

QoS Aware Slice Resource Management using Deep Reinforcement Learning in IoT Networks

Kamran Zia⁺, Alessandro Chiumento⁺, Paul Havinga⁺, Roberto Riggio^{*}, Yanqiu Huang⁺

⁺*Pervasive Systems Group, EEMCS University of Twente, Enschede Netherlands*

^{*}*Information Engineering Department, Polytechnic University of Marche, Ancona Italy*

[k.zia, a.chiumento, p.j.m.havinga, yanqiu.huang]@utwente.nl, *r.riggio@univpm.it

Abstract—WiFi, a widely used technology in IoT, provides QoS through IEEE 802.11e EDCA Access Categories (AC) which have fixed configuration with strict priorities. This makes WiFi QoS unsuitable to the rising QoS diversity, changing wireless conditions and network dynamics. QoS slicing, where network resources are divided into chunks for diverse QoS requirements, is a potent technology that can provide more flexible, adaptable and highly configurable QoS in WiFi based IoT networks. However, resource management of QoS slices with diverse IoT applications, limited network capacity and varying channel conditions is a complex and challenging task. Traditional queuing theoretic and optimization models become intractable to solve such complex and dynamic problem. Therefore, we have developed a Deep Reinforcement Learning (DRL) based slice resource management scheme to meet IoT QoS requirements in a real world 5Gempower controlled SDN network. Our proposed scheme outperforms Airtime Excess Round Robin (ATERR) scheme and no slicing-based scheme in terms of slice throughput (QoS) satisfaction. Moreover, our proposed scheme is tested in real world environment and can adapt to the changing slice requirements in an IoT network.

Index Terms—WiFi Slicing, Deep Reinforcement Learning, QoS in WiFi, Adaptive Slice Management, Airtime Management

I. INTRODUCTION

Internet of Things (IoT) has been built upon interconnection of physical devices, sensors, medical appliances and Radio Frequency Identification Tags. In the last few years, IoT networks have expanded at a fast pace in different industry verticals [1] including smart factories, healthcare, smart cities and agriculture sectors. Many new applications have been developed that employ machine learning algorithms by collecting data from distributed sensors, appliances and machines and draw intelligent inferences [2]. To enable IoT use cases, there are various Quality of Service (QoS) requirements of IoT sensors and applications in terms of throughput, latency, and reliability. More often, meeting these QoS requirements is crucial for the success of the IoT use cases.

Due to requirement of reliable QoS in present day IoT networks, access and core networks have incorporated various techniques to provide QoS to the different IoT traffic in the network. Wired networks enable QoS through Differentiated Services (DS) where flows are classified and managed separately as Voice, Video or Best Effort (BE) flows. Similarly, IEEE 802.11e based networks (WiFi) use Enhanced Distributed Channel Access (EDCA) Access Categories (AC) to differentiate between different types of traffic [3]. The QoS in WiFi networks through EDCA has 4 QoS categories and has fixed configuration of each category; therefore, it cannot support the rising diversity in QoS requirements and varying dynamics of the IoT networks. Although, IEEE 802.11aa defines further segregation of video and voice traffic to two classes and uses credit based schedulers to emulate a total of 6 access categories in WiFi however, channel access parameters like Contention Window (CW) and Arbitrary Inter Frame Spacing

(AIFS), which primarily control service differentiation, are defined only for the proposed 4 EDCA categories. Moreover, most of the channel access is consumed by voice and video traffic, due to very small contention windows compared to best effort and background traffic, thus affecting best effort traffic flows which are mostly being used by IoT sensors and devices. As a result, their QoS is adversely affected.

Many research works have proposed throughput enhancement [4] and delay reduction [5] schemes in WiFi based IoT networks for IoT QoS. Some works have even employed AI and ML as well in their works for QoS satisfaction and they always rely on EDCA access categories to provide QoS in WiFi based IoT networks [4] [6]. However, the problem is that EDCA has limited number of access categories with fixed configurations. Therefore, in order to dynamically manage the varying range of IoT QoS requirements with in the available resources, a more flexible and highly configurable QoS framework is required in WiFi based IoT networks that can also prioritize QoS flows over one another. Moreover, WiFi QoS mechanism needs to have more granular control over network resources through more access categories to support large number of applications/services.

To address the shortcomings in present WiFi based networks, Software Defined Networking (SDN) provides a flexible framework to perform QoS slicing in the network where network resources are sliced into multiple chunks. Each resource chunk can then be assigned to QoS slices and can then be managed through SDN controller to provide desired connectivities (QoS) in the network [7] [8]. This framework, on the one hand, enables creation of more access categories in WiFi networks and on the other hand, helps preserve QoS information while traffic traversing from differentiated services to access categories in wireless medium. However, managing network resources for access categories or QoS slices is a complex and challenging task. It involves a dynamic and adaptive management of network resources while keeping in view the channel conditions, applications QoS requirements and available network capacity. To address this challenge, we have proposed a Deep Reinforcement Learning (DRL) based slice resource management scheme that can learn network dynamics and provide improved QoS to the applications and sensors in the wireless IoT network. Our main contributions in this paper are:

- We have developed a slice throughput requirement estimation algorithm in 5Gempower SDN controller to determine different QoS slice throughput requirements autonomously.
- We have proposed an DRL based dynamic slice resource management algorithm that runs in a SDN controller and provides desired throughput to all slices in the WiFi based IoT network.
- Our algorithm is able to adapt to the changing throughput requirements and wireless conditions to provide desired throughput to the QoS slices

- Our proposed algorithms are tested in real world experimental test bed for validation.

II. RELATED WORK

QoS in wireless networks is an ongoing research problem that poses significant challenges compared to wired networks due to unpredictable nature of wireless medium. As such, many techniques have been proposed in literature to address the QoS problem in WiFi based networks. Authors in [9] have proposed a QoS slicing based solution in WiFi networks using 5Gempower SDN controller to provide QoS to two different types of flows. The system is able to provide more resources to QoS slices to provide better QoS to the applications compared to BE applications. However, they only considered two slices in the network. Authors in [10] proposed a Quadratically Constrained Quadratic Program (QCQP) based algorithm to assign airtime to different network slices to meet the latency requirements of applications in the network. The proposed scheme however, has limitations to employ such techniques in real-time to meet QoS requirements.

Authors in [11] proposed a Proportion Time Deficit Round Robin (PT-DRR) based slice resource assignment algorithm and used airtime allocation to network slices for QoS provisioning in the network. The proposed scheme is based on queuing theory and employs quantum assignments to the network slices based on airtime being used by different queues. However, airtime for packets also depends on the physical layer parameters like Modulation and Coding Scheme (MCS) and CW etc therefore, quantum assignments can only indirectly control the total airtime of each slice. The proposed scheme calculates the ratio of quantum for each slice based on available channel capacity and slice requirements and allocates them to the available slices. Since wireless medium and application requirements are a continually changing phenomenon, fixed quantum assignments can lead to poor QoS in the network.

Similar to PT-DRR, authors in [7] have proposed Adaptive Time Excess (ATERR) based resource allocation in QoS slicing framework to enable QoS in a WiFi network. In this work, the authors calculate the quantum assignments to the slice based on time excess and requested airtime rather than deficit [11] for the available network slices. The time excess is calculated based to consumed airtime by packets and remaining quantum for the slice (airtime consumed - remaining quantum). A negative time excess will let slice transmit its packets while packet transmission stops as soon as time excess become positive, indicating all quantum has been consumed in that round of scheduling.

Although quantum assignments can indirectly control the airtime each slice gets in the wireless medium, there are various other factors that affect the airtime a device gets in the WiFi network. These factors include Modulation and Coding Schemes (MCS), CW and ACK policy etc. A lower MCS will consume more airtime in the wireless medium to transmit packets (because of low bit rate) and hence the assigned quantum value will not be able to provide the desired QoS (throughput or latency) to the network slice. For these reasons, quantum assignments cannot be done based on requested airtime as it may vary based on MCS being used at the physical layer of the network. Therefore, knowing the MCS being used at the physical layer is also important during quantum assignments.

Although network slicing and slice resource management has been proposed in literature, it lacks adaptive resource management to continually meet QoS in IoT networks. In this context, AI and ML based approaches can help develop algorithms that can efficiently utilize network resources and can learn network dynamics

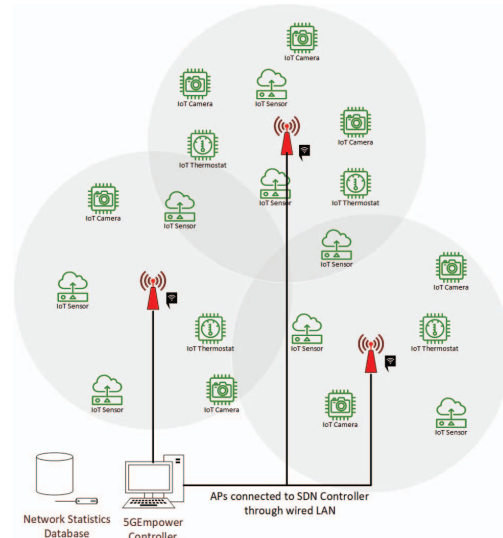


Fig. 1. Network Model

to enable a dynamic and evolving QoS management [6] [12]. Owing to AI and ML benefits and their adaptive and dynamic nature, we have proposed a DRL based slice resource management scheme for meeting varying throughput requirements of IoT traffic flows in a SDN controlled wireless IoT network.

III. NETWORK SETUP AND QoS SLICING

In order to develop the DRL based resource management for QoS slices, we have used the 5Gempower SDN framework [13]. 5Gempower is a multi-RAT SDN framework that supports both WiFi and LTE networks. It provides a web based user interface to interact with the network applications, create access control lists and create/delete QoS slices. It also provides a Python based Software Development Kit (SDK) to develop network applications which run as services on the SDN controller. The network statistics are collected from the empower agent running inside the Access Points (AP) being controlled by the 5Gempower SDN controller. The pictorial representation of the network model is given in Figure 1.

The network model comprises of an SDN controller communicating with an AP. The IoT devices are simulated using Raspberry Pi boards that are associated with the AP over IEEE 802.11n based WiFi standard. IoT devices are randomly distributed around the AP and are receiving traffic from the AP in the downlink. Each device is running multiple IoT applications requiring different levels of throughputs for the QoS flows in the downlink. A single SDN controller is controlling multiple APs inside the network.

In order to implement QoS slicing, we have considered M distinct applications/services in the network with different QoS requirements where $M=[1,2,3,\dots,m]$. To provide QoS to these M distinct applications, S slices are created where $S=[1,2,3,\dots,s]$ and S can be equal to or less than M . Applications requiring same level of throughput or latency are considered similar and belong to the same QoS slice. To prioritize similar applications in the same slice over one another, we have developed a traffic rule abstraction that enables QoS flows to be moved from low priority slice to the high priority slice and vice versa to reliably meet their QoS requirements.

In order to understand the slice throughput requirements, the SDN controller collects statistics like packet arrival rate and

average packet size, for fixed time intervals, from different QoS slices created in the AP. In this way, it can employ time interval based estimator to automatically estimate the slice throughput requirements. Subsequently the controller assigns network resources to the slices to meet these requirements. In case of traffic flows exceeding the available capacity at the APs, the low priority slices are affected thus ensuring QoS for high priority slices. To avoid such situations, admission control algorithms can be employed to ensure that traffic flows does not exceed the available network capacity.

IV. DRL FRAMEWORK FOR QoS SLICE RESOURCE MANAGEMENT

In order to address the challenges of diverse applications QoS, varying channel conditions and changing network dynamics, we have employed Deep Reinforcement Learning (DRL) to develop an adaptive slice resource management scheme. In our proposed scheme, a DRL agent is deployed inside the SDN controller that runs as a network application and takes the resource assignment decisions for the QoS slices. The primary objective of our DRL agent is to manage resource assignments to QoS slices such that slice throughput requirements are met under all conditions.

A. Problem Formulation

Since there is a single DRL agent that is learning optimal slice configuration for multiple slices, this slice resource management problem becomes a Multi-Objective problem where each slice satisfaction is a separate objective for the DRL agent. This is done through weight vector which contains weight of each of the objectives of DRL agent. Therefore, our problem can be represented by Multi Objective Markov Decision Process (MOMDP). The throughput of each slice would be the sum of throughputs of all users present in that slice and the quantum assignment to the slice. So we can represent it as follows:

$$Th_s = \sum_{k_s \in K_s} Th_k = f(Q_s, MCS, TX - Power, ACK) \quad (1)$$

where k represents the number of users belonging to slice s . The throughput of each slice will depend on the Quantum values, MCS, transmit power and ACK policy etc being used by the users in the slice. Therefore, the slice throughput would be a function of many OSI layer parameters in addition to quantum values as represented by the function in equation 1. The total achievable throughput, bounded by the network capacity, is given by:

$$Th_{total} = \sum_{s \in S} Th_s \quad (2)$$

The objective of our DRL agent is to satisfy the throughputs of all the slices in the network while also trying to maximize the aggregate throughput. Our optimization problem can be represented with the following objective function:

$$\max_{s \in S} \sum_{s=1}^S Th_s \quad (3)$$

subjected to,

$$C.1 \quad Th_s > TH_{es} \forall s \in S$$

$$C.2 \quad Q_{total} \leq 10000 \mu sec$$

where TH_{es} represents the estimated slice throughput and Q_{total} represents the total quantum value distributed among the slices. The value is kept at $10000 \mu sec$ to keep scheduling latency of the QoS flows under 10 ms.

B. DRL Framework

To employ DRL in this problem, the wireless environment and QoS requirements are represented with the help of environment state. There are numerous variables that can define the environment state but they eventually have an effect on achieved throughput. Therefore, we have defined the state of our DRL agent with the help of achieved throughput in each of the created slices in the network. For meeting slice's throughputs, our DRL agent assigns quantum values to the slice schedulers and follows Airtime Deficit Round Robin (ADRR) scheduling policy defined by [14] to prioritize the slices. ADRR uses airtime deficit in its scheduling which is calculated based on the MCS being used at the physical layer therefore, MCS indirectly becomes part of the DRL agent's state space. For better understanding of our DRL framework, the states, actions and rewards of our agent are described as follows:

1) *State*: The DRL agent defines its state with the help of throughput being achieved by each slice and their estimated throughput requirements as follows:

$$s_t = (Th_1, TH_{e1}, Th_2, TH_{e2}, \dots, Th_s, TH_{es}) \quad (4)$$

where Th_s represents the throughput achieved by slice s and TH_{es} represents the required throughput thresholds for slice s estimated by the SDN controller.

2) *Action*: The action that DRL agent can take is either to increase the quantum (resource) of the slice, keep the quantum same or decrease the quantum of the slice depending on the state of the agent. The action can be represented as follows:

$$a_t = (Q_{increase}, Q_{decrease}, Q_{neutral}) \quad (5)$$

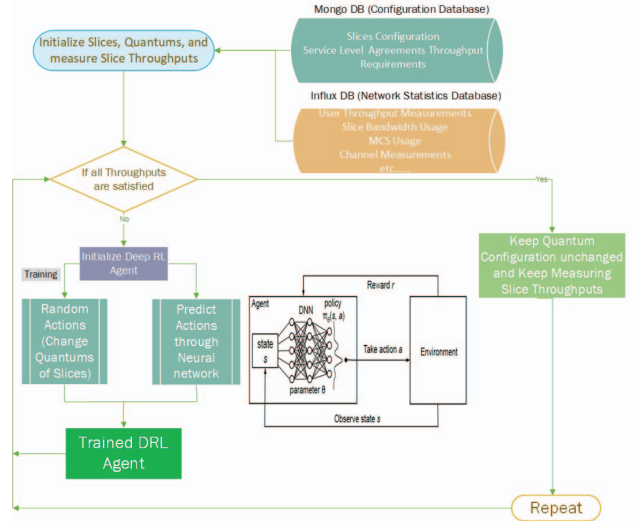


Fig. 2. Flow Chart of DDQN based Slice Resource Management Algorithm

3) *Reward*: The reward function is defined on the basis of throughput satisfaction level in each of the slice, represented as $R_t^s(s_t, a_t)$ and is given by:

$$R_t^s(s_t, a_t) = \begin{cases} 10 & \text{if } R_t^s \leq 110 \% \text{ satisfaction} \\ 7 & \text{if } R_t^s \geq 70 \% \text{ satisfaction} \\ 4 & \text{if } R_t^s \geq 50 \% \text{ satisfaction} \\ 1 & \text{if } R_t^s \geq 30 \% \text{ satisfaction} \\ -5 & \text{otherwise.} \end{cases} \quad (6)$$

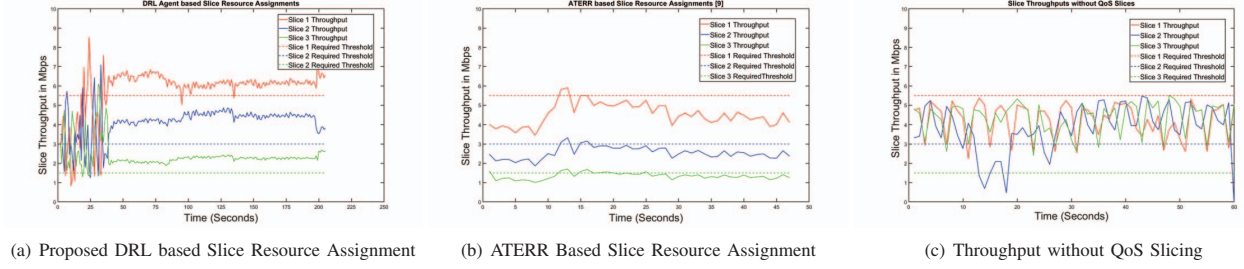


Fig. 3. Comparison of Slice Resource Assignment for QoS in WiFi based IoT Network

Each slice will get its own reward represented as $R_t^1(s_t, a_t)$, $R_t^2(s_t, a_t)$ and $R_t^s(s_t, a_t)$. The total reward that the agent gets at each time is calculated as follows:

$$R_t = \mathbf{W} * [R_t^1, R_t^2, \dots, R_t^s]^T \quad (7)$$

where \mathbf{W} is the weight vector representing the throughput requirements of the slices. The weight vector is calculated based on the available channel capacity that the AP estimates from time to time and slice throughput requirements. The calculation of the weight is done by the DRL using the available channel capacity agent as follows:

$$\mathbf{W} = [W_{s_1}/\zeta, W_{s_2}/\zeta, \dots, W_{s_m}/\zeta] \quad (8)$$

where ζ represents the total AP capacity that it estimates from time to time.

C. DRL Model: Neural Network Architecture

Because of the large number of states, actions and requirement of efficient quantum assignments to the slices, we have used deep neural network as the function approximator for our DRL agent. Such a function approximator learns the function "f" given in equation 1 through environment exploration. We have employed Double Deep Q Network (DDQN) with two hidden layers in our framework. The size of hidden layers are 256 and 128 respectively for both the evaluation and target Q-Network. We have used *Relu* activation function. As an optimizer for NN, Adam optimizer with a learning rate of 0.001 is used and value of discount factor (gamma) was kept at 0.8. To train the Q-Network, a random batch of 64 samples is read from the replay buffer of size 100000 to de-correlate the learning samples and to improve the learning efficiency. The hyper parameters including learning rate, layers sizes, discount factor and batch size were empirically chosen after running multiple experimentations under different QoS requirements and scenarios (their universality is left for further study). The size of the input to neural network is equal to the number of the slices in the network however, neural network can be trained with any number of slices depending on QoS classes in the network. After taking the optimal action (correct quantum assignment to slices) represented by equation 5, the agent gets a feedback from the environment in the form of reward calculated using equation ?? and moves into the new state. During this time step, Q-values of actions are calculated using the Bellman Optimality equation Afterwards, the neural network performs the update while minimizing the loss function. The flow chart of the DDQN algorithm used in our DRL based QoS Slice Resource Management framework is given in Figure 2.

V. PERFORMANCE EVALUATION

In order to evaluate the proposed DRL based QoS slice resource management solution, a 5GEmpower based real world test bed has been used. For WiFi connectivity, IEEE 802.11n standard has been used with 2.4 GHz band in real world environment where other WiFi access points and devices are sharing the channel with our setup to capture real world network effects. Iperf3 has been used to generate TCP and UDP traffic in the downlink to emulate different applications. For application QoS, Network (QoS) slicing was employed where slice requirements can be predefined in the network considering the service level agreements. In our evaluation framework, the slice requirements were estimated using time interval based packet estimator. The time interval of 5 seconds was used to make the throughput estimates. The slice throughput requirements were kept such that all available network capacity is utilized to create a resource constrained environment. The slice requirements were changed in run time by generating different traffic streams with different rates during the training and evaluation of algorithms. For the performance evaluation, we have considered three different slices in our study along with the default slice however, more number of slices can also be considered in the proposed framework depending on QoS classes. The DRL agent is deployed as Network Application in the SDN controller that takes input from the InfluxDB, where 5GEmpower stores its statistics, to observe its state and take suitable action as per learnt policy.

A. Slice Throughput Performance

In order to observe the performance of the DRL agent, we have included threshold requirements of the slices in our framework to measure performance of our resource assignments by the DRL agent. These threshold requirements are estimated in real time to create a dynamic and autonomous QoS delivery. The SDN controller continuously estimates slice throughput requirements and DRL agent learns the environment through exploration and exploitation. The DRL agent explores the environment during the training phase and uses larger quantum value changes ($700 \mu\text{sec}$) to quickly learn the slice requirements and wireless environment. Afterwards, the quantum changes are done in small steps ($200 \mu\text{sec}$) to stabilize the slice throughputs. In this way, the DRL agent is able to meet the slice throughput requirements as shown in Figure 3(a).

The large variations in throughputs in start are due to large quantum changes. Our proposed scheme is tested in real world scenario where other WiFi APs were being operated in the same frequency bands, our system faced interferences from other APs. The DRL agent is able to achieve the desired throughput by continually adapting the resource assignments as per channel conditions. From Figure 3(a), we can see that channel conditions at 95 sec and 130 sec goes bad for the slice 1 and system quickly

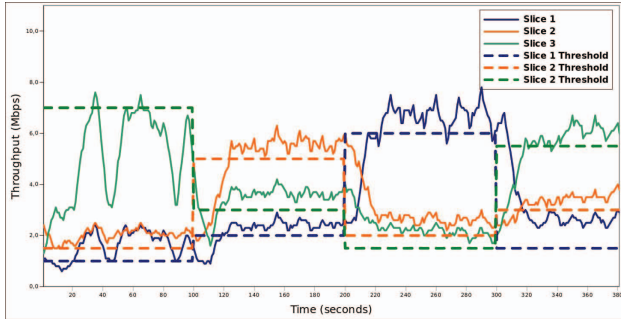


Fig. 4. DRL Agent Adaptation to Changing Slice Requirements

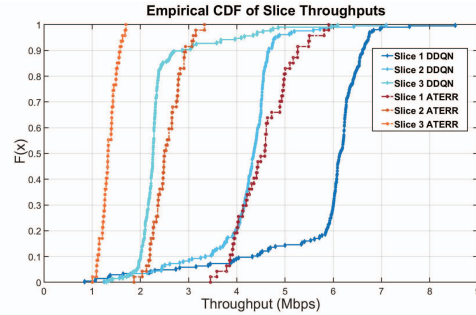


Fig. 5. Comparison of Empirical CDFs of Slice Throughputs

recovers it back to optimal configuration. When the channel conditions worsen such that the desired threshold requirements cannot be met, the AP divides the available capacity proportional to the overall slice requirements. On the contrary, ATERR [8] based resource assignment calculates quantum assignments as per available number of slices without looking at channel conditions and hence cannot always meet desired threshold requirements.

B. Adaptation of DRL Agent to changing Threshold Requirements

The throughput requirements of QoS flows in the network are not fixed and keep on changing with time. Moreover, keeping the throughput assignments fixed for different slices is also not wise as throughput from slice 1 can be assigned to other slices when there are no QoS flows present in slice 1. In order to test the performance of the DRL agent and its capability to adapt to changing slice throughput requirements, varying traffic with different packet sizes was generated in different slices. As our DRL agent was trained based for different slice requirements therefore, it quickly determines optimal slice resource configurations to meet their new requirements as shown in Figure 4. This is because the DRL agent learns the network dynamics and slice throughput requirements and tries to maximize the long term reward. This enables agent to keep past actions and slice requirements in consideration while taking resource allocation decisions and hence it quickly adapts to the changing throughput requirements.

C. Maximizing Aggregate Network Throughput

Optimal quantum assignments to the QoS slices in a dynamic wireless environment, specifically where multiple APs from other network are also operating can improve the aggregate network capacity and can provide higher throughput to the users in the slices. Since our proposed scheme tries to maximize the aggregate network throughput along with satisfying slice throughput requirements, we are able to achieve higher throughputs of slices and aggregate throughput compared to the ATERR scheme as shown in CDFs in Fig 5.

VI. CONCLUSION AND FUTURE DIRECTIONS

To address the shortcomings in present WiFi QoS mechanism and improving QoS satisfaction of IoT applications, we have proposed a Deep Reinforcement Learning based slice resource management algorithm that assigns efficient and dynamic resources to QoS slices for meeting their throughput requirements. The DRL agent adapts to changing slice requirements by employing feedbacks in the form of rewards and is able to meet the dynamic QoS needs in a wireless IoT network. Our framework provides more efficient, flexible and granular control over QoS provisioning in future IoT networks. In this work, we have tested the system with 3 slices only however, it can be tested with more than 3

slices and up to any number of slices. In our future work we plan to target lower layer parameters at the MAC and PHY layers like Contention Window (CW) size, Inter Frame Spacing (IFS), MCS, buffer sizes and transmit power etc to develop better control over slice resources. These parameters can bring change in slice throughputs and packet delays at faster timescales of the order of milliseconds therefore, they can make system more adaptable to quick changes in slice requirements.

REFERENCES

- [1] P. Karwel *et al.*, "Ericsson mobility report," *Ericsson AB, Technol. Emerg. Business, Stockholm, Sweden, Tech. Rep. EAB-21*, 2021.
- [2] W. Li, Y. Chai, F. Khan, S. R. U. Jan, S. Verma, V. G. Menon, X. Li, *et al.*, "A comprehensive survey on machine learning-based big data analytics for iot-enabled smart healthcare system," *Mobile Networks and Applications*, vol. 26, no. 1, pp. 234–252, 2021.
- [3] IEEE, "Ieee 802.11e-2005," *IEEE Standard for Information Technology*, 2005.
- [4] W. Wydmański and S. Szott, "Contention window optimization in iee 802.11ax networks with deep reinforcement learning," in *2021 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, 2021.
- [5] X. Jiang, H. Shokri-Ghadikolaei, G. Fodor, E. Modiano, Z. Pang, M. Zorzi, and C. Fischione, "Low-latency networking: Where latency lurks and how to tame it," *Proceedings of the IEEE*, vol. 107, no. 2, pp. 280–306, 2019.
- [6] S. Szott, K. Kosek-Szott, P. Gawx0142;owicz, J. T. Gx00F3;mez, B. Bellalta, A. Zubow, and F. Dressler, "Wi-fi meets ml: A survey on improving iee 802.11 performance with machine learning," *IEEE Communications Surveys Tutorials*, pp. 1–1, 2022.
- [7] M. Richart, J. Baliosian, J. Serrat, J.-L. Gorricho, and R. Agüero, "Slicing in wifi networks through airtime-based resource allocation," *Journal of Network and Systems Management*, vol. 27, no. 3, pp. 784–814, 2019.
- [8] M. Richart, J. Baliosian, J. Serrat, J.-L. Gorricho, and R. Agüero, "Slicing with guaranteed quality of service in wifi networks," *IEEE Transactions on Network and Service Management*, vol. 17, no. 3, pp. 1822–1837, 2020.
- [9] P. H. Isolani, N. Cardona, C. Donato, J. Marquez-Barja, L. Z. Granville, and S. Latré, "Sdn-based slice orchestration and mac management for qos delivery in iee 802.11 networks," in *2019 Sixth International Conference on Software Defined Systems (SDS)*, pp. 260–265, 2019.
- [10] P. H. Isolani, N. Cardona, C. Donato, G. A. Pérez, J. M. Marquez-Barja, L. Z. Granville, and S. Latré, "Airtime-based resource allocation modeling for network slicing in iee 802.11 rans," *IEEE Communications Letters*, vol. 24, no. 5, pp. 1077–1080, 2020.
- [11] M. Richart, J. Baliosian, J. Serrat, J.-L. Gorricho, R. Agüero, and N. Agoulmine, "Resource allocation for network slicing in wifi access points," in *2017 13th International Conference on Network and Service Management (CNSM)*, pp. 1–4, 2017.
- [12] E. Coronado, S. Bayhan, A. Thomas, and R. Riggio, "Ai-empowered software-defined wlangs," *IEEE Communications Magazine*, vol. 59, no. 3, pp. 54–60, 2021.
- [13] E. Coronado, S. N. Khan, and R. Riggio, "5g-empower: A software-defined networking platform for 5g radio access networks," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 715–728, 2019.
- [14] E. Coronado, R. Riggio, J. Villa1ón, and A. Garrido, "Lasagna: Programming abstractions for end-to-end slicing in software-defined wlangs," in *2018 IEEE 19th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, pp. 14–15, 2018.