Delay-aware Slicing and MAC Management using MCDA in IEEE 802.11 SD-RANs

Pedro H. Isolani*, Daniel J. Kulenkamp†, Johan M. Marquez Barja‡,
Lisandro Z. Granville§, Steven Latré*, and Violet R. Syrotiuk†

*Department of Computer Science, University of Antwerp - imec, Antwerp, Belgium
{pedro.isolani, steven.latre}@uantwerpen.be

†School of CIDSE, Arizona State University, Tempe, USA
{dkulenka, syrotiuk}@asu.edu

‡Department of Electronics - ICT, University of Antwerp - imec, Antwerp, Belgium
johann.marquez-barja@uantwerpen.be

§Institute of Informatics, Federal University of Rio Grande do Sul, Porto Alegre, Brazil
granville@inf.ufrgs.br

***Abstract*—Advanced wireless services and applications are demanding lower latency and better reliability. In IEEE 802.11 networks, slicing abstractions at the Medium Access Control (MAC) layer can provide precise resource allocation and traffic isolation as a means to meet these performance requirements. In this paper, we propose a delay-aware approach for network slicing and MAC management using Multi-Criteria Decision Analysis (MCDA) in IEEE 802.11 Software-Defined Radio Access Networks (SD-RANs). To enable such an approach, we monitor queueing delay statistics at the Software-Defined Networking (SDN) controllers and the Access Points (APs). We evaluate our approach in a testbed with two APs, controlled by Ryu and 5G-EmPOWER controllers, with four Stations (STAs) served by both Quality of Service (QoS) and Best Effort (BE) flows dynamically. We compare our approach to a state-of-the-art user association approach. Our results show that in addition to load balancing flows across APs, our approach enhances the QoS delivery at runtime using slicing.**

***Index Terms*—Software-Defined Networking, Network Slicing, MAC Management, IEEE 802.11 RAN**

I. Introduction

Today's advanced wireless services and applications are increasingly latency-sensitive [1]. Given the dynamic nature of the wireless environment, on-the-fly Medium Access Control (MAC) management is essential for efficient and reliable wireless communication [2]. To support traffic prioritization, IEEE 802.11e introduced the Enhanced Distributed Channel Access (EDCA) function [3]. EDCA defines access categories as priority levels, which determine how clients access the channel. Access categories with higher priority have more transmission opportunities than others and, once its clients reach the contention period, more time is reserved for transmission. However, EDCA does not ensure radio resource isolation among such access categories. Further, because EDCA is a distributed channel access method with no standard management interfaces, reliable connectivity cannot be guaranteed.

In contrast, network slices operate independently from one another and provide precise networking resources and traffic isolation among users and services [4]. The fact that slices can be dynamically instantiated, modified, and terminated implies a strong decoupling between software-based functions and the underlying network infrastructure. The 3GPP fifth generation of mobile networks (5G) architecture is embracing the Control/User Plane Split (CUPS) as one of the fundamental enablers for network programmability and End-to-End (E2E) network slicing [1]. In this context, Software-Defined Networking (SDN) has facilitated network innovation and simplified network management. Consequently, SDN-tailored systems are envisioned to ease the creation of logical and isolated networks via slice abstractions [5].

Current research efforts address network slicing and SDN to enhance resource utilization in IEEE 802.11 Radio Access Networks (RANs) [6]–[13]. There is extensive work on user association algorithms that focus on load balancing, mobility support, and fairness [14]–[16]. Such approaches often benefit from SDN to address resource management. Despite this work, deciding how to efficiently allocate, control, and manage users and slice resources remains challenging [17]. As well, latency-related constraints have not been considered.

In this paper, we propose a delay-aware approach for network slicing and MAC management using Multi-Criteria Decision Analysis (MCDA) in IEEE 802.11 Software-Defined Radio Access Networks (SD-RANs). We argue that, by combining the flexibility of network slicing and MAC management, load balancing of flows and enhanced Quality of Service (QoS) delivery can be addressed. To enable such an approach requires the monitoring of the queueing delay at both SDN controllers and Access Points (APs). We evaluate our approach in a testbed with two APs, controlled by Ryu and 5G-EmPOWER controllers, and four Stations (STAs) served by both QoS and Best Effort (BE) flows. Further, we compare our approach to a recent user association algorithm [16]. Our results show improved load balancing of flows across APs and QoS guarantees using slicing.

The paper is organized as follows. After a review of related work in §II, §III overviews our system architecture. §IV provides the algorithm based on MCDA for load balancing and an algorithm for airtime scheduling of slices. A description of our testbed, new monitoring capabilities, set-up, and results of experimentation are found in §V. Finally, in §VI we summarize and describe future work.

II. Related Work

Ensuring QoS in wireless networks is a longstanding research challenge that has only become more complex [18]. After the IEEE 802.11e amendment [3] established the foundations for traffic prioritization, many investigations started to focus on queue management [19]–[22]. Later, with the improvements provided by the IEEE 802.11n amendment [23], the focus shifted towards channel optimization and fairness [24]–[28]. Any solution requiring modifications to the driver (e.g., frame formats) becomes non-standard compliant.

Network slicing addresses precise infrastructure sharing, allowing reliable and improved QoS delivery [6]. Most consist of airtime-based Resource Allocation (RA) mechanisms for IEEE 802.11 network virtualization [7] [29] [30]. Airtime scheduling has been extensively studied as a means to overcome the well-known *IEEE 802.11 Performance Anomaly* [31]. Without such slicing capabilities, STAs equally share the available radio resources only if they experience the same or similar channel conditions. Otherwise, when an STA uses a lower bit rate, it results in performance degradation perceived by all.

Recent proposals [7]–[13] address network slicing in IEEE 802.11 networks. Richart et al. [7] proposed a resource allocation mechanism to achieve infrastructure sharing and slicing on Wi-Fi APs. The mechanism assigns a specific portion of airtime to each slice as a resource to be shared. However, the proposal was only assessed via simulation. Others [8]–[11] focused on practical implementations. Høiland-Jørgensen et al. [8] proposed a queueing scheme to provide airtime fairness and latency reduction under load. Aleixandri, Betzler, and Camps-Mur [9] proposed a scheduling algorithm to allocate airtime to a set of Wi-Fi interfaces while the authors in [10] proposed a policy enforcement mechanism to provide airtime fairness in WiFi networks. However, runtime slice orchestration based on latency requirements was not addressed.

Coronado et al. [11] proposed a framework that enables programmable and dynamic E2E network slicing over heterogeneous RANs. Deployed on a real-world testbed, slices and client traffic isolation have been evaluated through achieved throughput. However, given today's stringent latency-related requirements, slice resource allocation requires further research. Vassilaras et al. [17] state that future wireless networks have to consider E2E latency requirements for a service chain where feasible slice embedding must also satisfy and guarantee the QoS requirements. Yet deciding how to efficiently control and manage such resources at runtime remains challenging.

On another front, user association algorithms focus on load balancing, mobility support, and fairness. They often take advantage of the SDN centralized view to control and manage the network [14]–[16]. By gathering the Received Signal Strength Indicator (RSSI) and throughput measurements, seamless handovers are triggered, improving the overall throughput of the network. However, latency-related metrics are not considered within their decision-making process. Today's latency-sensitive applications motivate the need to consider delay within queues.

In previous work [12] [13], we evaluated the impact of runtime slice reconfiguration on the E2E latency using ICMP. Subsequently, we integrated the queueing delay measurements into the formulation of a QoS optimization problem. This work takes queueing delay measurements at both controllers and APs, and uses the results of the monitoring process in network slicing, MAC management, and seamless handovers, to enhance the SD-RAN resource utilization and QoS delivered.

III. System Overview

We consider a scenario where network slices are orchestrated by a logically centralized entity in an SDN-enabled network infrastructure. This entity has a global view of and control over all network resources. Multiple *tenants* (*i.e.*, virtual operators or service providers) share the infrastructure and have their specific Service Level Agreements (SLAs). These SLAs are translated into QoS requirements for the network to support. To meet such requirements, we propose the use of network slicing.

According to Richart et. al. [7], there are two variants of network slicing. The first abstracts the different services and ensures QoS within them, called Quality of Service Slicing (QoSS). The second defines slices as the traditional idea of network virtualization where a precise subset of network resources is allocated to each *tenant* and full control is provided, called Infrastructure Sharing Slicing (ISS). In our work, we use the QoSS definition. Figure 1 illustrates the architecture as a layered system model.

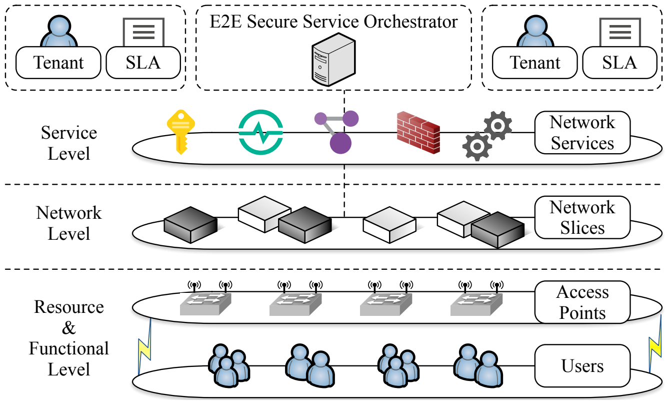


Fig. 1. Overall IEEE 802.11 architecture as a layered system model.

The IEEE 802.11 RAN consists of a set APs responsible for delivering data from services down to users in the network, *i.e.*, STAs. Each AP has resources to be shared and therefore to be managed properly. To represent the minimum chunk of wireless resources that can be assigned to a user, we use the

Resource Block abstraction [32]. A resource block defines a Wi-Fi interface at a given AP, identified by the network interface identifier (i.e., MAC address), operating channel (e.g., 1, 6, 11), and the type of channel (e.g., High Throughput (HT) 20MHz, Very High Throughput (VHT) 40MHz). In our system, each AP has only one interface and therefore only one resource block; henceforth, resource blocks are synonymous with APs. Figure 2 shows a simplified queue structure along with the data traffic flow within an AP.

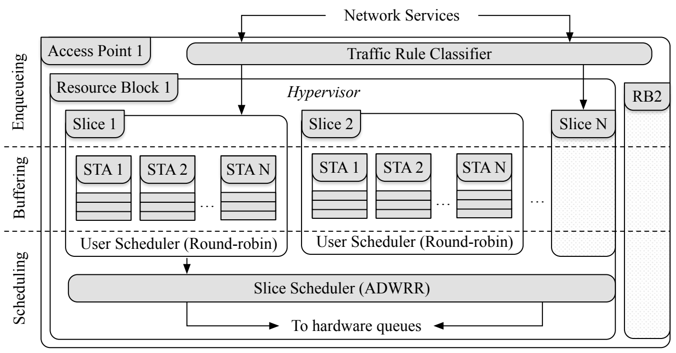


Fig. 2. Simplified slice queue structure and data traffic flow in an AP.

For each AP, frames are classified into queues as slices based on the definition of traffic rules (e.g., OpenFlow rules). Frames in such slices/queues are dequeued following the Airtime Deficit Weighted Round Robin (ADWRR) scheduling scheme [11]. In ADWRR, a portion of airtime, or *quantum* Q^s, is allocated to each slice s in each transmission round. When a larger value for Q^s is assigned to a slice s more radio resources are allocated to s; this provides a mechanism to support services with stricter performance requirements. Nevertheless, it is important to notice that inactive traffic rules do not cause any performance degradation to the system.

IV. Delay-aware SDN-based approach

MCDA is a sub-discipline of operations research that explicitly evaluates multiple conflicting criteria in decision making. We apply MCDA whenever we want to decide to which AP a flow is assigned according to the high-level objectives of balancing the load among APs while satisfying the delay constraints of QoS flows. Our user association algorithm is integrated into our delay-aware approach for network slicing and MAC management in IEEE 802.11 SD-RANs. We describe the details of our approach next.

A. Formulating the Load Balancing Problem using MCDA

The RAN consists of a set B of APs, responsible for delivering n services to a set M of mobile STAs. Each service is instantiated in a slice, with S^b denoting the slices of AP b. Each STA t is therefore served by a subset $\alpha \subseteq S^b$ of slices.

MCDA evaluates multiple conflicting criteria in decision making. We select six criteria for MCDA to evaluate for an AP b: (*i*) the overall channel load θ^b; (*ii*) the total measured dequeueing rate $\mu^b = \sum_{s \in S^b} \mu^s$; (*iii*) the total expected load $\mu^b_{\text{EXP}} = \sum_{s \in S^b} \sum_{t \in t^b_\alpha} \mu^{s,t}_{\text{EXP}}$, where $\mu^{s,t}_{\text{EXP}}$ is the expected load for STA t in slice s given by its dequeueing rate; (*iv*) the total measured queueing delay $D^b = \sum_{s \in S^b} D^s$, where D^s is the queueing delay within slice s; (*v*) t^b_{RSSI}, the RSSI perceived at b from STAs within range; and (*vi*) an indicator variable t^b_α which evaluates to true if STA t is associated with AP b. The first four criteria are minimized to avoid resource overuse. We use (*iv*) to avoid APs and with a high number of active or overflowing queues. This reduces the chance of a Network Interface Card (NIC) overload and channel saturation. The last two criteria are maximized to improve the chances of using higher data rates, and of fewer connection disruptions.

The weight of each criterion depends on the flow type, either QoS or BE. We use the Analytic Hierarchy Process (AHP) [33] to inform our selection of weights for each flow type, and then tune the resulting weights to avoid the ping-pong effect. Another consideration is that we want BE flows to be more likely to undergo handovers than QoS ones because handovers are detrimental to delay. The MCDA criteria and the resulting weights by flow type ($\mathcal{W}_{\text{BE}}$ and $\mathcal{W}_{\text{QoS}}$) are listed in Table I.

TABLE I
MCDA criteria, objectives, and weights for AP b

Criterion	Objective	$\mathcal{W}_{\text{BE}}$	$\mathcal{W}_{\text{QoS}}$	Description
θ^b	MIN	0.10	0.10	Overall channel load of b.
μ^b	MIN	0.15	0.10	Measured dequeueing rate of b.
μ^b_{EXP}	MIN	0.40	0.20	Overall expected load of b.
D^b	MIN	0.15	0.20	Measured queueing delay of b.
t^b_{RSSI}	MAX	0.10	0.20	Measured RSSI from STA t of b.
t^b_α	MAX	0.10	0.20	True if STA t is associated with b.

Several guidelines for choosing the appropriate method to solve an MCDA problem are given in [34]. Given that our problem has quantitative weights, a quantitative scale of comparisons, no uncertainty in the decision problem, and the decision problem is characterized by a complete ranking, we select the Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) [35]. TOPSIS ranks the alternative solutions by minimizing the distance to the positive ideal solution and the farthest geometric distance from the negative ideal solution.

B. Monitoring Queueing Delay

To implement our MCDA approach, several extensions were required on the APs and also at the controllers. To obtain the queueing delay statistics, we extended the agent implementation (running on APs) with a frame-tracking functionality. This enabled calculation of the average time spent by each slice on an AP. Extensions both to the *OpenEmpower* protocol (used for the controller and AP communication) and to the Ryu controller made use of the new statistics. This required several handler apps to be implemented to maintain and calculate the needed metrics. The metrics are reported as a Simple Moving Average (SMA) of the last ten measurements. Finally, two management apps implement our user association and network slicing algorithms.

Algorithm 1 User Association Algorithm

```
Input:
 1: $every$                                  ▷ configuration loop interval (20 sec used)
 2: $\mathcal{C}, \mathcal{W}_{\text{QoS}}, \mathcal{W}_{\text{BE}}$            ▷ set of MCDA criteria and weights
 3: $\forall s \in S^b : D^s_{\text{QoS}}$       ▷ max-avg queueing delay of each slice $s$
 4: $\forall s \in S^b : \mu^{s,t}_{\text{EXP}}$       ▷ expected dequeueing rate of each STA $t$

 5: loop $every$
 6:     for each $b \in B$ do                          ▷ iterate over all APs
 7:         $b_{\text{STATS}} \leftarrow$ GETRBSTATS($b$)
 8:         $\mu^b_{\text{EXP}} =$ GETRBEXPECTEDLOAD($b$)
 9:     for each $t \in M$ do                          ▷ iterate over all STAs
10:         $\mu^t_{\text{EXP}} \leftarrow$ GETSTAEXPECTEDLOAD($t$)
11:         $t_{\text{WEIGHTS}} \leftarrow$ GETSTAWEIGHTS($t$)
12:         for each $b \in B$ do
13:             if $t^b_\alpha = true$ then $\mu^b_{\text{EXP}} = \mu^b_{\text{EXP}} - \mu^t_{\text{EXP}}$
14:             TOPSIS.ALTERNATIVE($\mathcal{C}, t_{\text{WEIGHTS}}, \mu^b_{\text{EXP}}, b_{\text{STATS}}$)
15:         $b_{\text{BEST}} \leftarrow$ TOPSIS.BESTALTERNATIVE()
16:         if $t^{b_{\text{BEST}}}_\alpha \neq true$ then
17:             DOHANDOVER($t, b_{\text{BEST}}$)
18:
19: function GETSTAWEIGHTS($t$)
20:     for each $s \in S^b$ do              ▷ iterate over all slices of an AP
21:         if $D^s_{\text{QoS}}$ then return $\mathcal{W}_{\text{QoS}}$
22:     return $\mathcal{W}_{\text{BE}}$
23:
24: function GETSTAEXPECTEDLOAD($t$)
25:     $\mu^t_{\text{EXP}} \leftarrow 0$
26:     for each $s \in S^b$ do              ▷ iterate over all slices of an AP
27:         $\mu^t_{\text{EXP}} = \mu^t_{\text{EXP}} + \mu^{s,t}_{\text{EXP}}$
28:     return $\mu^t_{\text{EXP}}$
29:
30: function GETRBEXPECTEDLOAD($b$)
31:     $\mu^b_{\text{EXP}} \leftarrow 0$
32:     for each $t \in M$ do                          ▷ iterate over all STAs
33:         if $t \in \text{t}^b_\alpha$ then
34:             $\mu^b_{\text{EXP}} = \mu^b_{\text{EXP}} +$ GETSTAEXPECTEDLOAD($t$)
35:     return $\mu^b_{\text{EXP}}$
```

C. Using MCDA in the User Association Algorithm

Our user association algorithm is given in Algorithm 1. At a high level, this algorithm solves the load balancing problem formulated in §IV-A. To do such a thing, this algorithm periodically decides to which AP STAs should be assigned using an MCDA method and then perform the handovers accordingly. To make the handover decisions, the algorithm first gathers the monitored statistics from all APs (line 7). The expected load of each AP depends on how STAs are distributed and their flow demands (line 8). The expected load of t is calculated based on its active flows and corresponding types. Once obtained, the handover decision for t can be made.

To avoid the ping-pong effect caused by the expected load, we subtract the expected load of an STA t from the overall expected load of the AP with which it is connected (line 13). This prevents the expected load of t from affecting its own handover decisions. Next, TOPSIS solves the MCDA problem returning b_{BEST}, the highest-ranked AP according to the criteria (line 15). An STA undergoes a handover only if is not associated with its top-ranked AP b_{BEST} (line 17).

Algorithm 2 Network Slicing Algorithm

```
Input:
 1: $every$                          ▷ configuration loop interval (5 sec used)
 2: $\forall s \in S^b : D^s_{\text{QoS}}$       ▷ max-avg queueing delay of each slice $s$
 3: $Q^s_{\text{MIN}}, Q^s_{\text{MAX}}$       ▷ min, max quantum (10 us, 12 000 us used)
 4: $Q^s_{\text{INC}}, Q^s_{\text{DEC}}$     ▷ increase, decrease factors (10 %, 30 % used)

 5: loop $every$
 6:     for each $b \in B$ do                          ▷ iterate over all APs
 7:         RECONFIGURE($b$, REQUIREMENTSMET($b$))
 8:
 9: function REQUIREMENTSMET($b$)
10:     for each $s \in S^b$ do          ▷ iterate over all slices of an AP
11:         if $D^s_{\text{QoS}}$ then
12:             if $D^s_{\text{QoS}} > D^s$ then return $Q^s_{\text{DEC}}$
13:     return $Q^s_{\text{INC}}$
14:
15: function RECONFIGURE($b, Q^s_{\text{FACTOR}}$)
16:     for each $s \in S^b$ do          ▷ iterate over all slices of an AP
17:         if $D^s_{\text{QoS}} == \emptyset$ then
18:             $Q^s_{\text{CRR}} \leftarrow$ GETCURRENTQUANTUM($s$)
19:             $Q^s_{\text{NEW}} \leftarrow Q^s_{\text{CURR}} \times Q^s_{\text{FACTOR}}$
20:             if $Q^s_{\text{NEW}} > Q^s_{\text{MAX}}$ then $Q^s_{\text{NEW}} \leftarrow Q^s_{\text{MAX}}$
21:             if $Q^s_{\text{NEW}} < Q^s_{\text{MIN}}$ then $Q^s_{\text{NEW}} \leftarrow Q^s_{\text{MIN}}$
22:             if $Q^s_{\text{NEW}} \neq Q^s_{\text{CURR}}$ then $b$.SETSLICE($Q^s_{\text{NEW}}$)
```

D. Network Slicing Algorithm

Algorithm 2 is responsible for adapting the network slice configurations at runtime. Based on the max-avg queueing delay threshold and the quantum adjustments, the network slicing algorithm aims to satisfy the QoS requirements of the QoS flows by reallocating resources from the BE slices to the QoS slices. Periodically, this algorithm checks, for each AP, whether the requirements of all QoS slices are met. When all requirements of an AP are met, the quantum is increased by a factor of Q^s_{INC} (line 13), releasing resources until all slices share the AP equally. Otherwise, the quantum is decremented by a factor of Q^s_{DEC} (line 12), leaving more resources for the QoS-constrained slices. Q^s_{MIN} and Q^s_{MAX} are thresholds that prevent traffic in BE slices from being blocked and from exceeding a maximum Q^s configuration, respectively. A new quantum Q^s_{NEW} is set for a slice on an AP only when it differs from its current one.

It is important to emphasize that, according to the ADWRR scheduling scheme, inactive traffic rules, i.e., slices, do not cause any performance degradation to the system. The limit of resources in which a slice might utilize only proceeds when the AP is saturated and the remaining resources must be allocated to other active slices.

V. Performance Evaluation

A. Methodology and Workload

Figure 3 shows the real testbed used to evaluate our approach. It is made up of a single computer hosting the SDN controllers, two APs, and four STAs. The APs are based on the PC Engines APU2D4 (x64) processing board, equipped with one Qualcomm Atheros AR958x 802.11 a/b/g/n each. The STAs are Raspberry Pis 4 Model B+ with 802.11b/g/n/ac.

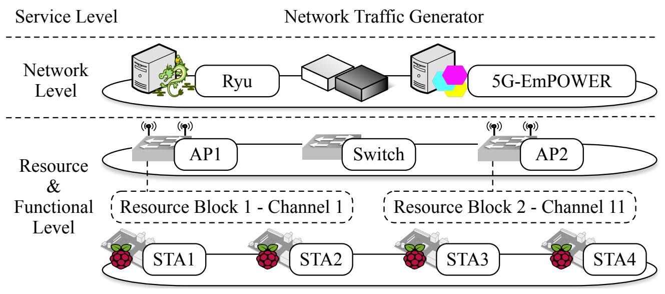

Service Level
Network Traffic Generator
Network Level
Ryu
5G-EmPOWER
Resource & Functional Level
AP1
Switch
AP2
Resource Block 1 - Channel 1
Resource Block 2 - Channel 11
STA1
STA2
STA3
STA4

Fig. 3. Testbed deployment scenario.

In our scenario, APs and STAs are positioned about 2 meters apart from one another. Therefore, we have set the APs to operate on non-overlapping channels, specifically on channels 1 and 11. The supported Modulation and Coding Scheme (MCS) rate indices are from 0 to 7 because the STAs operate in the $2.4\,\mathrm{GHz}$ band. Our experiments were conducted in a closed office environment with little to no external interference.

To perform network slicing, our approach relies on the 5G-EmPOWER platform[1], which includes an SD-RAN controller, a backhaul controller implementation of the SDN controller Ryu[2], and an agent that runs at each AP. To demonstrate our approach, we generate five User Datagram Protocol (UDP) flows with each flow representing a different service in the network. For each flow, a dedicated slice with the default quantum Q^s of $12\,000\,\mathrm{us}$ is instantiated. Downstream traffic is generated from the SDN controller towards the four STAs, with workload parameters given in Table II. To avoid static flow rates and arrival times, we generated the flows following the Poisson distribution with MGEN[3] having a fixed frame size of $1024\,\mathrm{bytes}$. The experiments were run for five minutes.

TABLE II
WORKLOAD PARAMETERS USED DURING EXPERIMENTATION

Event	Time (sec)	Flow	STA	$\mu^{s,t}_{\mathbf{EXP}}$	$D^s_{\mathbf{QoS}}$
1	10	QoS 1	1	2 Mbps	30 ms
2	40	BE 1	2	4 Mbps	NA
3	70	BE 2	2	4 Mbps	NA
4	100	BE 3	3	4 Mbps	NA
5	130	BE 1	2	0 Mbps	NA
		BE 2	2	0 Mbps	NA
6	160	BE 1	2	15 Mbps	NA
		BE 2	2	15 Mbps	NA
		BE 3	3	15 Mbps	NA
7	190	QoS 2	4	2 Mbps	50 ms

During an experiment, the SDN controller periodically polls for monitoring statistics on both APs. The average and the median of the last measurement window is calculated. The median is only used to avoid the masking effect in the presence of outliers. The polling frequency and window size are configurable but are set to $1\,\mathrm{sec}$ and 10 measurements, respectively. Handovers and slice re-configurations are triggered periodically, according to Algorithms 1 and 2, with configurable intervals set to $20\,\mathrm{sec}$ and $5\,\mathrm{sec}$, respectively.

[1]https://github.com/5g-empower/5g-empower.github.io
[2]https://osrg.github.io/ryu/
[3]https://www.nrl.navy.mil/itd/ncs/products/mgen

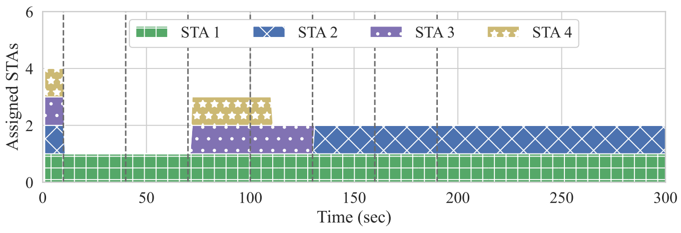
B. *Results and Analysis*

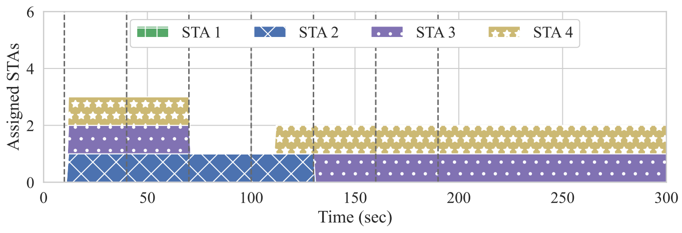
Figure 4 illustrates the STA association status as well as the throughput and queueing delay per slice for a representative experiment. Vertical dashed lines mark the events (see Table II). Figures 4a and 4b show to which AP each STA is assigned during the experiment and, therefore, when handovers occurred. Initially, because there are no active flows in the network, only the RSSI and channel load measurements distinguish the ideal AP, so all STAs were assigned to AP 1 (Figure 4a). After the first flow starts (at second 10), the overall expected load, measured load, and queueing delay of the AP increase. This causes other STAs (which are not receiving data) to be handed over to AP 2 (Figure 4b). At event 3, AP 2 has more resources being used (two $4\,\mathrm{Mbps}$ flows versus one $2\,\mathrm{Mbps}$ flow on AP 1). This causes STAs 3 and 4 to be reassigned to AP 1. Figures 4c and 4d present the dequeueing rate per slice.

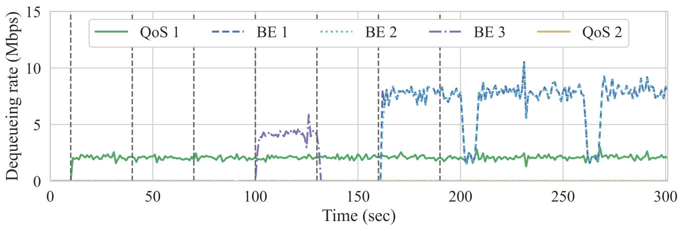
At event 5, when BE flows 1 and 2 are stopped, STA 3 using slice BE 3 is reassigned while it had an active flow. In this case, because the APs are operating on different channels, the Channel Switch Announcement (CSA) mechanism is triggered. CSA is defined by the IEEE 802.11h amendment to enable APs to announce switching to a new channel before their transmission begins on that channel. Beacon messages containing the CSA information are sent to the STA before it switches to the new channel. This allows STAs, which support CSA, to transit to the new channel with minimal downtime. Indeed, the flow that was reassigned was running on a BE slice while the first flow, running on a QoS slice (QoS 1), had no connection disruptions. Figures 4e and 4f present the queueing delay per slice.

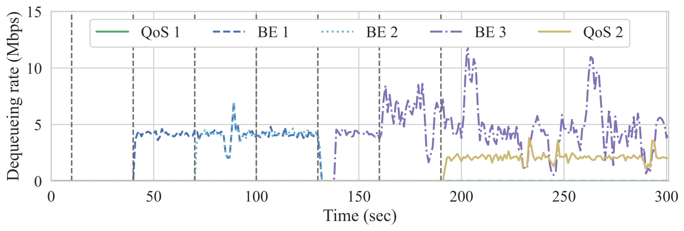
At event 6, the three BE flows increase their throughput to $15\,\mathrm{Mbps}$. This event was introduced to determine if QoS can be ensured with higher demands in the network. When this event occurs, the queueing delay for slice QoS 1 was not satisfied for a short period (i.e., above the $30\,\mathrm{ms}$ requirement) because the BE slices are using as much as possible from their airtime available. This increased the delay of other slices on the same AP, therefore, slice adaptations were required.

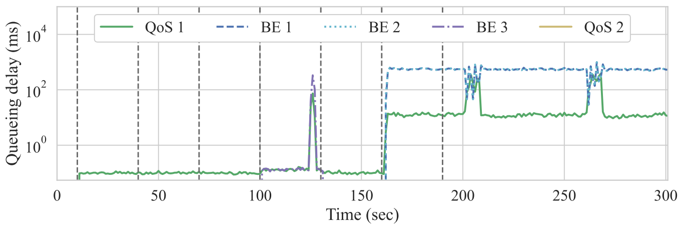
Figure 5 shows the quantum values for all slices during the experiment. The Q^s of all BE slices on the AP were reduced according to Algorithm 2 which released resources to the QoS slice. Specifically, in the second 211, the Q^s values for the two BE slices dropped by $30\,\%$, allowing the delay requirements of the QoS slice to be satisfied. Later, resources were released to the BE gradually, with Q^s values increased by a $10\,\%$ factor until it reaches the maximum threshold. At event 7, a new QoS flow is introduced. Since there are 3 active flows assigned to AP 1 and its overall queueing delay is currently the highest, the AP selected for this new flow is AP 2. As Figure 4f shows, the new QoS flow is introduced at second 190 and its queueing delay of $50\,\mathrm{ms}$ is guaranteed for most of the time with analogous slice re-configurations on AP 2.


(a) Assignment status of STAs with AP 1.


(b) Assignment status of STAs with AP 2.


(c) Dequeueing rate of AP 1.


(d) Dequeueing rate of AP 2.


(e) Queueing delay (log scale) of AP 1.

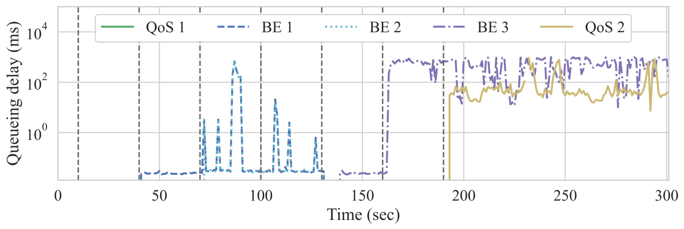

(f) Queueing delay (log scale) of AP 2.

Fig. 4. STA assignment status, dequeueing rate, and queueing delay per slice over the experiment timespan running our proposed approach.

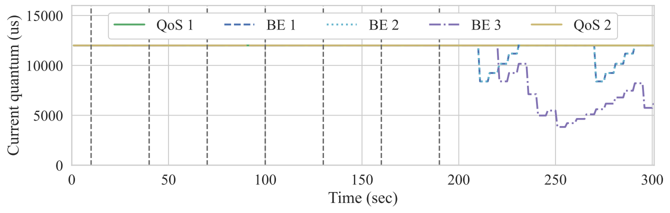


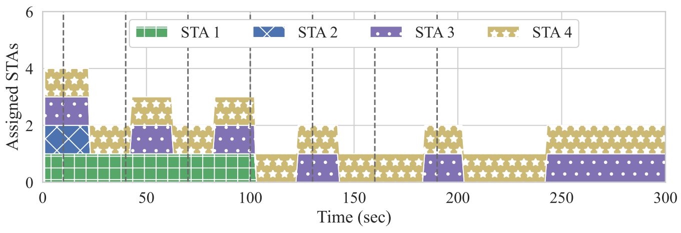
Fig. 5. Current quantum of the slices over the experiment timespan.

Overall, during this experiment, the channel load was insignificant, being on average $0.5\,\mathrm{Kbps}$ with a standard deviation of $0.2\,\mathrm{Kbps}$ on each of channels 1 and 11. During their active periods, QoS flow 1 and 2 had averages very close to the expected dequeueing rate ($2\,\mathrm{Mbps}$) while BE 1, 2, and 3, from event 6, had around $7\,\mathrm{Mbps}$ each. In this case, more than half of the frames are waiting within the queues until the NIC is ready and it is their turn to transmit. Therefore, such BE flows experience queueing delays of almost $1\,\mathrm{sec}$ as presented in Figures 4e and 4f.

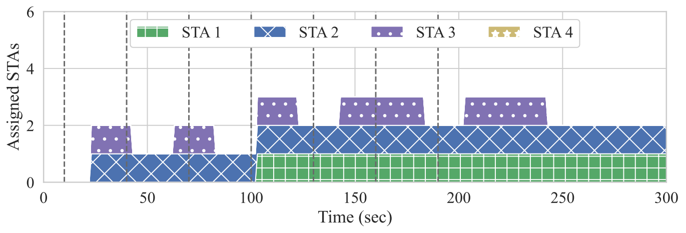
C. Comparative Study

We now compare our approach to a state-of-the-art approach for user association [16]. Gómez et al. proposed an algorithm that improves the aggregated goodput compared to traditional approaches based on signal strength. To achieve such results, their proposal focuses on three indicators to trigger the user re-association process: (*i*) average RSSI of an AP, (*ii*) AP load, and (*iii*) channel occupancy. The first indicator (*i*) refers to the average of the uplink RSSIs for all STAs connected to the APs. The second (*ii*) represents the actual load of the APs in the network, while the third (*iii*) represents the load of the channels in which APs are operating.

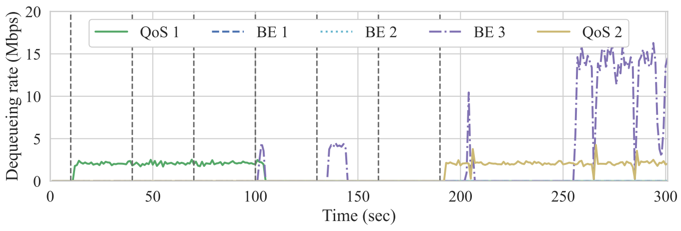
Gómez et al. [16] claim that the ping-pong effect can be reduced and fairer resource utilization can be ensured by using the maximum $\max_i$, minimum $\min_i$, and median η_i values for an indicator i. When $\max_i - \min_i > \eta_i$, the best AP is determined and handovers might be performed. To determine the best AP to connect, For all STAs, the proposed algorithm computes the maximum value for the average RSSIs, perceived at each AP, times the load of the AP and channel. However, the ping-pong effect might occur in scenarios where APs perceive similar RSSIs. For example, assuming two APs perceiving similar RSSIs from STAs and there is an uneven load on APs or channel occupancy. In this case, at each reconfiguration, STAs will be reassigned to the AP with a lower load or the one operating in the less busy channel. Consequently, AP loads and the channel occupancy will be swapped and the same behavior will be observed in the next reconfiguration, causing STAs to ping-pong.



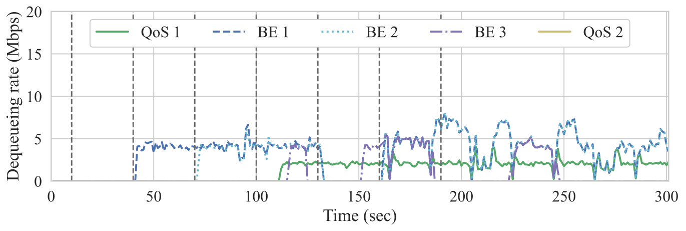
(a) Assignment status of STAs with AP 1.



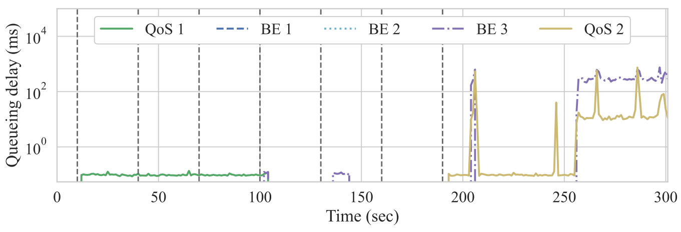
(b) Assignment status of STAs with on AP 2.



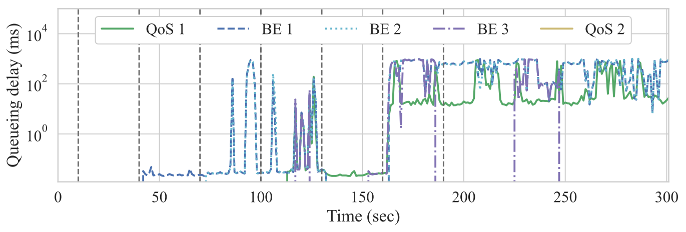
(c) Dequeueing rate of AP 1.



(d) Dequeueing rate of AP 2.



(e) Queueing delay (log scale) of AP 1.



(f) Queueing delay (log scale) of AP 2.

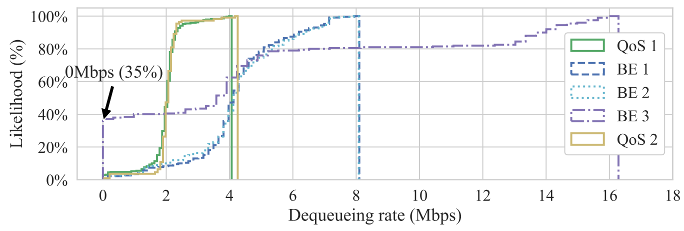
Fig. 6. STA assignment status, dequeueing rate, and queueing delay per slice over the experiment timespan running the approach from Gómez et al. [16].

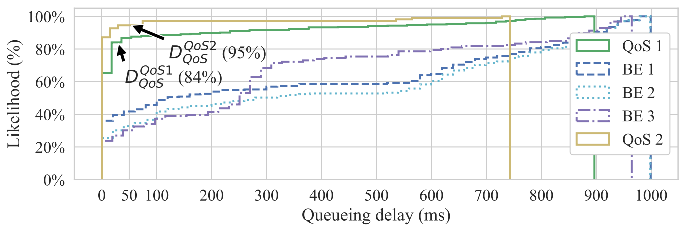
In Gómez et al. [16], the user association algorithm verifies the indicators sequentially, and decisions are based on only one indicator at a time. We evaluate their algorithm in a $5\,\mathrm{min}$ experiment, using the same configuration loop interval as our user association algorithm (Algorithm 1) and the same workload parameters presented in Table II.

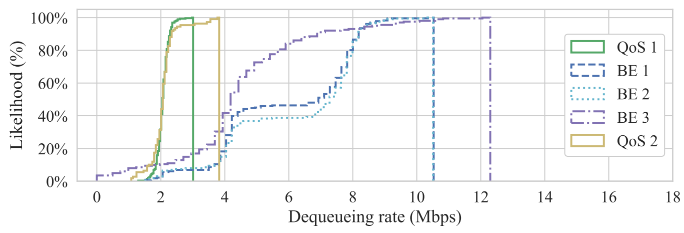
Figure 6 illustrates the STA association status, dequeueing rate, and queueing delay per slice during a $5\,\mathrm{min}$ experiment. As before, the vertical dashed lines mark the events (see Table II). Figures 6a and 6b show to which AP each STA is assigned during the experiment. As can be seen, the number of handovers triggered during this experiment is higher, especially for STA 3. The total number was 12 while only 8 where performed using our approach. Besides, with our approach, only one handover occurred with an STA that had an active flow and the flow was a BE flow. On the other hand, using the algorithm presented in [16], a connection disruption happened for the STA running a QoS flow. At the second 100, STA 1 was moved from AP 1 to AP 2 while the QoS 1 flow was active. Because the APs are operating on different channels, the STA has to switch its channel and, due to this overhead, no data was received for about $8\,\mathrm{seconds}$.

As expected, the ping-ping effect might occur for STAs perceived by multiple APs with similar RSSI values. In this experiment, STA 3 has suffered 9 handovers where 6 happened when BE 3 flow was active. The impact of the handovers as well as the expected dequeueing rate and delay requirements for QoS flows were not considered. Our proposal, on the other hand, considers weighted criteria so that STAs running BE flows are more likely to suffer handovers while STA running QoS flows tend to remain connected to the same AP. Figure 7 shows the Cumulative Distribution Function (CDF) of both dequeueing rate and queueing delay per slice running the flows of both approaches. As we can see, due to the excessive handovers performed by STA 3 running [16], BE 3 flow has not received data for $35\,\%$ of the experiment duration while it is no more than $5\,\%$ of the time with our approach.

Figures 7b and 7d present the likelihood of the queueing delay of each slice using the CDF. The plot shows that QoS 1 flow, which was active for most of the experiment time, had its delay below its requirement D_{QoS}^{QoS1} with the probability of $95\,\%$ while the probability was $84\,\%$ with the other approach. This is due to the network slicing algorithm that interactively adapts a slice's airtime according to the delay requirements. For the QoS 2 flow, our approach could only maintain its delay lower than D_{QoS}^{QoS2} with a $76\,\%$ probability compared to a probability of $95\,\%$ using the approach by Gómez et al. [16]. However, with closer examination, we can see that at around the second 100 mark, the BE 3 flow has stopped on AP 1 and this has favored QoS 2 flow that was running along with it.


(a) Dequeueing rate per slice with the approach from Gomez et al. [16].


(b) Queueing delay per slice with the approach from Gomez et al. [16]


(c) Dequeueing rate per slice with our proposed approach.

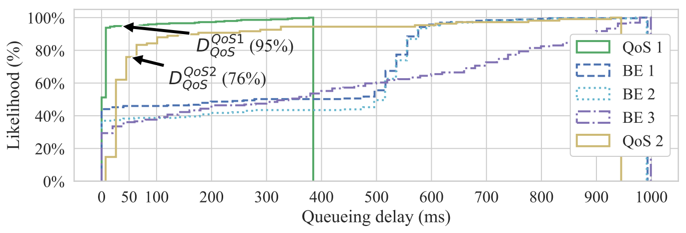
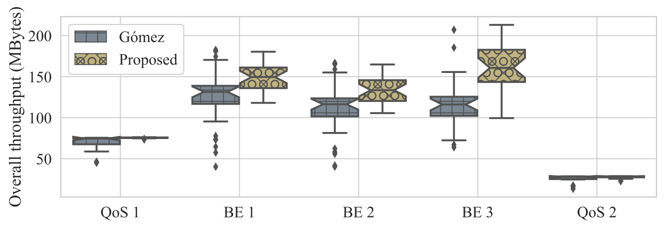

(d) Queueing delay per slice with our proposed approach.

Fig. 7. CDF for the dequeueing rate and queueing delay for the approach from Gómez et al. [16] and for our approach.


(a) Overall throughput of slices.

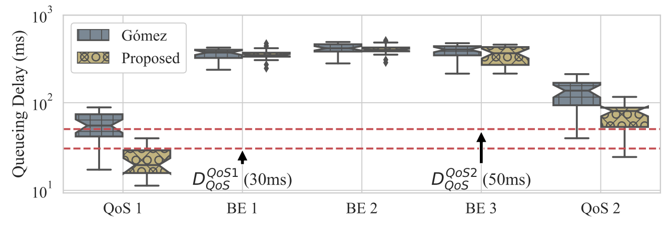

(b) Queueing delay (log scale) of slices.

Fig. 8. Box-and-whiskers plots showing the overall throughput and average queueing delay of slices for 30 experiments.

To get an overall picture, we ran 30 more experiments using both approaches. Then, we computed the average queueing delay and the average overall throughput achieved per slice at the end of each experiment. In this manner, we can evaluate, besides the overall throughput achieved per slice, whether the queueing delay requirements were maintained (on average) during the experiment timespan. Figure 8 presents the overall throughput and queueing delay per slice using both approaches. Figure 8a shows that our approach has less variability and fewer outliers, not to mention higher average overall throughput, especially for the BE flows. Most importantly, the average queueing delay of QoS 1 flow—which remained active for most of the experimentation—was not guaranteed using [16] (see Figure 8b).

Differently, our approach maintained the QoS 1 flow below its requirements. Both of the 1st and 3rd quartiles (interquartile range) were below 30 ms, meaning that about 75 % of the averages were below its queueing delay requirements (D_{QoS}^{QoS1}). However, for the QoS 2 flow, in both approaches, fewer than 25 % percent of averages were below its requirements (D_{QoS}^{QoS2}). Despite the averages hiding disparities, our slicing algorithm needs some time to react and enhance the QoS.

VI. Conclusion

In this paper, we proposed a delay-aware approach for network slicing and MAC management using MCDA in IEEE 802.11 SD-RANs. To perform load balancing in our network, we use the TOPSIS method to decide which AP STAs are assigned and slices are allocated. Six criteria were used for the decision-making with different weights defined for the QoS-constrained and BE services. Unlike existing work in load balancing, our approach periodically analyzes the queueing delay of all slices and performs runtime airtime-based slice re-configurations to enhance the QoS when needed. Through experimentation in real hardware, the results show that our approach is capable of performing runtime load balancing and ensures QoS even with higher demands. As future work, we plan to deploy our solution on a large scale testbed.

Acknowledgment

This research received partial funding from a research project of the Flemish Research Fund (FWO), from the European Union's Horizon 2020 Research and innovation program, under grant agreement No. 826284 (ProTego), and from the U.S. National Science Foundation NeTS Award 1813451.

References

[1] 5GPPP Architecture Working Group. (2017) View on 5G Architecture. [Online]. Available: https://5g-ppp.eu/wp-content/uploads/2018/01/5G-PPP-5G-Architecture-White-Paper-Jan-2018-v2.0.pdf

[2] P. H. Isolani, M. Claeys, C. Donato, L. Z. Granville, and S. Latré, "A Survey on the Programmability of Wireless MAC Protocols," *IEEE Communications Surveys Tutorials*, pp. 1–1, 2018.

[3] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification. Amendment 7: Medium Access Control (MAC) Quality of Service (QoS), ANSI/IEEE Std 802.11e, LAN/MAN Standards Commit- tee of the IEEE Computer Society Std., 2005.

[4] 3GPP, "Study on management and orchestration of network slicing for next generation network," 3GPP, Tech. Rep. TR 28.801 V15.0.0T, 2017.

[5] 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, June 2019.

[6] M. Richart, J. Baliosian, J. Serrat, and J. Gorricho, "Resource Slicing in Virtual Wireless Networks: A Survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 462–476, Sep. 2016.

[7] M. Richart, J. Baliosian, J. Serrati, J. 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)*, Nov 2017, pp. 1–4.

[8] T. Høiland-Jørgensen, M. Kazior, D. Täht, P. Hurtig, and A. Brunstrom, "Ending the Anomaly: Achieving Low Latency and Airtime Fairness in WiFi," in *2017 USENIX Annual Technical Conference (USENIX ATC 17)*. Santa Clara, CA: USENIX Association, 2017, pp. 139–151. [Online]. Available: https://www.usenix.org/conference/atc17/technical-sessions/presentation/hoilan-jorgesen

[9] J. J. Aleixendri, A. Betzler, and D. Camps-Mur, "A practical approach to slicing Wi-Fi RANs in future 5G networks," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1–6.

[10] T. Høiland-Jørgensen, P. Hurtig, and A. Brunstrom, "PoliFi: Airtime Policy Enforcement for WiFi," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE Press, 2019, p. 1–6. [Online]. Available: https://doi.org/10.1109/WCNC.2019.8885440

[11] E. Coronado, R. Riggio, J. Villa1ón, and A. Garrido, "Lasagna: Programming Abstractions for End-to-End Slicing in Software-Defined WLANs," in *2018 IEEE 19th International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*. IEEE, 2018, pp. 14–15.

[12] 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 IEEE 802.11 Networks," in *2019 Sixth International Conference on Software Defined Systems (SDS)*, June 2019, pp. 260–265.

[13] 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 IEEE 802.11 RANs," *IEEE Communications Letters*, vol. 24, no. 5, pp. 1077–1080, 2020.

[14] E. Coronado, R. Riggio, J. Villalón, and A. Garrido, "Wi-balance: Channel-aware user association in software-defined Wi-Fi networks," in *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, 2018, pp. 1–9.

[15] E. Zeljković, N. Slamnik-Kriještorac, S. Latré, and J. M. Marquez-Barja, "ABRAHAM: Machine Learning Backed Proactive Handover Algorithm Using SDN," *IEEE Transactions on Network and Service Management*, vol. 16, no. 4, pp. 1522–1536, 2019.

[16] B. Gomez, E. Coronado, J. Villalón, R. Riggio, and A. Garrido, "User Association in Software-Defined Wi-Fi Networks for Enhanced Resource Allocation," in *Proc. of IEEE WCNC*, Seoul, South Korea, 2020.

[17] S. Vassilaras, L. Gkatzikis, N. Liakopoulos, I. N. Stiakogiannakis, M. Qi, L. Shi, L. Liu, M. Debbah, and G. S. Paschos, "The Algorithmic Aspects of Network Slicing," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 112–119, Aug 2017.

[18] M. Agiwal, A. Roy, and N. Saxena, "Next Generation 5G Wireless Networks: A Comprehensive Survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 3, pp. 1617–1655, thirdquarter 2016.

[19] H. Luo and M. Shyu, "An Optimized Scheduling Scheme to Provide Quality of Service in 802.11e Wireless LAN," in *2009 11th IEEE International Symposium on Multimedia*, Dec 2009, pp. 651–656.

[20] P. Serrano, A. Banchs, P. Patras, and A. Azcorra, "Optimal Configuration of 802.11e EDCA for Real-Time and Data Traffic," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 5, pp. 2511–2528, Jun 2010.

[21] W. L. Pang, D. Chieng, and N. N. Ahmad, "Adaptive Priority Sliding Admission Control and Scheduling Scheme for DCF and EDCA WLANs," *Wireless Personal Communications*, vol. 70, no. 1, pp. 295–321, May 2013. [Online]. Available: https://doi.org/10.1007/s11277-012-0695-2

[22] E. Charfi, C. Gueguen, L. Chaari, B. Cousin, and L. Kamoun, "Dynamic frame aggregation scheduler for multimedia applications in IEEE 802.11n networks," *Transactions on Emerging Telecommunications Technologies*, vol. 28, no. 2, p. e2942, 2017.

[23] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Amendment 5: Enhancements for Higher Throughput, ANSI/IEEE Std 802.11n, LAN/MAN Standards Committee of the IEEE Computer Society Std., 2009.

[24] M. G. Sarret, J. S. Ashta, P. Mogensen, D. Catania, and A. F. Cattoni, "A Multi-QoS Aggregation Mechanism for Improved Fairness in WLAN," in *2013 IEEE 78th Vehicular Technology Conference (VTC Fall)*, Sep. 2013, pp. 1–5.

[25] D. Kim and S. An, "Throughput enhancement by Dynamic Frame Aggregation in multi-rate WLANs," in *2012 19th IEEE Symposium on Communications and Vehicular Technology in the Benelux (SCVT)*, Nov 2012, pp. 1–5.

[26] S. V. Azhari, O. Gurbuz, and O. Ercetin, "QoS based aggregation in high speed IEEE802.11 wireless networks," in *2016 Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, June 2016, pp. 1–7.

[27] B. Maqhat, M. Dani Baba, R. A. Rahman, and A. Saif, "Performance analysis of fair scheduler for A-MSDU aggregation in IEEE802.11n wireless networks," in *2014 2nd International Conference on Electrical, Electronics and System Engineering (ICEESE)*, Dec 2014, pp. 60–65.

[28] S. Seytnazarov and Y. Kim, "QoS-Aware Adaptive A-MPDU Aggregation Scheduler for Voice Traffic in Aggregation-Enabled High Throughput WLANs," *IEEE Transactions on Mobile Computing*, vol. 16, no. 10, pp. 2862–2875, Oct 2017.

[29] K. Nakauchi, Y. Shoji, and N. Nishinaga, "Airtime-based resource control in wireless LANs for wireless network virtualization," in *2012 Fourth International Conference on Ubiquitous and Future Networks (ICUFN)*, July 2012, pp. 166–169.

[30] K. Guo, S. Sanadhya, and T. Woo, "ViFi: Virtualizing WLAN Using Commodity Hardware," in *Proceedings of the 9th ACM Workshop on Mobility in the Evolving Internet Architecture*, ser. MobiArch '14. New York, NY, USA: ACM, 2014, pp. 25–30. [Online]. Available: http://doi.acm.org/10.1145/2645892.2645893

[31] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," in *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)*, vol. 2, March 2003, pp. 836–843 vol.2.

[32] R. Riggio, M. K. Marina, J. Schulz-Zander, S. Kuklinski, and T. Rasheed, "Programming Abstractions for Software-Defined Wireless Networks," *IEEE Transactions on Network and Service Management*, vol. 12, no. 2, pp. 146–162, June 2015.

[33] K. D. Goepel, "Implementation of an Online Software Tool for the Analytic Hierarchy Process (AHP-OS)," *International Journal of the Analytic Hierarchy Process*, vol. 10, no. 3, Dec. 2018. [Online]. Available: https://www.ijahp.org/index.php/IJAHP/article/view/590

[34] J. Wątróbski, J. Jankowski, P. Ziemba, A. Karczmarczyk, and M. Zioło, "Generalised framework for multi-criteria method selection," *Omega*, vol. 86, pp. 107–124, 2019.

[35] C.-L. Hwang and K. Yoon, "Methods for multiple attribute decision making," in *Multiple attribute decision making*. Springer, 1981, pp. 58–191.