QoS-aware Reinforcement Learning for Multimedia Traffic Scheduling in Home Area Networks

Sabrine Aroua, Giacomo Quadrio, Yacine Ghamri-Doudane, Ombretta Gaggi, Claudio Enrico Palazzi

QoS-aware Reinforcement Learning for Multimedia Traffic Scheduling in Home Area Networks

Sabrine Aroua[1], Giacomo Quadrio[2], Yacine Ghamri-Doudane[1], Ombretta Gaggi[2], and Claudio Enrico Palazzi[2]
[1]La Rochelle University, Laboratory of Informatics, Image and Interaction (L3I), France
[2]University of Padua, Department of Mathematics "Tullio Levi-Civita", Italy
{sabrine.aroua,yacine.ghamri}@univ-lr.fr {gquadrio,gaggi,cpalazzi}@math.unipd.it

***Abstract*—Cloud-based interactive multimedia applications such as virtual games and video streaming are gaining high popularity. However, giving the high bandwidth consumption, the remote execution can negatively impact the quality of the multimedia traffic. In such a realm, data travel different communication networks from the cloud to the final users crossing the last meters the home's access point (AP). In such a scenario, the quality-of-service (QoS) support is a challenging task, particularly in the home network environment, with heterogeneous applications simultaneously running and consuming the available bandwidth. To address this issue, we propose ReiLeCS, a Reinforcement Learning-based Controller and Scheduler for interactive multimedia traffic in Home Area Networks (HAN). Through reinforcement learning and the maximization of a reward function, it enables the AP to schedule the arriving multimedia traffic from the cloud according to their required QoS. Simulation results using real multimedia traffic conditions demonstrate that ReiLeCS achieves better performances compared with existing packet scheduling policies.**

***Index Terms*—interactive multimedia, QoS, Reinforcement Learning (RL), wireless**

I. Introduction

Cloud technologies represent a game changer in how we enjoy multimedia contents [1]. The possibility to transfer the storage and the computational power to remote servers allows users to have access to services and experiences that, until a few years ago, would have required dedicated multimedia supports (e.g., DVDs) or dedicated hardware (e.g, game consoles). Instead, to enjoy them, tiny clients or any other device already present at home suffice.

The plethora of services that can be offered through the cloud is vast, but the most challenging are the interactive multimedia services (e.g., cloud gaming), which have to guarantee a reactive interaction with the users. For this reason, this new category of services requires both high amount of bandwidth and low end-to-end latency [2], [3]. This is not a simple task but some solutions are already being studied.

Figure 1 depicts a typical scenario for the remote fruition of multimedia contents. Relaying on remote execution, the data is usually generated and transmitted from the cloud-server to an access point (AP) of a home area network (HAN) which sends the data to the user's device. The data are transmitted from the cloud-server to the AP through data connections, e.g., fast optical connections or even mobile 4G or 5G connections. However, when the data reach the HAN, smart home appliances and multiple users could be connected to the AP and intensively using the network to enjoy either high quality or interactive multimedia services [4]. A scenario that is more and more frequent, especially during these days where the COVID-19 pandemic drastically transformed the global network usage [5], [6]. Therefore, the HAN available bandwidth could become crowded and the provision of the required QoS for multimedia applications challenging.

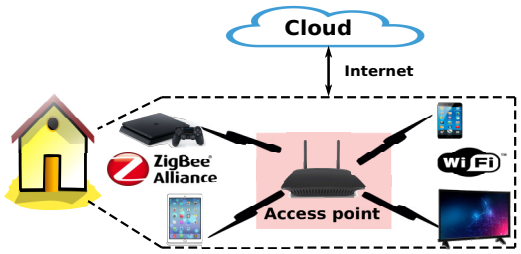

Fig. 1. Architecture for remote multimedia run.

Therefore the AP must smartly deliver the multimedia traffic to the final users in order to guarantee a proper quality of experience (QoE), taking ito account the QoS requirements of each application and the available network resources. In this paper, we propose the **Rei**nforcement **Le**arning-based **C**ontroller and **S**cheduler (ReiLeCS) for interactive multimedia traffic in HANs. ReiLeCS enables the AP to adapt its multimedia traffic control and scheduling to the available bandwidth in the HAN in such a way to deliver the traffic flows according to their required QoS. We identify the QoS of the different services in terms of data-rate, latency and reliability.

We exploit the heterogeneous QoS requirements of the multimedia traffic and we classify them into multiple classes of traffic. Then, we formulate the scheduling problem as a Markov decision-based framework to which we associate a carefully designed reward function by considering the QoS requirements of every traffic class. ReiLeCS deploys Reinforcement Learning (RL) techniques to enable the AP to learn the policy that best and adequately delivers the traffic purely through experience [7]. Our goal is to achieve a stable and balanced traffic scheduling with QoS support for a long time horizon by maximizing the expected cumulative discounted reward. Simulation results demonstrate that ReiLeCS consid-

This work is partially funded by the Department of Mathematics of the University of Padua through the BIRD191227 project.

erably improves the system performance.

The paper is organized as follows. Section II overviews the related works. In Section III, we present the system model, then we describe our proposed solution, ReiLeCS, in Section IV. In Section V we provide and discuss our evaluation results. Finally, conclusions are drawn in Section VI.

II. Related Work

As mentioned, the QoS and QoE requirements to properly enjoy interactive multimedia and high quality video streaming are not easy to reach; researchers have hence already focused their attention on this problem.

For instance, in Bujari *et al.* [8], the authors chose to deploy a TCP-Vegas like algorithm on the top of the home gateway in order to make different network flows coexist. The designed algorithm exploits the type of protocol that the different network flows uses to prioritize some packets. In Corbillon *et al.* [9] instead, the authors use a different approach based on statistics. Basically, it exploits the peculiarities of 360-degree virtual reality videos to design a statistical model able to reduce the bandwidth consumption and, at the same time, to maintain the video quality. The model performs a prediction of the user's head movements and transmits only the packets related to the portions of the 360 video that is most plausible that will be displayed. To prioritize packets only on the basis of the service's protocol could be a weak strategy in the presence of multiple interactive services running on the HAN. However, even if we are not considering 360 videos, the idea to exploit the specific peculiarities of the considered services is a good starting point to formulate our solution.

Among the various techniques under consideration, one of the most promising regards acting directly on the scheduling algorithm that manages the packet transmission. In [10] the authors try to manage multiple video streaming services in the same AP through the use of a scheduling policy called Earliest Positive-Debt Deadline First (EPDF). In this policy, the AP schedules first the packet with the earliest deadline from those whose associated clients possess strictly positive truncated time debts. A similar approach comes from [11] where an algorithm called Delay-Prioritized Scheduling (DPS) is proposed for the management of real time traffic in 3GPP LTE systems. This strategy aims to maximize the throughput while satisfying the QoS requirements of the real time user's applications. To do that, the algorithm uses the instantaneous downlink signal-to-noise ratio values and the packet delay information of each user. These solutions cannot be directly applied to our scenario; yet, we agree that the focusing on the packet scheduling algorithm is an effective approach.

The scheduling algorithms can be enhanced with the aid of RL techniques. The authors of [12] use RL to design a cellular network scheduler that dynamically adapts to the traffic variation and to different reward functions to optimally schedule Internet of Things traffic. In [13] RL is used to schedule the packets on the basis of specific QoS parameters of diverse smart grid applications that operate in cognitive radio sensor networks. The objective is to mitigate problems such as electromagnetic interference, equipment noise or obstructions. The objective of our research is clearly different; yet, the adopted QoS parameters are well suited for the interactive multimedia and the problem that we address.

The aforementioned approaches are not adequate for a dynamic scenario with multiple cloud-based interactive multimedia services running in the same HAN; nevertheless, the use of RL has been proven as a good decision maker in dynamic environments. This leads us to opt for a scheduling algorithm based on RL to take into account the specific QoS characteristics of the running services. The final objective is to help the AP to manage heterogeneous network flows and to guarantee the related QoS requirements.

III. System Model

A. System Overview

We consider the HAN scenario where different users take advantage of the cloud computing technology and enjoy multiple multimedia applications (services). The users' devices are connected to the Internet through a local AP. They exploit the wireless HAN, e.g., Wi-Fi, to communicate with the AP which ensures the two way communication between the users' devices and the cloud-server. From one side, it transmits the users' computational requests to the cloud. Then, from the other side, it sends back the cloud responses to the final devices. In this contribution, we focus on the communication from the AP to the final users (downlink) as this has the highest requirements in terms of bandwidth resources. Our aim is to achieve an efficient data packet scheduling on the downlink. In Table I, we overview the most important multimedia applications, their packet inter-arrival times (λ) and their QoS specications attributed in terms of data rate (β), delay (τ), and packet-error-rate (α). As shown in the table, the considered applications have heterogeneous QoS requirements. Thus, to fulfill our objective, we exploit this aspect and classify the received data packets into multiple classes of services.

In the rest of this section, we present the AP model for the traffic control into multiple classes of services. Then, we describe the mathematical formulation of our problem.

B. Access Point Model for Multiple Classes of Traffic

We consider m the number of interactive multimedia applications that the users enjoy at home. The AP classifies the data flows that arrive from the cloud into m classes according to the QoS requirements of each application. As

TABLE I
Multimedia applications and their QoS specifications.

Service	β (Mbps)	τ (ms)	α (%)	λ (ms)
Cloud Gaming (1080p)	10 to 15 (DWN) 0.5 to 1 (UP)	≤ 160	1	0.92
4K Video Streaming	20 to 25 (DWN)	Preload is possible	1	0.76
Online Gaming	1.5 to 3 (DWN) 0.5 to 1 (UP)	≤ 160	5	17.06

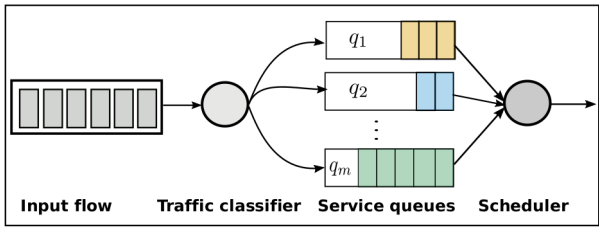

Fig. 2. Queuing model for multimedia traffic in the AP.

depicted in Figure 2, the AP uses the set $\mathcal{Q} = \{q_1, \ldots, q_m\}$ of m queues to store the different data flows before being scheduled to their final receiver. When a data packet of class i arrives to the AP, it is stored in the queue q_i. Then, the AP proceeds to its scheduling and transmission. We characterize the QoS attributes $(\beta_i, \tau_i, \alpha_i)$ of every class of traffic i by adequately chosen minimum and maximum threshold values $\{(\beta_i^{min}, \beta_i^{max}), (\tau_i^{min}, \tau_i^{max}), (\alpha_i^{min}, \alpha_i^{max})\}$. The threshold interval of each attribute defines its tolerance in a class i in terms of data rate, latency and packet-error-rate.

To achieve an efficient scheduling of the data packets with the same QoS requirements, the AP schedules the packets stored in the same queue q_i, according to the First Come First Served (FCFS) policy. We assume that time is divided into continuous scheduling periods (SP). During every SP, the AP aims at exploiting the available bandwidth and schedules the multiple classes of the data packets according to their QoS requirements. Therefore, at the beginning of every SP, the AP makes multiple consecutive scheduling decisions to transmit or not the first packet q_i^1 from the head of every queue q_i ($i \in \{1, \ldots, m\}$). Then, as far as the AP senses available bandwidth, it continues to make new decisions. In this context, we denote by n the scheduling decision number n that AP makes during time and by b_n the available bandwidth at the time this scheduling decision was taken.

Based on the described model, we present in the following the mathematical formulation of our scheduling problem.

C. Mathematical Formulation

Throughout the different scheduling decisions, the AP has to maximize the utility to send multimedia packets in the HAN subject to the data rate, latency and reliability constraints of the different classes of traffic. Therefore, we formulate the problem as:

$$\text{Maximize} \quad \sum_{n=1}^{\infty} \sum_{i=1}^{m} u_{n,i}(\beta_i^1, \tau_i^1, \alpha_i^1) \tag{1a}$$

$$\text{subject to} \quad \beta_i^1 \leq b_n, \tag{1b}$$

$$\tau_i^1 \leq \tau_i^{max}, \tag{1c}$$

$$\alpha_i^1 < \alpha_i^{max}, \ \forall i \in \{1, \ldots, m\}. \tag{1d}$$

The objective function (1a) that the AP wants to maximize will be presented and designed in Section IV-B. Constraint (1b) ensures that during the scheduling decision n, a packet q_i^1 is admitted only when its data rate requirement can be satised through the available bandwidth b_n while the previously scheduled and transmitted packets continue to be served. Constraint (1c) makes sure that q_i^1 can be delivered within the given ow delay. Similarly, constraint (1d) ensures that the packet error rate of q_i^1 is under α_i^{max}.

Formulation (1) presents a sequential decision making problem. Considering the dynamic arrival of packets, the variable bandwidth and the heterogeneous QoS requirements of the different traffic classes, the problem is difficult to solve. Thus, we opt for the use of a reinforcement learning (RL) approach. In essence, using such an approach, the AP learns the optimal policy via successive interactions with the environment and updates its knowledge through the reward feedback.

Our RL-based scheduler is presented in the next section.

IV. Our Reinforcement Learning Approach for Multimedia Traffic Scheduling

Reinforcement Learning (RL) is a set of techniques that allows an agent to take actions and interact with an environment so as to maximize a total reward. RL is always paired with a Markov Decision Process (MDP). Thus, in this paper, we need to first transform our packet scheduling problem into a MDP. Then, we can design our RL approach.

In this section, we first design the MDP and the reward function associated to our problem. Then, we present the learning mechanisms that we use in our solution.

A. Markov Decision Process Model Design

A typical MDP consists of the set (*agent*, *state*, *action*, *reward*) as shown in Figure 3.

In our approach, we present the MDP as:

- *Agent*: an agent is an entity that performs the learning task. In our Reinforcement Learning-based Controller and Scheduler (ReiLeCS), the smart home's AP is responsible for scheduling the sets of packets in the different queues and transmitting them over the available bandwidth.
- *State*: a state of the system is the information that the agent can obtain when observing its surrounding environment. Before making the scheduling decision n, we define the system's state s_n as:

$$s_n = (b_n, s_{n,1}, s_{n,2}, \ldots, s_{n,m}) \tag{2}$$

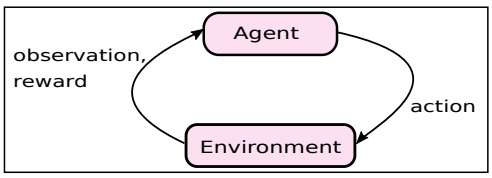

Fig. 3. MDP model.

s_n is composed of the available bandwidth (b_n) and the set $\{s_{n,1}, \ldots, s_{n,m}\}$ where $s_{n,i}$ is the state of the queue q_i ($i \in \{1, \ldots, m\}$). We design $s_{n,i}$ as:

$$s_{n,i} = \left(\beta_i^1, \frac{z_i^1}{\tau_i^{max}}, \frac{d_{n,i}}{\alpha_i^{max}}\right) \quad (3)$$

In Formulation (3), β_i^1 presents the data rate of q_i^1. If q_i is empty then β_i^1 equals 0. Instead, z_i^1 presents the waiting time of the packet q_i^1 in q_i. Therefore, the quotient $\frac{z_i^1}{\tau_i^{max}}$ measures whether the waiting time of q_i^1 is below the maximum allowed delay τ_i^{max}. Similarly, the third element evaluates whether the AP exceeds the maximum threshold α_i^{max} in terms of packet error rate and $d_{n,i}$ presents the number of packets that have been dropped before making the scheduling decision n.

- *Action*: an action a_n indicates how the agent responds to the observed state s_n. In our system, an action a_n is the set $(a_{n,1}, \ldots, a_{n,m})$ where $a_{n,i}$ presents the AP scheduling decisions, to transmit the packet q_i^1 or to keep it waiting till the next decision making.
- *Reward*: a reward represents an evaluation of the decision made by the agent. In our system, the AP uses the reward to evaluate the long-term performance of its scheduling decision. The design of the reward function is discussed in more details in the next section.

B. Reward Function

After every scheduling decision, the AP assesses the performance of the action a_n on the state s_n. It uses the reward function $u_n(a_n, s_n)$ that transforms the QoS performance of the multiple scheduled classes of traffic into a utility value. Specifically, $u_n(a_n, s_n)$ can be expressed as:

$$u_n(a_n, s_n) = \sum_{i=1}^{m} u_{n,i}(\beta_i^1, \tau_i^1, \alpha_i^1) \quad (4)$$

It is the sum of $u_{n,i}$ ($i \in \{1, \ldots, m\}$) where $u_{n,i}$ measures the impact of the action $a_{n,i}$ on the state $s_{n,i}$ of the queue q_i. We design $u_{n,i}$ as:

$$u_{n,i}(\beta_i^1, \tau_i^1, \alpha_i^1) = u_{n,i}^{\text{send}}(\beta_i^1) - u_{n,i}^{\text{dropped}}(\alpha_i^1) - u_{n,i}^{\text{stored}}(\tau_i^1) \quad (5)$$

$u_{n,i}^{\text{send}}$, $u_{n,i}^{\text{dropped}}$ and $u_{n,i}^{\text{stored}}$ are expressed in the equations (6), (7) and (8), respectively. More in detail, $u_{n,i}^{\text{send}}$ expresses the bandwidth used to transmit packets from the queue q_i. Instead, $u_{n,i}^{\text{dropped}}$ presents an evaluation of the number of packets that have been dropped. In this study, a packet is considered to be lost if it is not scheduled and transmitted before its expiration time. The third value, $u_{n,i}^{\text{stored}}$, provides an evaluation of the scheduling decision on the delay of the head-of-queue packet q_i^1. It is involved in the utility function $u_{n,i}$ only if the AP decides to not transmit q_i^1, i.e., $a_{n,i} = 0$. As the AP keeps q_i^1 waiting for the following scheduling decisions, both its waiting time and $u_{n,i}^{\text{stored}}$ increase.

$$u_{n,i}^{\text{sent}}(\beta_i^1) = \beta_i^1 a_{n,i} \quad (6)$$

$$u_{n,i}^{\text{dropped}}(\alpha_i^1) = \frac{d_i(t)}{\alpha_i} \quad (7)$$

$$u_{n,i}^{\text{stored}}(\tau_i^1) = \frac{z_i^1}{\tau_i^{\max}}(1 - a_{n,i}) \quad (8)$$

During a scheduling period, the AP aims at maximizing u_n. Its goal is to maximize the bandwidth usage, minimize the waiting time of the different stored data packets in the m queues and minimize the number of dropped data packets. By using RL, the objective is to consider the different decisions made sequentially over time. As a consequence, our goal is to maximize the expected cumulative discounted reward r:

$$r = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t u_n(a_n, s_n)] \quad (9)$$

where $\gamma \in (0, 1]$ is a factor discounting future rewards.

C. ReiLeCS

In this section, we present the Reinforcement Learning (RL) solution for our multimedia packet scheduling.

One of the dominant RL approach is the value function approach. For each state or state-action pair the value function, i.e., reward, is calculated. Thus, the agent takes advantages of its training period to discover possible state-action pairs and save their associated value functions in a table. As a consequence, the value function approach has several limitations, especially with models that require a large state space; it is limited to tasks with small numbers of states and actions.

In our case, during the scheduling decisions, a very large number of the encountered states may have never been experienced before (during the training phase). In fact, new users may join the network and use different multimedia services. Also, the available bandwidth can change due to interference or the deployment of new smart home appliances such as smart meters or smart surveillance camera. For these reasons, we use the linear gradient descent to get a compact and generalized representation of an estimated reward function [7]. We express the reward function estimator as follows:

$$\widehat{f}(s_n, \theta) = \theta^T \phi(s_n) = \theta^T \nu \quad (10)$$

In Equation (10), ν is the set of features of the state s_n. The set of parameters θ presents the key of $\widehat{f}(s_n, \theta)$ since this function faithfully approximates the reward function u_n. Thus, the AP exploits the training phase and updates the weights θ in a way to minimize the error between u_n and $\widehat{f}$. The set θ is updated as:

$$\theta = \theta + \alpha(u_n - \widehat{f}(s_n, \theta))\nu \quad (11)$$

where α is the learning rate ($\alpha \in [0, 1]$).
The approximator $\widehat{f}(s_n, \theta)$ is differentiable. The derivative of $\widehat{f}(s_n, \theta)$ with respect to θ is ν.

$$\frac{\partial \widehat{f}(s_n, \theta)}{\partial \theta} = \nu \quad (12)$$

In Algorithm 1, we present in details the ReiLeCS training algorithm. During every episode, i.e. a number (Max) of sequential decisions, the AP discovers new states s_n and at the same time updates the weight θ. The objective here is to enable the AP to make the optimal packet scheduling decision. Therefore, the AP needs to not only focuses on the immediate rewards but also the cumulative rewards on the long term. Thus, it has to avoid local optimal rewards and explore new actions that may yield higher rewards in the future. To this end, the AP uses the ϵ-Greedy algorithm to balance the exploitation-exploration trade-off (line9 - line13).

Algorithm 1 ReiLeCS training algorithm.

1: Input: learning rate α, exploration probability ϵ
2: Initialization: Randomly initialize θ
3: **for** each episode **do**
4: $n = 1$
5: **for** $n \in [1, \ldots, Max]$ **do**
6: **while** the bandwidth is available **do**
7: Observe the state s_n
8: Choose a random probability p (ϵ-greedy)
9: **if** $p < \epsilon$ **then**
10: randomly select an action a_n
11: **else**
12: $a_n = \arg\underset{a_n}{\text{Max}} \widehat{f}(s_n, \theta)$
13: **end if**
14: Calculate u_n
15: $\theta = \theta + \alpha(u_n - \widehat{f}(s_n, \theta))\nu$
16: $n = n + 1$
17: **end while**
18: **end for**
19: **end for**

V. Performance Evaluation

In this section, we present some numerical evidences to demonstrate the effectiveness of ReiLeCS. We first consider the scenario of a HAN where different users are enjoying the multimedia applications reported in Table I ($m = 3$). Then, in order to emulate realistic network traffic, we recorded real traffic measurement from Wireshark, the open-source packet analyzer, to generate the data sets that we have used to train and test our solution. Every data set is characterized by a number of users (from 1 to 5) per multimedia application.

First, we investigate the performance of the training algorithm, i.e., Algorithm 1. When the learning-rate α equals 0.01, we measure the convergence of the mean squared error (mse) between the reward u_n and the estimated reward function $\widehat{f}$. Figure 4 shows the variation of mse according to the episode number. As can be observed, at the beginning of the training process, the achievable mse fluctuates widely. Then, after a sufficient number of episodes, this fluctuation becomes negligible and the mse converges to its optimal value. This is due to the fact that the AP continuously updates the weights θ based on the feedback error it receives ($u_n - \widehat{f}$). As a result, after a sufficient number of episodes, the AP acquires a good knowledge of the reward u_n.

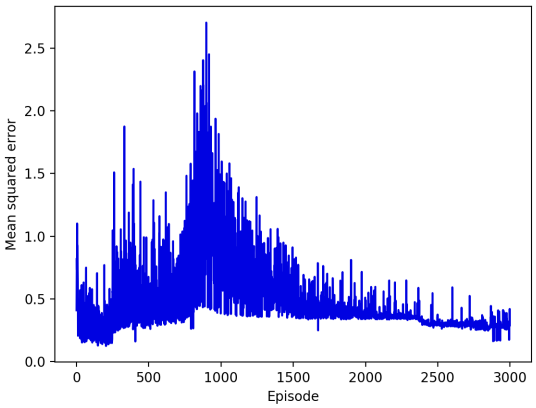

Fig. 4. Convergence of the mean squared error of ReiLeCS.

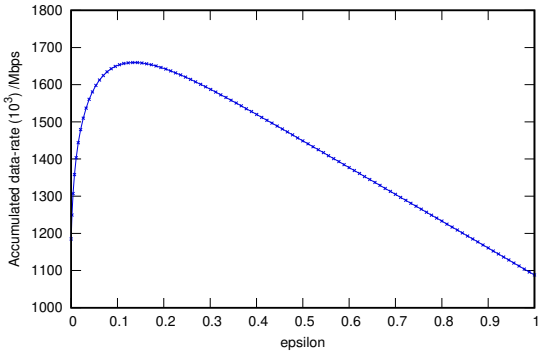

Fig. 5. Exploitation-exploration trade-off.

In Figure 5, we focus on the exploitation-exploration trade-off. Thus, when the available network bandwidth equals 1300 Mbps and varying the values of ϵ, we measure the accumulated exploited data-rate. The chart shows that exploring new actions and states that do not result in the best reward can improve future scheduling decisions. For our approach the best usage of the bandwidth is achieved when ϵ equals 0.01. Then, as we increase ϵ, i.e., explore more random action and states, the performance degrades.

In Figure 6, we depict the variation of the accumulated data-rate with the total available bandwidth and we compare the performance of ReiLeCS with the greedy scheduling policy and the random policy. When using the greedy scheduling policy, the AP focuses on maximizing the reward during the current scheduling decision n without considering its impact on future rewards. With the random policy, while the AP senses available bandwidth, it chooses the packet to transmit randomly, without taking into account any of the QoS requirements of the considered multimedia applications. As a result, the chart shows that our approach outperforms the greedy and the random policies. The greedy policy maximizes

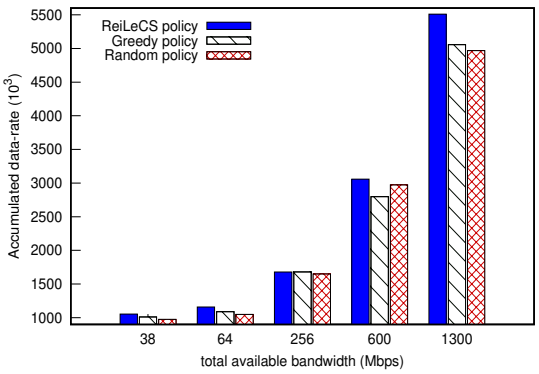

Fig. 6. Bandwidth usage performance.

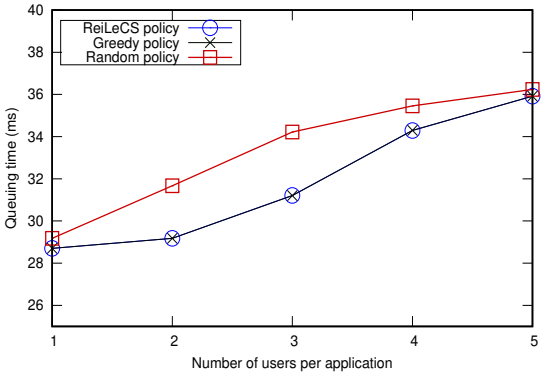

Fig. 7. Queuing time performance.

the instantaneous reward u_n and does not explore new actions that may provide better scheduling decisions in the future. The random policy takes the decision even more blindly. ReiLeCS takes advantage of the ϵ-greedy algorithm; it explores new states and actions to avoid local optima with the hope of finding a global optimum.

Finally, in Figure 7 we evaluate the capacity of ReiLeCS to satisfy the delay requirements of the considered applications. Since cloud gaming possesses the most challenging combination of small inter-arrival time λ_i and stringent delay requirements (previously shown in Table I), we focus on the mean queuing time of the cloud gaming packets while varying the data sets (i.e., the number of users enjoying the applications). As reported in the chart, the measured delay increases with the number of users per application since the channel becomes more and more crowded. ReiLeCS and the greedy policy achieves an almost identical waiting time even if the former was able to exploit more bandwidth as shown in Figure 6. They both consider the QoS delay requirement during their scheduling decisions; thereby, they prioritize the scheduling of packet with the highest delay requirements. As foreseeable, for the various data sets, ReiLeCS outperforms the random policy since the latter completely ignores the QoS requirements. We can hence conclude that the estimated reward function $\widehat{f}(s_n, \theta)$ succeeds in considering the heterogeneous QoS requirements of the applications and prioritizing the packet flows with with the most stringent requirements.

VI. Conclusion

In this paper, we proposed ReiLeCS, the Reinforcement Learning-based Controller and Scheduler for interactive multimedia traffic in HANs. In the context of cloud based multimedia content delivery, ReiLeCS enables a smart home's AP to schedule the cloud-generated multimedia packets according to their QoS requirements. To this aim, before being scheduled, the data packets are classified into multiple classes of services. Then, RL is employed to allow the AP to take proper scheduling decisions in order to maximize a reward function we designed to consider the QoS requirements. ReiLeCS aims at maximizing the expected cumulative reward. It exploits the ϵ-greedy algorithm to avoid local optima and improve future scheduling decisions. We evaluated ReiLeCS using real multimedia traffic conditions. The simulation results show that ReiLeCS is effectively trained to optimize multimedia traffic scheduling. Moreover, it demonstrates its capacity to consider the variant QoS requirements and to prioritize the scheduling of traffic with the most stringent QoS requirements.

References

[1] P. Lal and S. S. Bharadwaj, "Understanding the impact of cloud-based services adoption on organizational flexibility: An exploratory study," *J. of Enterprise Information Management*, vol. 29, no. 4, Jul 2016.

[2] G. Quadrio, A. Bujari, C. E. Palazzi, D. Ronzani, D. Maggiorini, and L. A. Ripamonti, "Network analysis of the steam in-home streaming game system," in *Proc. of the 14th Annual IEEE Consumer Communications and Networking Conference*, Jan 2017.

[3] M. Claypool, D. Finkel, A. Grant, and M. Solano, "Thin to win? Network performance analysis of the OnLive thin client game system," in *Prof. of NetGames 2012*, Jan 2012.

[4] A. Sabrine, E. K. Inès, G.-D. Yacine, and S. Leila, "A distributed cooperative spectrum resource allocation in smart home cognitive wireless sensor networks," in *IEEE Symposium on Computers and Communications (ISCC)*, Feb 2017.

[5] BBC, "Netflix to cut streaming quality in Europe for 30 days," https://www.bbc.com/news/technology-51968302, Apr 2020.

[6] Nokia, "Network traffic insights in the time of COVID-19: April 9 update," https://www.nokia.com/blog/network-traffic-insights-time-covid-19-april-9-update/, Apr 2020.

[7] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," in *Reinforcement Learning: An Introduction*, T. M. Press, Ed., 2015.

[8] A. Bujari, M. Massaro, and C. E. Palazzi, "Vegas over access point: Making room for thin client game systems in a wireless home," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 12, Dec 2015.

[9] X. Corbillon, A. Devlic, G. Simon, and J. Chakareski, "Optimal set of 360-degree videos for viewport-adaptive streaming," in *Proc. of the 25th ACM International Conference on Multimedia*, Oct 2017.

[10] I.-H. Hou and R. Singh, "Scheduling of access points for multiple live video streams," in *Proc. of MobiHoc'13*, Jul 2013.

[11] K. Sandrasegaran, H. A. Mohd Ramli, and R. Basukala, "Delay-prioritized scheduling for real time traffic in 3GPP LTE system," in *Proc. of IEEE WCNC 2010*, Apr 2010.

[12] S. Chinchali, P. Hu, T. Chu, M. Sharma, M. Bansal, R. Misra, M. Pavone, and S. Katti, "Cellular network traffic scheduling with deep reinforcement learning," in *The Thirty-Second AAAI Conference on Artificial Intelligence*, Feb 2018.

[13] G. A. Shah, V. C. Gungor, and O. B. Akan, "A cross-layer QoS-aware communication framework in cognitive radio sensor networks for smart grid applications," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 3, pp. 1477–1485, Aug 2013.