Airtime-based Resource Allocation Modeling for Network Slicing in IEEE 802.11 RANs

P. H. Isolani, N. Cardona, C. Donato, G. A. Pérez, J. M. Marquez-Barja, L. Z. Granville, and S. Latré

***Abstract*—In this letter, we propose an airtime-based Resource Allocation (RA) model for network slicing in IEEE 802.11 Radio Access Networks (RANs). We formulate this problem as a Quadratically Constrained Quadratic Program (QCQP), where the overall queueing delay of the system is minimized while strict Ultra-Reliable Low Latency Communication (URLLC) constraints are respected. We evaluated our model using three different solvers where the optimal and feasible sets of airtime configurations were computed. We also validated our model with experimentation in real hardware. Our results show that the solution time for computing optimal and feasible configurations vary according to the slice's demand distribution and the number of slices to be allocated. Our findings support the need for precise RA over IEEE 802.11 RANs and present the limitations of performing such optimizations at runtime.**

***Index Terms*—Resource Allocation, Network Slicing, Quality of Service (QoS), URLLC, IEEE 802.11 RANs.**

I. Introduction

THE fifth generation of mobile networks (5G) aims to enable applications to run with lower latency, more reliability, massive connectivity, and improved energy efficiency. The ITU Radiocommunication Sector (ITU-R) has defined three main uses for 5G: Enhanced Mobile Broadband (eMBB), Ultra-Reliable Low Latency Communication (URLLC), and Massive Machine Type Communications (mMTC); these are envisioned to be deployed in the coming years. Among the stringent requirements, low latency is seen as crucial and URLLC as the key enabler in this new age of connectivity.

Network slices, besides operating independently from one another, provide networking resource and traffic isolation among users and services [1]. In the literature, many proposals focus on network slicing for IEEE 802.11 networks [2]. Network slicing is being used to address the Resource Allocation (RA) problem in IEEE 802.11 Radio Access Networks (RANs), providing the required bandwidth and allocating resources accordingly [3]–[5]. However, in addition to the required bandwidth, End-to-End (E2E) latency and reliability should be considered as well. Moreover, deciding how to efficiently manage slice resources is still an open issue.

P. H. Isolani, C. Donato, and S. Latré are with the IDLab, Department of Computer Science, University of Antwerp - imec, Antwerp, Belgium (e-mail: {pedro.isolani, carlos.donato, steven.latre}@uantwerpen.be).

N. Cardona and J. M. Marquez-Barja are with the IDLab, Department of Applied Engineering, University of Antwerp - imec, Antwerp, Belgium (e-mail: {nelson.cardona-cardenas, johann.marquez-barja}@uantwerpen.be)

N. Cardona is also with Smart Networks and Services, Fondazione Bruno Kessler Via Sommarive 18, 38123, Trento, Italy (e-mail: ncardona@fbk.eu).

G. A. Pérez is with the FOTS lab, Department of Computer Science, University of Antwerp, Belgium (e-mail: guillermoalberto.perez@uantwerpen.be)

L. Z. Granville is with the Computer Networks Group, Federal University of Rio Grande do Sul, Brazil (e-mail: granville@inf.ufrgs.br).

Nowadays, advanced 5G services and applications have stringent Quality of Service (QoS) requirements, which make the RA problem even more complex. To cope with dynamic and unstable wireless environments and to meet QoS requirements, network slices might be instantiated, modified, and terminated at runtime. Therefore, such slices have to be properly and dynamically managed. To the best of our knowledge, this is the first work to exploit the flexibility of slice airtime allocation considering the IEEE 802.11 RAN resource availability and the stringent latency Key Performance Indicator (KPI) related to URLLC. We use a scheduling policy able to assign different airtime configurations to each slice, named Airtime Deficit Weighted Round Robin (ADWRR) [5]. Thus, each slice can be configured according to their priority.

In this letter, we focus on an RA model that optimizes the overall queueing delay of the system while QoS constraints are respected. Given that flows are distributed among slices that, in this case, represent abstractions of queues, we model this problem using concepts of Queuing Theory. For that, we propose a Quadratically Constrained Quadratic Program (QCQP). We evaluated our model using Advanced Process OPTimizer (APOPT), Interior Point OPTimizer (IPOPT), and Z3 [6] solvers. We then validated our model with experimentation in real hardware. Our results show that runtime optimization is limited by the demand distribution and the number of slices to be allocated. Our findings support the need for precise RA over IEEE 802.11 RANs and present the limitations of performing such optimizations at runtime.

II. IEEE RAN-RA Problem Formulation

The IEEE 802.11 RAN consists of a set Access Points (APs) responsible to deliver data from different services to several users in the network, i.e., Stations (STAs). Each AP has resources to be shared and therefore has to be properly managed. Multiple *tenants* (i.e., virtual operators or service providers) share from the same infrastructure and have their specific Service Level Agreements (SLAs). These SLAs are translated into QoS requirements that the network has to support (e.g., minimum throughput, maximum allowed E2E latency, and acceptable packet loss ratio). To meet such requirements, we propose the use of network slicing. According to Richart et al. [3], there are two variants of network slicing. The first abstracts the different services and ensures QoS within them, which is referred to as 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, which

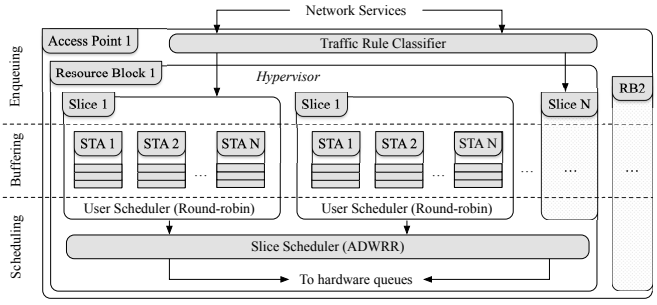


Fig. 1. Simplified slice queue structure along with the data traffic flow.

is called Infrastructure Sharing Slicing (ISS). As we focus on QoS within a slice as being a service, we use the QoSS variant.

A. Problem Statement & Assumptions

To represent the minimum chunk of wireless resources that can be assigned to a user, we use the *Resource Block abstraction* from Riggio et al. [7]. This abstraction defines a WiFi interface at a given AP, identified by the network interface identifier (e.g., Medium Access Control (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). Therefore, each resource block has its transmission capabilities, e.g., maximum dequeuing rate μ^b_{MAX}.

For each *resource block*, frames are first classified into the different queues as slices, based on the definition of traffic rules (e.g., OpenFlow rules). Figure 1 depicts a simplified queue structure along with the data traffic flow within a single AP. Thereafter, frames belonging to such slices/queues are dequeued following the ADWRR scheduling scheme [5]. With such a scheduling scheme, different portions of airtime, *a.k.a.*, *quantums*, are allocated to each slice s in each transmission round, denoted Q^s. In this manner, larger values for Q^s can be assigned to slices supporting services with stricter performance requirements and, hence, more radio resources are allocated.

According to the dequeuing process of the ADWRR scheduling policy, the *hypervisor* only serves traffic rules whose expected transmission time, estimated by a rate control algorithm (e.g., Minstrel), is smaller than a deficit counter. In each transmission round of a given slice s, a quantum Q^s is added to this deficit counter. If this counter is greater than the estimated airtime consumed to transmit the current frame, transmission occurs and the counter is decremented by the used airtime. Otherwise, if the transmission time of a given frame exceeds the deficit counter, this is held back until the next round of the scheduler. Besides, each traffic rule contains multiple aggregation buffers, one for each user in the slice, which can be scheduled as intended. Nevertheless, inactive traffic rules do not cause any performance degradation.

As stated in [5], the expected transmission airtime A for a packet with L bits long can be approximated by:

$$A = \frac{1}{P(R_{\mathrm{BEST}})}\left(\mathrm{DIFS} + \frac{L}{R_{\mathrm{BEST}}} + \mathrm{SIFS} + T_{\mathrm{ACK}}\right). \quad (1)$$

In Equation 1, R_{BEST} is the Modulation and Coding Scheme (MCS) with the highest throughput and delivery probability of transmitting a frame and receiving the WiFi Acknowledgment (ACK) using the MCS R_{BEST}. Note that Shortest Interframe Spacing (SIFS), DCF Interframe Spacing (DIFS), and the time for receiving the Acknowledgement T_{ACK} are considered.

Given that network flows are distributed among slices and slices represent abstractions of queues, we model this problem using Queuing Theory. Besides, since such flows' arrivals are independent of one another and the traffic pattern is random, we consider that flows' arrivals follow a Poisson distribution. Nonetheless, we acknowledge the inability to capture traffic burstiness in which characterizes some data traffic patterns. However, our focus is to find the optimal assignment of Q^s and validate it through real hardware experimentation.

B. Mathematical Model

The described network slicing RA problem can be approached through optimization techniques. In this section, we present a QCQP approach for constraints. Since slices represent queues and flow's arrivals follow a Poisson distribution with exponential services, in Kendall's notation, our model can be represented as sets of M/M/1. Given n services to be delivered by a resource block b, n slices are instantiated. For each slice s in the set of slices S^b of a resource block b, the maximum dequeuing rate μ^b_{MAX}, the dequeuing rate λ^s, the frame size FRM^s_{size}, and the airtime A^s used for its transmissions are provided. With such an input, the average queueing delay W^s of each slice can be calculated and the maximum average queueing delay W^s_{QoS} can be ensured. Besides respecting the service delivery rate and queueing delay constraints, we aim to find the optimal assignment of Q^s that minimizes the overall queueing delay. Therefore,

$$\mathrm{Minimize} \sum_{b\in B}\sum_{s\in S^b} W^s, \quad (2)$$

subject to the following set of constraints:

1) Queueing delay: The average queueing delay W^s of each slice s in resource block b is calculated as follows:

$$\forall s \in S^b : W^s = \frac{L^s}{\lambda^s}, \quad (3)$$

where L^s is the average number of frames in slice s. Notwithstanding, queueing delay constraints have to be ensured, hence,

$$\forall s \in S^b : W^s \leq W^s_{\mathrm{QoS}}. \quad (4)$$

Nonetheless, the average number of frames L^s is given by:

$$\forall s \in S^b : L^s = \frac{{\rho^s}^2}{1-\rho^s}, \quad (5)$$

where ρ^s defines the service utilization on a given slice.

2) Service Utilization: The service utilization ρ^s of a slice s in a resource block b is given by:

$$\forall s \in S^b : \rho^s = \frac{\lambda^s}{\mu^s}, \quad (6)$$

subject to the actual dequeuing rate μ^s, assigned to the slice s. To prevent more arrivals than the server is capable of handling, we determine that the ρ has to respect the following constraint:

$$\forall s \in S^b : \rho^s < 1, \quad (7)$$

ensuring that frames are dequeued at a faster pace than enqueued, thus preventing queues to grow indefinitely.

3) Service delivery rate: According to the ADWRR algorithm, the rate in which frames are dequeued in a slice is given by its available airtime by the airtime used for its transmissions. This also has to consider all other competing slices within the same resource block. Therefore, the frame dequeuing rate μ^s_{FRM} is calculated as follows:

$$\forall s \in S^b, \forall b \in B : \mu^s_{\mathrm{FRM}} = \begin{cases} 1 & \text{if } \frac{Q^s}{A^s} \geq 1 \\ \frac{Q^s}{A^s} & \text{otherwise.} \end{cases} \quad (8)$$

Subsequently, the allocated dequeuing rate μ^s is given by:

$$\forall s \in S^b, \forall b \in B : \mu^s = \frac{\mu^s_{\mathrm{FRM}}}{\sum_{i \in S^b} \mu^i_{\mathrm{FRM}}} \cdot \mu^b_{\mathrm{MAX}}. \quad (9)$$

It is important to emphasize that the computation of μ^b_{MAX} is not within the scope of this letter. Since μ^b_{MAX} depends on how lower MAC and physical layers would behave according to network conditions, we assume that any Software-Defined Networking (SDN)-based admission control system can be introduced to estimate such capacity. We focus on the assignment of airtime portions within the upper MAC layer, thus, we have considered μ^b_{MAX} as an input of our problem. Besides, μ^s represents the dequeuing rate reserved for slice s, ensured by the hypervisor. Thus, in cases where the channel is saturated and the transmission limit of the resource block is reached, each of the slices receives its respective μ^s. On the other hand, slices share the remaining resources equally.

4) Resource Allocation: Slices in a resource block can only allocate a limited amount of resources, i.e., μ^s. Intuitively, the sum of such allocated resources within a resource block cannot exceed its maximum capabilities. However, to be able to optimize the overall queueing delay of the system, all the available resources of the resource block should be allocated:

$$\forall b \in B : \sum_{s \in S^b} \mu^s = \mu^b_{\mathrm{MAX}}. \quad (10)$$

Besides, slices must provide the required dequeuing rate λ^s.

$$\forall s \in S^b, \forall b \in B : \mu^s \geq \lambda^s. \quad (11)$$

Nevertheless, each resource block b must have enough resources to support its demands. Thus,

$$\forall s \in S^b, \forall b \in B : \sum_{s \in S^b} \lambda^s < \mu^b_{\mathrm{MAX}}. \quad (12)$$

Otherwise, there is no feasible solution. With such a model, our goal is to find values for the Q^s decision variables that dictate the portion of airtime allocated to each slice on each resource block. In this manner, besides satisfying all performance constraints in terms of dequeuing rate and queueing delay, the remaining resources are optimally shared. Hence, reducing the overall queueing delay of the entire system.

III. Performance Evaluation

A. Methodology and Workload

The parameters for our evaluation are defined in Table I. We set equal values for the airtime A^s at each round of the ADWRR scheduling algorithm. Likewise, to ensure that the problem is always feasible, we set all W^s_{QoS} to $1\,\mathrm{s}$.

TABLE I
List of input parameters used during the evaluation

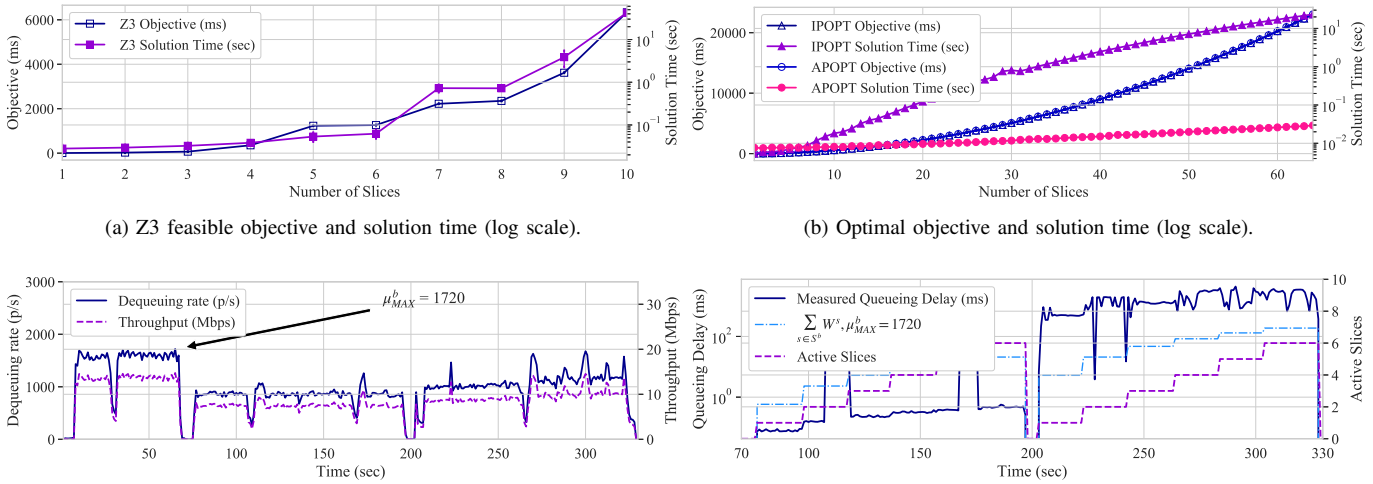
Parameter	Value	Description
n	From 1 to 64	Number of slices in a resource block b. $n \in \mathbb{N}$
S^b	From 1 to 64	Set of slices in a resource block b.
$\forall s \in S^b : A^s$	$12\,000\,\mathrm{us}$	Airtime needed to transmit a frame in slice s.
μ^b_{MAX}	$1600\,\mathrm{p/s}$	Maximum dequeuing rate of resource block b.
$\forall s \in S^b : \mathrm{FRM}^s_{size}$	$1024\,\mathrm{bytes}$	Frame size in slice s.
μ^b_{GAP}	$\mu^b_{\mathrm{MAX}} \cdot 0.1$	Dequeing rate gap in resource block b.
$\forall s \in S^b : \lambda^s$	$\frac{\mu^b_{\mathrm{MAX}} - \mu^b_{\mathrm{GAP}}}{n}$	Required dequeuing rate in slice s.
$\forall s \in S^b : W^s_{\mathrm{QoS}}$	$1\,\mathrm{s}$	Maximum average queueing delay in slice s.

To solve our model, we first identify if the problem is feasible using the Z3 Satisfiability Modulo Theories (SMT) solver [6]. Then, we check how fast the optimal solution can be found with APOPT and IPOPT solvers, using GEKKO [8]. As a benchmark, we used a laptop equipped with an Intel Core i7 (2,2 GHz) processor and 8 GB 1600 MHz DDR3. We opted to use APOPT and IPOPT because they are based on different approaches, which have fundamental differences in handling constraints. IPOPT is based on the Interior Point Method (IPM) in which has proven to scale very well to large Nonlinear Programming (NLP) problems with a small number of constraints. On the other hand, APOPT is based on active-set Sequential Quadratic Programming (SQP) in which scale very well to large NLP problems with a high number of constraints. Nonetheless, Z3 is an SMT solver which is used in several program analysis, verification, test case generation projects. With Z3, is assured that a given problem is feasible and it has been proven to be fast in satisfiable instances.

To validate our model, we run an experiment in a single resource block using the parameters defined in Table I. A computer connected to the wired segment generates different flows towards a single STA, each passing through a different slice. To estimate μ^b_{MAX}, we generate bursts with frame size equal to $1024\,\mathrm{bytes}$ until the maximum transmission capability is reached. With the μ^b_{MAX}, we draw the expected average queueing delay according to our model. Thereafter, we generate different flows following the Poisson distribution and we introduce up to 6 slices configured with equal Q^s. Alongside this, we measured the queueing delay during the whole experiment run. We also verified when ≈50% of the resources remain to be allocated, avoiding channel saturation/conditions to affect the results. For this experimentation, we used an AP based on the PC Engines APU2D4 (x64) processing board, equipped with one Qualcomm Atheros AR958x 802.11 a/b/g/n and a Raspberry Pi 4 B with an 802.11b/g/n/ac for the STA.

B. Evaluation Results and Analysis

Figure 2 illustrates the objective function and the solution time for computing feasible and optimal solutions according to the number of slices. The results reported are the average of 10 runs. We can observe in Figure 2a that Z3 has reasonable performance in terms of execution time, especially



(a) Z3 feasible objective and solution time (log scale).

(b) Optimal objective and solution time (log scale).

(c) Resource block μ^b along the experiment run.

(d) Queueing delay (log scale) and number of slices along the experiment run.

Fig. 2. Z3, APOPT, and IPOPT feasible and optimal solutions computation along with real hardware experimentation.

for the problems where the number of slices is less than 7. However, nonlinear arithmetic is undecidable. Therefore, its semi-algorithm does not guarantee, for every input, a yes-or-no-answer within a reasonable amount of time. Hence, for the larger instances of the problem where a solution is not found within $10\,\mathrm{min}$, we cancel the experiments. On the other hand, Figure 2b shows that both APOPT and IPOPT solve the problem for all sets of slices with the same objective. Moreover, as expected, both objective and solution time increase along with the number of slices, i.e., queues. In most cases, APOPT solves the problem faster and with the same optimal solution.

Figure 2c presents the dequeuing rate and the achieved throughput while Figure 2d presents the number of active slices, measured queueing delay, and expected $\sum_{s \in S^b} W^s$ for a given μ^b_{MAX}. In the first 70 seconds, we measure μ^b_{MAX} with $1720\,\mathrm{p/s}$ while the burst of frames was introduced. Results shown from the second 70 to 200 and from the second 200 to the end are relative to μ^b_{GAP} of ≈50% and 10%, respectively. As we can see in Figure 2c, the measured queueing delay follows the expected pattern, increasing with the number of slices. However, when resources available are near the limit (e.g, ≈10%), due to channel conditions, devices' processing capabilities, or lower MAC and physical layer behavior, the measured queueing delay increases above the expected. Therefore, it remains a challenge to estimate μ^b_{MAX} according to the current channel conditions and network configuration.

IV. Conclusion

In this letter, we have studied the optimal allocation of network slices in IEEE 802.11 RANs. We have proposed an airtime-based RA model where slices are configured to support strict QoS requirements such as the URLLC. To compute optimal and feasible solutions, we conducted experiments using APOPT, IPOPT, and Z3 solvers. Besides, we validated our model with experimentation in real hardware. Our implementation is publicly available on GitHub[1]. Our results show that the solution time for computing the optimal airtime configuration varies according to the number of slices to be allocated and the strictness of the QoS constraints. Our findings support the need for precise RA over IEEE 802.11 RANs and present the limitations of performing such optimizations at runtime. As future work, we plan to approach such a problem with fast control algorithms.

[1]https://github.com/phisolani/optimal_wifi_slicing

Acknowledgment

This research received partial funding from the Flemish Government under the "Onderzoeksprogramma Artificile Intelligentie (AI) Vlaanderen" program and from the European Union's Horizon 2020 Research and innovation program, under grant agreement No. 826284 (ProTego).

References

[1] 3GPP, "Study on management and orchestration of network slicing for next generation network," 3GPP, Tech. Rep. TR 28.801 V15.0.0T, 2017.

[2] 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.

[3] M. Richart, J. Baliosian, J. Serrati, J. Gorricho, R. Agero, 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.

[4] 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.

[5] E. Coronado, R. Riggio, J. Villalón, 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)*, 2018, pp. 14–15.

[6] L. de Moura and N. Bjørner, "Z3: An efficient SMT solver," in *Tools and Algorithms for the Construction and Analysis of Systems*, C. R. Ramakrishnan and J. Rehof, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 337–340.

[7] 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.

[8] L. Beal, D. Hill, R. Martin, and J. Hedengren, "GEKKO optimization suite," *Processes*, vol. 6, no. 8, p. 106, 2018.