Section 4.5.2 of [27]. Neither slow nor fast fading terms have been considered. To have realistic spectral efficiency values, the transmitted power at each BS has been adjusted to have a data rate around 800 kbps at the cell edge. The rest of the parameters are chosen from Annex C of [27] and are summarized in Table I.

Each user connects to the cell with the lowest path loss. We consider Shannon's bound to derive the achievable spectral efficiency in the radio link (i.e., log$_2$(1+SNR) bits/s/Hz).

2) Vehicular traffic

To apply our algorithm in a scenario as close to reality as possible, we have taken a dual approach: we are using realistic vehicular traces and we consider V2X traffic from a currently available vehicular service.

TABLE I: RADIO PARAMETERS.

BS parameter	**Value**
Antenna height	30m
Noise figure	9 dB
Antenna gain	15dB
UE parameters	**Value**
Maximum transmitted power	21dBm
Antenna gain	9 dB
Noise figure	9 dB

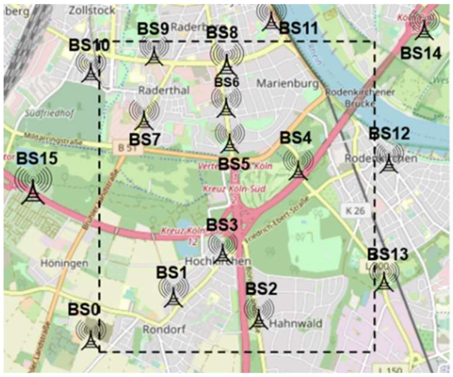

Fig. 2: Considered scenario in Cologne.

With regards to the vehicular traces, we have considered the TAPAS Cologne dataset [28], which is public and contains 24 hours of realistic vehicular traces. Out of the whole dataset, this work focuses on studying the timeframe between 17:42 and 18:32, as it is when roads become more densely congested as studied in [23].

The considered vehicular service delivers ETSI Day 1 safety services for connected cars defined based on [29]. These services comprise the generation of CAM messages, which provide periodic awareness such as the position and basic status of the vehicle to its immediate neighborhood. In this work we consider that these messages are sent using the Uu interface only. Moreover, we are not considering multimedia broadcast/multicast service (MBMS) since it is not widely available in current systems. Among the different permitted CAM interarrivals time [30], we assume that it is constant and of 500 milliseconds. The message size modelling is taken from [31]. The range at which CAM messages will be forwarded, is a square centered in the vehicle generating the CAM message and with side 400 meters. Table II summarizes the vehicular service considered.

3) Impact of vehicles mobility

Having described the cells parameters and the vehicular traffic, we assess the impact the vehicular service has on the cells deployed in terms of requested load.

Based on our previous study in [23], the cell experiencing the greatest traffic variation is BS3, as it is close to a junction. Particularly, there is a traffic jam in BS3 for vehicles merging from the highway traversing from South to North to the highway heading towards the West (BS4).

To assess the impact a traffic jam has in terms of requested network load we focus on the downlink load analysis. Indeed, the absence of MBMS provokes that a CAM message originated in a certain vehicle is forwarded in downlink to all vehicles within the target area defined in Table II. This leads to a multiplicative increase of the number of downlink CAM messages during a traffic jam. In this respect, Fig. 3 shows the requested load both in BS3 (blue) and BS4 (red) over the considered time frame. The figure shows how BS3 is overloaded from 18:05 to 18:15 approximately. In turn, the absence of traffic jams in BS4 enables to safely transfer load from BS3 to BS4, which is where the main traffic flow is heading to.

B. Q-learning training

In order to train the algorithm, we apply data augmentation to the original Cologne dataset. Particularly, we take 500 different sets of the vehicles' trajectories in the described timeframe. Each of them is defined as an episode. Moreover, to cover different situations we consider a variable size of the connected users, ranging from 65% to 80% of the whole number of vehicles. Thus, out of the 5.013 vehicles traversing the interest area, the 500 episodes consist of random samples of different sizes ranging from 3.258 to 4.010 users traversing the interest area. Moreover, to replicate the overload shown in Fig. 3, the available resources in the network have been adapted to the size of the set considered. Note that the smaller the size of the set taken is, the less representative the episode can be from the real dataset.

TABLE II: CONSIDERED VEHICULAR SERVICE

V2X traffic	Value
Vehicular service	CAM
Radio interface	Uu
MBMS capable	No
CAM interarrival time	500 ms
Packet size	300Byte with probability 1/5 170Byte with probability 4/5
Downlink message area	1.6×10^5 m^2

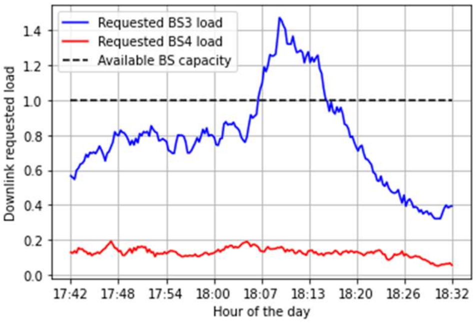

Fig. 3: Requested downlink load at BS3 and BS4.

With respect to the ε-greedy algorithm, we assume a decreasing value of ε over the 500 episodes to exploit the knowledge gained over the training phase, which we consider to be done offline. In particular, the value of ε is updated as follows:

$$\varepsilon = 0.3 - \left\lceil {}^{n}/_{100} \right\rceil * 0.05, \tag{12}$$

where n denotes the n^{th} training episode.

Furthermore, given the fact that $\rho_t^{BS_i}$, $\rho_t^{BS_j}$ and $\rho_t^{BS_i,edge}$ are continuous parameters, we quantise them in bins of 0.04 to prevent an exacerbated increase of the state space size.

With regards to the parameters defined in Section II, ρ^{thre} and β_{step} are fixed to 0.8 and 1 dB respectively. Note that both parameters are related since high values of ρ^{thre} will require a large β_{step} to adapt to the steep increase of resources demand. However, this approach might not transfer the optimal load to BS4 due to its lack of granularity. The CIO range, β_{min}, and β_{max}, have been bounded to -4 and 12 dB respectively, favoring positive values for the load transfer from BS3 to BS4 but also introducing negative offset values to avoid biasing the best policy search. Table III provides a full description of the parameters chosen.

In order to validate our testing, Fig. 4 shows the average reward obtained for each of the 500 training episodes if a

TABLE III: ALGORITHM PARAMETERS

Training parameter	Value
Number of episodes	500
Timestep considered	15 seconds
Number of epochs	200
Learning rate (α)	0.9
Discount rate (γ)	0.75
State space size ($\lvert S \rvert$)	680.000
Action space size ($\lvert A \rvert$)	3
ρ^{thre}	0.8
β_{step}	1 dB
β_{min}	-4 dB
β_{max}	12 dB

timestep of 15 seconds is considered. It can be appreciated how the average reward gets to its maximum value after approximately the first 450 episodes.

C. Performance evaluation in inference mode

To assess the performance of the algorithm, we evaluate the trained algorithm using the described scenario. We consider the totality of the Cologne dataset. Note that for the evaluation, ε is set to 0.

In this respect, Fig. 5 and Fig. 6 present, respectively, the evaluation of the algorithm in terms of requested load and the CIO evolution over time. Fig. 5 illustrates that the original requested load in BS3 (plotted in blue) is successfully balanced (green plot), and the overload is mitigated. In turn, BS4 (plotted in purple) experiences an increase of the load, ranging from an original 0.14 (red line) to a maximum value of 0.88 observed at 18:10. Beyond the observation that BS3 load is lower than BS4 load, the algorithm considers from training experience that BS3 may need room for accommodating more load. However, increasing the CIO value (blue plot of Fig. 6) causes that a UE now reconnected to BS4 will experience lower spectral efficiency than with BS3, having an increase on the required resource blocks. Specifically, considering the time frame during which load is being balanced, this resource demand increase is of a 7.96%, which is 25.61% less than the one we would observe if a static handover offset of 10 dB was applied as in [23].

D. Benchmarking with the optimal agent

This section studies how close the proposed approach is from an optimal agent. To this end, we consider that the optimal agent operates with the same state, action, and reward function as the agent of the proposed algorithm. The main difference between the optimal agent and our algorithm resides on the policy. While our algorithm learns from the described training phase, we consider that the optimal agent always takes the action (i.e., adjusts the CIO) providing the highest overall expected reward in each state (i.e., $\pi(s_t)$ such as $a_t = \underset{a_t}{\operatorname{argmax}} Q(s_t, a_t)$). For computational simplicity, we assume $\gamma = 0$. The rest of the parameters are as defined in Table III.

In order to compare the performance of our algorithm with an optimal agent, we evaluate both of them considering a broader scope than the original Cologne dataset. To this end, we apply data augmentation and consider 50 different sets of trajectories from the original dataset as in the Q-learning training subsection. However, the different episodes consider a larger number of vehicular trajectories, ranging from 85% to 95% of the original dataset. We also set ε to 0.

Fig. 7 shows the boxplot of the overload time for each of the 50 episodes in three situations: when no MLB is used, when the proposed algorithm is applied, and when the optimal agent is evaluated. It can be observed how our algorithm effectively reduces the overload time compared to the legacy cases. In particular, the minimum overload time without applying our algorithm (16 minutes and 30 seconds) is more than three times the maximum overload time if our algorithm is applied (4 minutes and 45 seconds). Although the optimal agent would have achieved a complete overload mitigation, we obtain a remarkable average reduction overload time of a 91.87%.

Beyond presenting the overload time reduction, Fig. 8 presents the difference between the requested load of BS3 and BS4 when roads are most populated. This indicates how the load is being effectively balanced. It can be appreciated how the requested load of both cells is significantly balanced if our MLB algorithm is in place, decreasing the average value by a 67.67%. In this case the optimal agent provides an average load difference of 0.042, which sets our algorithm a 29.17% away from the optimal agent.

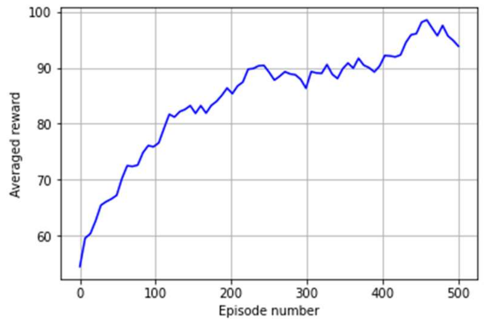

Fig. 4: Averaged reward over the 500 experiments.

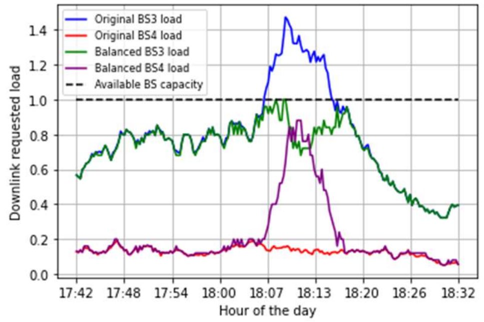

Fig. 5: Algorithm impact on the requested load.

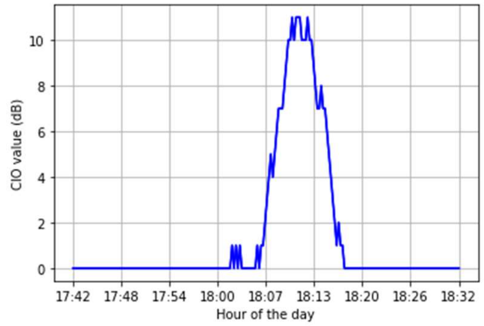

Fig. 6: CIO evolution over time.

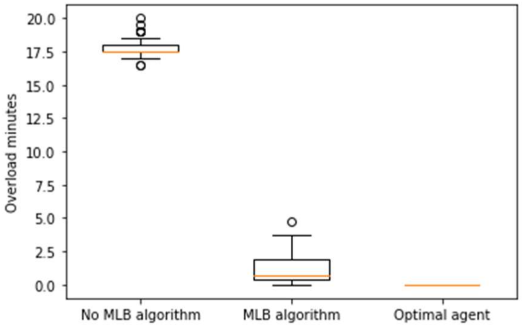

Fig. 7: Overload mitigation performance with the evaluation dataset.

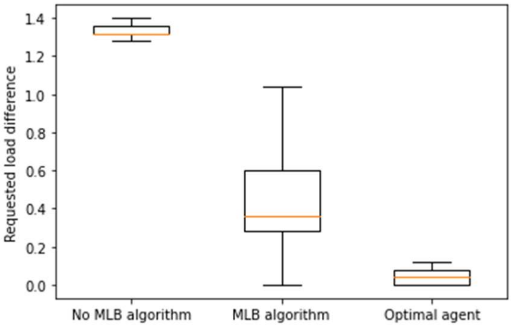

Fig. 8: requested load difference of both cells.

IV. Conclusions and Future Work

This work has proposed and evaluated a Q-learning algorithm to mitigate cell overload situations in vehicular scenarios using realistic vehicular traces. The presented results have shown that the algorithm effectively mitigates the presented overload during the studied timeframe, transferring most of its load to the neighboring cell with an increase in the demanded resources of a 7.96%. Besides, the algorithm has delivered a remarkable performance, mitigating the presented overload during 91.87% of the time.

Based on these promising results, our future work envisages the generalization of the proposed algorithm to more complex scenarios, where different context information of the vehicular environment can be acquired, and the flexibility brought by Q-learning algorithms can be further exploited. Among the different challenges to be addressed, we intend to focus on studying the coordination between overloaded neighboring cells and the load distribution between an overloaded cell among different neighbors. With this future solution, we aim to provide an autonomous, coordinated, and scalable response to the challenges vehicular mobility poses on radio resources availability.

Acknowledgment

This research was partially supported by the Spanish Centre for the Development of Industrial Technology and the Ministry of Economy, Industry and Competitiveness under grant/project CER-20191015 / Open, Virtualized Technology Demonstrators for Smart Networks (Open-VERSO) and by the Spanish Ministry of Economic Affairs and Digital Transformation and the European Union – Next Generation EU, in the framework of the Recovery Plan, Transformation and Resilience (Call UNICO I+D 5G 2021, ref. number TSI-063000-2021-29).

References

[1] 3GPP, "3GPP TS 22.186 version 16.2.0 Release 16," 2020.

[2] 3GPP, "3GPP TS 23.287 version 16.3.0 Release 16".

[3] 3GPP, "3GPP TS 23.286 version 16.6.0 Release 16," 2021.

[4] 3GPP, 3GPP TS 36.300 version 9.10.0 Release 9, 2013.

[5] A. Lobinger, S. Stefanski, T. Jansen and I. Balan, "Load Balancing in Downlink LTE Self-Optimizing Networks," in *2010 IEEE 71st Vehicular Technology Conference*, 2010.

[6] M. M. Hasan, S. Kwon and J.-H. Na, "Adaptive Mobility Load Balancing Algorithm for LTE Small-Cell Networks," *IEEE Transactions on Wireless Communications,* vol. 17, no. 4, pp. 2205-2217, 2018.

[7] J. J. Gonzalez-Delicado, J. Gozalvez, J. Mena-Oreja, M. Sepulcre and B. Coll-Perales, "Alicante-Murcia Freeway Scenario: A High-Accuracy and Large-Scale Traffic Simulation Scenario Generated Using a Novel Traffic Demand Calibration Method in SUMO," *IEEE Access,* vol. 9, pp. 154423-254434, 2021.

[8] S. S. Mwanje and A. Mitschele-Thiel, "A Q-Learning Strategy for LTE Mobility Load Balancing," in *IEEE 24th Symposiom on Presonal, Indoor and Mobile Radio Communications*, London, 2013.

[9] M. Z. Asghari, M. Ozturk and J. Hämäläinen, "Reinforcement Learning Based Mobility Load Balancing with the Cell Individual Offset," in *IEEE 93rd Vehicular Technology Conference* , 2021.

[10] S. Mwanje, L. C. Schmelz and A. Mitschele-Thiel, "Cognitive Cellular Networks: A Q-Learning Framework for Self-Organizing Networks," *IEEE Transactions on Network and Service Management,* vol. 13, no. 1, pp. 85-99, 2016.

[11] Y. Xu, W. Xu, Z. Wang, J. Lin and S. Cui, "Load Balancing for Ultradense Networks: A Deep Reinforcement Learning-Based Approach," *IEEE Internet of Things Journal,* vol. 6, no. 6, 2019.

[12] H.-H. Chang, H. Chen, J. Zhang and L. Liu, "Decentralized Deep Reinforcement Learning Meets Mobility Load Balancing," *IEEE Transactions on Networking,* 2022.

[13] G. Alsuhli, H. A. Ismail, K. Alansary, M. Rumman, M. Mohamed and K. G. Seddik, "Deep Reinforcement Learning-based CIO and Energy Control for LTE Mobility Load Balancing," in *IEEE Annual Consumer Communications & Networking Conference*, 2021.

[14] N. Aljeri and A. Boukerche, "Load Balancing and QoS-Aware Network Selection Scheme in Heterogeneous Vehicular Networks," in *2020 IEEE International Conference on Communications*, 2020.

[15] Z. Li, C. Wang and C.-J. Jiang, "User Association for Load Balancing in Vehicular Networks: An Online Reinforcement Learning Approach," *IEEE Transactions on Intelligent Transportation Systems,* vol. 18, no. 8, pp. 2217-2228, 2017.

[16] M. Trullenque, O. Sallent, D. Camps-Mur, J. Escrig and C. Herrranz, "Analysis of Vehicular Scenarios and Mitigation of Cell Overload due to Traffic Congestions," in *2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring)*, Helsinki, 2022.

[17] O-RAN Alliance, O-RAN Working Group 3 Near-Real-time RAN Intelligent Controller E2 Service Model (E2SM) KPM.

[18] O-RAN Alliance, O-RAN Working Group 3 Near-Real-ime RAN Intelligent Controller E2 Service Model (E2SM), RAN Control.

[19] 3GPP, TS 28.552 version 16.6.0 Release 16.

[20] R. Sutton and A. Barto, Reinforcement Learning: An Introduction, The MIT Press, Second edition, 2018.

[21] C. Watkins and P. Dayan, "Q-learning," *Machine Learning,* vol. 8, pp. 279-292, 1992.

[22] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, The MIT Press, 2018.

[23] M. Trullenque, O. Sallent, D. Camps-Mur, J. Escrig, C. Herranz-Claveras, J. Nasreddine and J. Pérez-Romero, "On Alleviating Cell Overload in Vehicular Scenarios," in *IEEE 96th Vehicular Technology Conference*, 2022.

[24] CellMapper, "Signal tiles and towers," CellMapper, [Online]. Available: https://www.cellmapper.net/. [Accessed 4th September 2021].

[25] OpenStreetMap contributors, *Planet dump retrieved from https://planet.osm.org,* https://www.openstreetmap.org, 2017.

[26] 3GPP TS 38.104, "5G; NR; Base Station (BS) radio transmission and reception," Jul. 2018.

[27] 3GPP , "TR 36.942 v16.0.0.0 (Release 16)," 2020.

[28] S. Uppoor and M. Fiore, Large-scale urban vehicular mobility for networking research, Amsterdam: VNC, 2011.

[29] ETSI TS 102 637 - 2, "Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications.," Mar. 2011.

[30] ETSI, TS 103 175 Intelligent Transport Systems (ITS); Cross Layer DCC Management Entity for operation in the ITS G5A and ITS G5B medium, 2015.

[31] L. M. W. Lopez, C. F. U. Mendoza, J. Casademont Serra and D. Camps Mur, "Understanding the impact of the PC5 resource grid design on the capacity and efficiency of LTE-V2X in vehicular networks," *Wireless communications and mobile computing,* 2020.