Analysis of a Multi-Armed Bandit Solution to Improve the Spatial Reuse of Next-Generation WLANs

Anthony Bardou[a], Thomas Begin[a], Anthony Busson[a]

[a] *Univ. Lyon, ENS de Lyon, UCBL, CNRS, LIP, 42 allee d'Italie, Lyon, 69007, France*

Abstract

The next generation of WLANs will be, for the most part, ubiquitous in urban areas, densely deployed, and implementing the latest amendment of IEEE 802.11 standard, namely 802.11ax also known as Wi-Fi 6. Among the main purposes of 802.11ax is the improvement of the spatial reuse of radio channels by allowing the dynamical update of the sensitivity threshold and the transmission power at each node. In this regard, our contributions are twofold. First, we investigate the performance improvement resulting from a more efficient spatial reuse of radio channels with 802.11ax. Second, we introduce a centralized solution based on the Multi-Armed Bandit (MAB) framework and a sub-sampling technique to quickly discover an appropriate configuration of the sensitivity threshold and transmission power at each access point. We evaluate our solution with the network simulator ns-3 on different network topologies. The simulation results show the ability of our solution to quickly and robustly adjust these parameters of access points in order to significantly improve the behavior of WLANs.

Keywords: Spatial Reuse, IEEE 802.11 WLANs, Power Control, Multi-Armed Bandits

1. Introduction

Wireless Local Area Networks (WLANs) have become ubiquitous in urban areas. They have overtaken wired networks and mobile networks to become the primary means of connecting end-users to the Internet. According to Cisco's forecasts [1], Wi-Fi hotspots will grow four-fold from 2018 to 2023. Globally, there will be nearly 628 million public Wi-Fi hotspots by 2023, up from 169 million hotspots in 2018. The de-facto norm for WLANs is the

802.11 standard [2] of IEEE, better known under its commercial branding Wi-Fi. In its simplest form, a WLAN is made of wireless stations (STAs) and an Access Point (AP) that relays the traffic of STAs back and forth to the wired network using radio wave communication.
In many urban places where a large number of end-users may be needing Internet access (e.g., medium or large enterprises, universities, train stations, airports, shopping centers), WLANs are expanded to involve multiple APs. APs are typically deployed at a short distance from other APs of the same WLAN forming a dense WLAN. Network administrators often rely on a central controller, typically implemented as software, for the remote control and management of the APs' fleet. This greatly eases maintenance tasks such as the deployment of a homogeneous setting over the APs (e.g., for a security update) or performance optimization routines (e.g., efficient coordination of the APs).
With a dense mesh of APs, one could expect WLANs to be able to ensure proper radio coverage, to perform high physical transmission rates between STAs and APs, and overall to sustain an important traffic load. However, in practice, WLANs' performance are sometimes viewed as insufficient with some STAs struggling at transmitting or receiving their data. This is because the bottleneck resource is not the number of APs but rather the scarcity of space on the radio spectrum.
The radio bands used by 802.11 are divided into different channels and each AP must be assigned to a channel. In general, the network administrators aim at staggering adjacent APs on different, non-overlapping radio channels to enable simultaneous transmissions. Unfortunately, the number of radio channels is limited and any channel assignment strategy will find its limitations when the density of APs is high. However, simultaneous transmissions of two APs on the same channel may yet be achieved if the distance between the two is long enough to significantly dampen their mutual interference at the destinations. In the current WLANs, this latter property, known as the spatial reuse of the radio channel, is not exploited at its full potential.
The 802.11ax standard, which was approved in 2021 and is marketed as Wi-Fi 6, introduces a new feature to further improve the spatial reuse of radio channels. Two key parameters, namely the transmission power and the sensitivity threshold, referred to as `TX_PWR` and `OBSS/PD` respectively, have become tunable and can be set independently for each 802.11ax device. The setting of these parameters greatly improves the benefits of spatial reuse and thus the performance of WLANs. However, no algorithms were included

in the standard and it is up to the manufacturer or to the WLAN controller to decide how these parameters should be set.
Configuring these two parameters is a difficult problem for three main reasons. First, any efficient configuration of these parameters is tightly linked to the WLAN topology (i.e., arrangements of APs and STAs) and will likely be inefficient, if not counterproductive, for another WLAN. Second, the problem is subject to the so-called curse of dimensionality. Because each AP has 2 parameters, each ranging over 21 different values, the number of possible network configurations grows in $O(2^{N_A})$ with N_A denoting the number of APs in the WLAN. This exponential growth of the size of the state space precludes the use of a brute-force approach even for a medium-sized WLAN. Third, the configuration of these parameters must be found relatively quickly and without any noticeable disruption of service for the STAs. On the other hand, the WLAN controller can perform configuration tests, and observe the resulting WLAN behavior on the STAs performance. This paves the way for the use of data-driven approaches such as reinforcement learning techniques to perform the search for an efficient configuration of the WLAN parameters.
In this paper, our objective is to improve the spatial reuse of radio channels of WLANs by modifying the configuration of the `TX_PWR` and `OBSS/PD` parameters of each AP. We cast the search for this parameter setting as a Multi-Armed Bandit (MAB) problem to which we propose a fast and efficient solution. The current paper extends a preliminary solution presented in [3] in several ways. It contains a more detailed comparison of our proposed strategy with the existing solutions as well as a robustness study to determine if the parameter configurations found by our solution remain relevant if the WLAN's workload changes drastically. Through them, we demonstrate that our solution can significantly improve the behavior of WLANs, leading in particular to a fairer share of the radio channel among the STAs, even if the WLAN's workload is prone to significant variations. More precisely, our contributions are as follows:

- The efficiency of any reinforcement solution heavily relies on the definition of its reward function. We devised a reward function that accounts for the potential issues of WLANs (unfairness, starvations) and reflects the overall goodness of a network configuration from the standpoint of a network administrator.

- While a uniform sampling may initially appear a natural choice to explore the space of network configurations, we opt for a Gaussian

mixture approach. This choice leverages a certain degree of smoothness in the rewards of similar network configurations and contributes to ensuring seamless and uninterrupted connectivity to the STAs.

- Using several WLAN scenarios run on a realistic network simulator, namely ns-3, we show the superiority of our approach at addressing the spatial reuse problem over traditional ways of performing the sampling and optimization steps within the MAB framework.

- Through ns-3 simulations, we verify the robustness of the configuration found by our solution by evaluating its performance when facing other levels of WLAN's workload and comparing them with those attained under the 802.11 default configuration.

2. IEEE 802.11ax and spatial reuse

2.1. Overview of 802.11ax

The IEEE 802.11ax amendment builds on the strengths of 802.11ac and gave birth to the sixth generation of Wi-Fi, aka Wi-Fi 6. Unlike 802.11ac, 802.11ax is a dual-band operating on 2.4 and 5 GHz and pushes towards more flexibility, predictability, and scalability.

2.1.1. Flexibility

802.11ax introduces a new power-saving mechanism called Target-Wakeup Time (TWT) [4]. With TWT, the sleep time of STAs is no longer synced by the APs' beacons. Instead, STAs negotiate with their associated AP to schedule recurrent Service Periods (SPs) in which they will wake up for their transmissions. This scheme outperforms previous mechanisms such as TIM (e.g., [5]) and segmentation TIM (e.g., [6]) and enables significant power savings gains for battery-powered STAs such as Internet-of-Things (IoT) devices.

2.1.2. Predictability

The performance of Wi-Fi are often perceived as uncertain and this can be troublesome for applications such as videoconferencing and IoT. To address this issue, 802.11ax introduces the use of OFDMA which redefines the way STAs and APs access the shared medium (radio channel) (e.g., [4, 7]). With OFDMA, 802.11ax may implement and reserve contention-free periods for STAs that need precise control (determinism) over their performance.

2.1.3. Scalability

Last but not least, 802.11ax pushes the boundaries of Wi-Fi when operating in dense environments. First, 802.11ax introduces denser modulations, an increased number of spatial streams, and reduced per-symbol overhead. Second, 802.11ax enables improved spatial reuse of the radio channels by dynamically selecting appropriate levels for the sensitivity threshold `OBSS/PD` and the transmission power `TX_PWR`. For instance, in current WLANs where these parameters are static, some STAs may transmit with too much power given their proximity to their associated AP, thereby generating unnecessary interference.

2.2. State of the art on spatial reuse

The literature related to the spatial reuse of radio channel can be classified into four major categories based on whether papers address the issue of channel allocation or the tuning of the `TX_PWR` and `OBSS/PD` parameters, and on whether the proposed solutions are based on analytical modeling, or conversely, mostly data-driven approaches. In this section, we review the literature associated with each group.

2.3. Allocating the radio channels

The first and foremost way of improving the spatial reuse of radio channels in an 802.11-based WLAN is obviously to allocate the same channel to multiple of its APs. Indeed, provided that two APs do not sense each other, they can then transmit at the same time without any risk of interfering, nor any need to share the communication resource materialized by the radio channel. The search for optimized solutions when multiple radio channel allocations exist is known as the channel allocation (CA) problem. Existing solutions to this problem are either model-based algorithms where an analytical model of the WLAN helps evaluate the quality of an allocation, or data-driven solutions where allocations are appraised through real measurements.

In model-based solutions, the CA problem can be tackled as a coloring problem with specific constraints. In the first generations of the 802.11 standard, channels had a fixed size of 20 MHz. For example, a centralized algorithm has been proposed in [8]. Conflicts between APs are represented by a graph, and the solution aims to allocate different channels/colors to APs that are adjacent in this graph. The algorithm was evaluated using a real testbed. With the recent amendments to the 802.11 standard, several 20MHz channels can be aggregated into a 40, 80, or 160MHz channel. This technique known

as channel bonding (CB) hardens the CA problem as the number of possible allocations increases significantly. In [9], a distributed algorithm named SA (Spectrum Assignment for WLAN) is formulated as an optimization problem. For a given topology, the algorithm aims at minimizing interference between APs while taking into account the preferences of APs for certain channel width. Another model-based approach is investigated in [10] where the analytical model accounts for both collisions and interference. In the case of an 802.11ac-based WLAN where the objective is to maximize the throughput for a given traffic demand, the authors approached the CB problem as an optimization problem whose solution is found through a genetic algorithm.
To be effective, model-based approaches require vast pieces of knowledge on the WLAN (topology, traffic, radio propagation, parameter setting, etc.), accurate analytical estimates of the network performance, and a relatively simple derivation. Unfortunately, it is often hard to meet these requirements and modeling approaches may be regarded as too inaccurate to handle the CA problem. On the other hand, data-driven approaches based on measurements collected from different WLAN configurations are intrinsically free of these constraints and thus appear promising. Some rely on heuristics (e.g., [11, 12]) while others make use of machine learning techniques (e.g., [13]). In [11], the authors propose a set of decentralized algorithms to allocate 20MHz channels to APs in order to efficiently reuse radio channel thereby increasing the overall network capacity. These algorithms are based on local measurements where, iteratively, each AP selects a channel with a certain probability, measures the channel performance for a certain period, and then adapts the probabilities for this channel accordingly. In [12], the proposed algorithm is based on the activity of the channels. When an AP tests a new channel, it associates a satisfaction score based on what it has been able to send on this channel during a certain period. If the score is satisfactory, the AP remains on this channel. Otherwise, it resumes its exploration efforts on other channels. In [13], the authors resort to a reinforcement learning algorithm to explore in real-time new configurations and to exploit the ones that offer better performance. The authors use a MAB approach with the Thompson sampling algorithm to select the new configurations to evaluate.

2.4. Tuning the `TX_PWR` and `OBSS/PD` parameters

Tuning the `TX_PWR` and `OBSS/PD` parameters (related to the transmission power and to the sensitivity threshold of nodes, respectively) is a secondary

means (beyond CA) to improve the spatial reuse of a radio channel. Pioneering efforts were made in 2004 with [14] in which the authors present an analytical model for deriving the optimal sensitivity threshold in a Wi-Fi mesh network. The physical carrier sensing threshold is tuned dynamically on each node as a function of the channel conditions. In [15], it is the transmission power that is tuned to increase the throughput and minimize the communication energy consumption. However, it is only in 2021 with the 802.11ax amendment that IEEE officially introduced the adaptation of the `TX_PWR` and `OBSS/PD` parameters thereby setting the technological context and constraints. In practice, the large number of parameters and the complexity of the physical layer in a radio environment hinders the use of such analytical model-based solutions. Instead, measurement-based techniques appear as natural candidates to this adaptation problem.

Practical approaches have been proposed in [16] and [17] to adapt the values of `TX_PWR` and `OBSS/PD`. In [16], the authors present a relatively simple way of dynamically tuning these two parameters. Using the Expected Transmission Count (ETX) value, their algorithm estimates a new value for `TX_PWR` as well as for `OBSS/PD`. In [17], a distributed solution aims to adapt dynamically the `OBSS/PD` as a function of the received signal strength. More precisely, the difference of signal strength between the frames in reception and the interfering frames (from other APs) is used to set a new value for `OBSS/PD`. The authors can control the likeliness of concurrent transmissions and thereby the level of "aggressiveness" in the selected configuration using an internal parameter of the algorithm.

More recently, some works have proposed methods inspired by machine learning techniques to address the issue of tuning the `TX_PWR` and `OBSS/PD` parameters. In [18, 19], the authors formalize the problem of allocating the radio channel of APs and setting their `TX_PWR` and `OBSS/PD` parameters as a MAB problem. In both cases, the MAB algorithm is applied at each AP in a distributed way. The two solutions mostly differ in terms of their reward definition. In the first solution, the reward at each AP corresponds to its throughput, which can be described as a "selfish" solution since each AP tries to optimize its own reward independently of the other nodes. Conversely, the second reward revolves around a max-min function of the throughputs of the current AP and of its direct neighbors (set of nodes for which the current AP senses traffic). Using a home-made simulator, the authors show that their solution significantly outperforms the default configuration of the WLAN and that the selfish reward may lead to unfair situations between APs or STAs.

Eventually, in [20], the authors propose an offline federated learning solution using simple feed-forward neural networks to predict the performance of 802.11ax-based WLANs. Although some promising results have been obtained against vanilla solutions, the model was trained offline on synthetic data obtained from a home-made simulator. The ability of the solution to handle other WLANs remains open.
To summarize, only a limited number of studies have tackled the issue of setting the `TX_PWR` and `OBSS/PD` parameter in an attempt to increase the spatial reuse of radio channel for WLANs. Data-driven approaches such as machine learning techniques appear well suited to deal with the intrinsic complexities of this issue. Unlike a couple of previous works that proposed distributed approaches wherein each AP sets its parameters based on its knowledge [18, 19], we introduce a centralized solution in which the WLAN controller configures the parameters of the APs composing its fleet. We propose a novel definition for the reward function specifically tailored to WLAN performance. Additionally, we devise a Gaussian mixture-based approach to explore the high dimensional state space of configurations. Finally, to the best of our knowledge, we are the first to use the popular open source network simulator ns-3, which includes a realistic representation of the physical, link, network, transport and application layers, to show the efficiency of our solution at setting the `TX_PWR` and `OBSS/PD` parameters. We believe that the use of this well-established simulator strengthens the validation of our solution.

3. WLAN under study

We consider a WLAN comprising multiple APs and stationary STAs as well as a controller that configures and manages the WLAN. STAs are associated to the AP with the strongest signal strength. To access the radio channel, APs and STAs use a listen-before-talk scheme referred to as carrier-sense multiple access with congestion avoidance (CSMA/CA) and accomplished by the distributed coordination function (DCF) in the 802.11 standards. DCF requires each node (AP and STA) willing to transmit to first sense the radio channel state for a short period of time. If the channel is sensed busy, the node will defer its transmission for a random period of time called backoff. If the channel is sensed idle (or after the backoff timer has come to zero), the node is allowed to transmit its frame. For more details, we refer the interested reader to [21].

The 802.11 standards rely on a clear channel assessment (CCA) function to indicate if the radio channel is perceived as busy or idle. Although other options are made possible, CCA is most often performed by comparing the power of the received signal (in dBm) to a given ceiling threshold often referred to as sensitivity and denoted by `OBSS/PD`. If the former exceeds the latter, the radio channel is considered busy. Otherwise, it is detected as idle. Until the recent release of 802.11ax, the `OBSS/PD` was set to a constant value (e.g., -82dBm for 802.11n). Analogously, the transmission power denoted by `TX_PWR`, which deviates from the received signal power due to the path loss and shadowing effects, was also constant and often set to 20dBm [22]. The latest amendment of 802.11, namely 802.11ax, enables the values of `TX_PWR` and `OBSS/PD` to be dynamically changed within certain ranges (e.g., [4]). While `TX_PWR` can take all values in between 1 and 21dBm, `OBSS/PD` can vary from -82 to -62dBm provided the two parameters meet relation (1), given by [23]. In our case, we assume that the WLAN controller is able to set the `TX_PWR` and `OBSS/PD` values for each AP.

$$\texttt{OBSS/PD} \leq \max(-82, \min(-62, -82 + (20 - \texttt{TX\_PWR}))). \tag{1}$$

A node is said to be in conflict with another if the former is made unable to transmit (due to the outcome of its CCA function) when the latter is currently transmitting. Conflicts between APs heavily influences the performance of a WLAN. They reduce channel interference and the probability of colliding frames but they also tend to limit the number of simultaneous transmissions in a WLAN and hence the spatial reuse of a radio channel. Due to the importance of conflicts in the understanding of a WLAN behavior, it is a common practice to represent WLANs by their conflict graph between APs (e.g., [8, 24, 25]). Note that conflicts of STAs are typically not represented in conflict graphs as the vast majority of traffic in WLANs is downstream (STAs typically generate at least an order of magnitude less traffic than APs). Figure 1 shows two conflict graphs associated with the same WLAN but with different settings of their AP's `TX_PWR` and `OBSS/PD` parameters. We can see the corresponding conflicts between APs when all APs have the same setting (default value). Conversely, when APs have different settings for their `TX_PWR` and `OBSS/PD` parameters, we observe that, with this particular setting, the number of conflicts between APs, which are no more symmetrical, has significantly decreased.

Several performance metrics are worth of interest to evaluate the efficiency

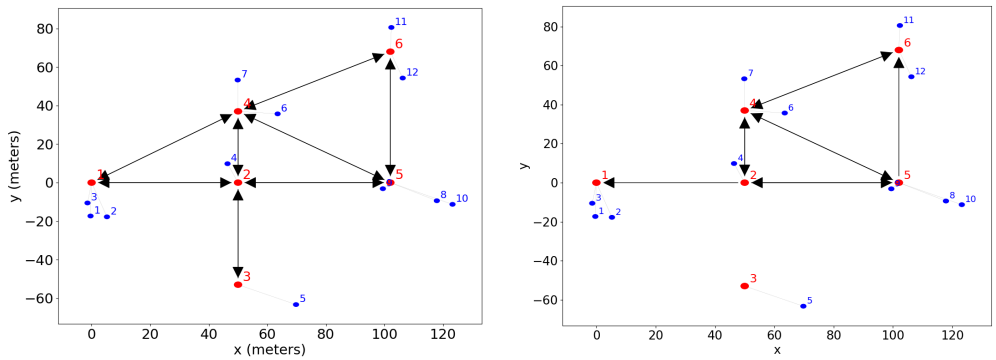

Figure 1: Example of two conflict graphs resulting from different settings of `TX_PWR` and `OBSS/PD` for a same WLAN. Red dots represent APs and blue dots represent STAs while the numbers are only here to ease the identification of nodes. Left: default configuration (`TX_PWR`, `OBSS/PD`) = (20,-82) dBm for all APs. Right: (`TX_PWR`, `OBSS/PD`) = (15,-81), (18,-80), (17,-79), (19,-82), (20,-82), (19,-82) dBm.

of a WLAN at providing wireless access to its STAs. First, the aggregate throughput (also known as system throughput) represents the sum of the throughputs of all individual STAs in the WLAN. Second, the fairness in the distribution of access to the radio channel among STAs is another critical factor. Measures of fairness such as Jain's index or proportional fairness (PF), based on the individual throughputs of all STAs, are common means to determine whether certain STAs are receiving a disproportionate share of the radio resource at the expense of other STAs. Indeed, certain STAs may struggle to access the radio channel due to an unfavorable location in the conflict graph. These STAs are said to be in starvation of throughput and they represent a major issue for network administrators. In this paper, a STA is considered to be starving if it cannot obtain at least a given percentage α, say 10%, of the throughput they would have in the absence of other STAs. Third, the frame error rate (FER) of each STA, which indicates the percentage of frames lost due to collisions and poor channel condition, can also be worth of interest to network administrators. As discussed earlier in this section, the setting of `TX_PWR` and `OBSS/PD` on each AP can significantly change these performance metrics. For instance, Figure 1 shows the conflict graph associated with a WLAN for two different parameter settings. While the default setting (see Figure 1) leads to an aggregate throughput of 600Mbps, a Jain's index of 0.42, and a number of starving STAs at 8, a more appropriate setting of these parameters (illustrated again by Figure 1) can

shift these values to 900Mbps, 0.55, and 2, respectively.

4. Proposed solution

In our solution, we search for a correct network configuration using an agent located in the network controller. We assume that regular reports on the individual throughputs of STAs are forwarded to the WLAN controller, which implements a MAB approach. In this framework, each network configuration c_i ($i > 1$) represents an arm that the agent can pull (thus performing a trial) to obtain a reward r_i^k with k designating the trial index. Note that the rewards of an arm i are drawn from a probability distribution $\mathcal{D}(\theta_i)$ whose parameters θ_i are unknown from the agent but invariant in time since the network under study is assumed to be stationary. In this section, we present an efficient strategy $\pi(k)$, which determines which arm to pull (WLAN configuration to test) at each trial (time step) k.
Given the exponential growth of the cardinality of the configuration space C with the number N_A of APs (bounded by 21^{2N_A} since an AP has two tunable parameters of 21 values each), our problem is more precisely framed as an Infinitely Many-Armed Bandit (IMAB). Thus, in practice, the network controller cannot explore the whole set of arms in a reasonable amount of time and must instead work on a subset of C, referred to as the reservoir. In fact, the optimal arm is likely to not even be considered during the search process.
Table 1 summarizes the principal notation used in our proposed solution together with their numerical values chosen empirically for our simulations.

4.1. Reward function

In a Reinforcement Learning (RL) problem, the choice of the reward function is a critical step and its definition can deeply influence the outcome of the optimization process. In the case of a WLAN, the reward function aims at quantifying the quality of a network configuration. However, as discussed in Section 3, there are several performance metrics to assess the quality of a WLAN, and thus different ways of combining them. From the standpoint of a network administrator, a WLAN configuration is considered favorable if it ensures a fair share of throughput among the APs and STAs. More precisely, we enumerate, by order of importance, three criteria to take into account: (i) the number of STAs that are starving for throughput should

Table 1: Principal notation for the proposed method and their corresponding values in the simulations.

Parameter	**Value**	**Description**
C	Relative to topology	Configuration space
N_A	Relative to topology	Number of APs
N_S	Relative to topology	Number of STAs
T_i	Relative to topology	Throughput of STA i
T_i^A	Relative to topology	Attainable throughput of STA i
T^-	Relative to topology	STAs in starvation situation
T^+	Relative to topology	STAs not in starvation situation
α	0.1	Starvation threshold parameter
ϵ	0.1	Exploration rate for strategies
n	2	Sample size in Algorithm 1
K	6	Number of Gaussians in Algorithm 2
δ	$\frac{1}{1+N_s}$	Hypothesis parameter in Algorithm 2

be minimized, (ii) the fairness between STAs should be maximized, (iii) the aggregate throughput of the network should be maximized.
Satisfying both criteria (ii) and (iii) at the same time is challenging. Indeed, in most topologies, increasing fairness between throughputs of STAs is made at the expense of a lower aggregate throughput. Conversely, increasing the network aggregate throughput often implies a decrease in fairness. In order to reach a natural trade-off between those two metrics, we build our reward function on a normalized version of the proportional fairness (PF) of the station throughputs as given by Equation 2. Note that in Equation 2, the STAs' throughputs $T_i, i \in [1, N_S]$ are normalized by their attainable throughputs $T_i^A, i \in [1, N_S]$. T_i^A is simply defined as the throughput that the STA i would obtain in the absence of all other stations. Hence, T_i^A can be seen as an upper bound for T_i. Then, the normalized throughputs are multiplied with each other to obtain a ratio belonging to $[0, 1]$ that represents the quality of the compromise found between criteria (ii) and (iii).

$$\mathrm{PF}(T, T^A) = \prod_{i=1}^{N_S} \frac{T_i}{T_i^A} \tag{2}$$

We account for criterion (i) by ensuring that any network configuration with a higher number of stations in starvation situations than another configuration obtains a lower reward value. In our case, we consider that STA i is starving for throughput whenever its throughput T_i is less than a given fraction, denoted by $\alpha \in [0, 1]$, of its attainable throughput T_i^A. The value of α reflects the desired level of service for the WLAN and its STAs. Given the high expectation of end-users on WLANs performance, a network administrator should probably select α somewhere in the range [0.05 to 0.25]. For our numerical results, we choose $\alpha = 0.1$ but, for the sake of completeness, we study the impact of the value of α on the quality of the optimization at the end of Section 5. With $\alpha = 0.1$, a station having a throughput less than 10% of its attainable throughput is said to be in starvation. Having defined the notion of starvation, we regroup STAs in starvation (those whose $T_i < \alpha T_i^A$) in a set T^- while the others are placed in a set T^+. Then, we compute the PF for each subset T^+ and T^-, we normalize them using their upper bounds (αT_j^A for T^- and T_j^A for T^+), and we combine them using Equation 3 to obtain our reward function. A centralized entity having access to the throughput of each of the N_S STAs can easily compute Equation 3 in $O(N_S)$ operations. Note that this definition forces our reward to evolve in disjoint intervals in $[0, 1]$, the selected one depending on the number of STAs in starvation, as depicted by Figure 2.

$$r_i^k = \frac{|T^-| \prod_{j \in |T^-|} \frac{T_j^-}{\alpha T_j^A} + |T^+| \left(N_S + \prod_{j \in |T^+|} \frac{T_j^+}{T_j^A} \right)}{N_S(N_S + 1)} \tag{3}$$

To measure the quality of a given strategy $\pi(k)$, we use the cumulative regret, which is the standard metric used in MAB problems. With μ^* denoting the best expected reward (i.e., $\mu^* = \max_c \mathbb{E}[r|c]$), the cumulative regret $R_n(\pi)$ on strategy π after n actions (or trials) taken by the agent, is defined by Equation 4.

$$R_n(\pi) = n\mu^* - \sum_{k=1}^{n} r_{\pi(k)}^k \tag{4}$$

A common practice to circumvent the infinite number of arms in an IMAB problem consists of restricting the exploration to a limited subset of solutions composed of random arms that constitute the reservoir. Typically, the selected arms are drawn uniformly from the whole set of arms (e.g., [26],

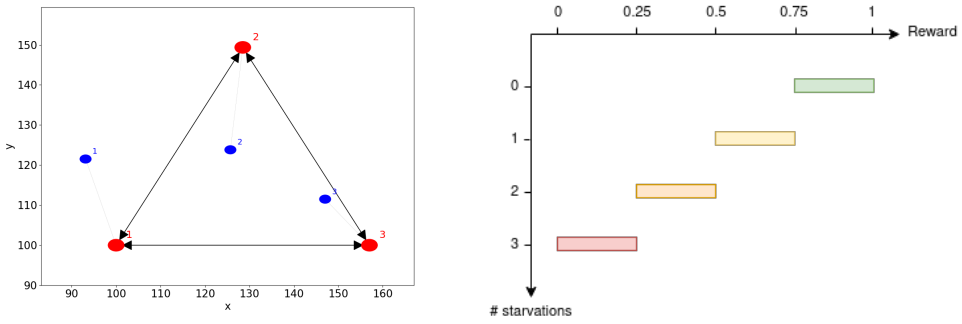

Figure 2: On the left, a simple topology composed of 3 APs and 3 STAs. On the right, the corresponding reward intervals, containing increasing values as the number of STAs in starvation decreases.

[27], [28]). In our case, this approach is not suitable as the vast majority of network configurations lead to poor solutions so that the reservoir would therefore consist most likely only of unsuitable solutions. However, unlike a typical IMAB problem, in which no hypothesis can be made on the relationship between the arms and their rewards, in our case, two neighbor network configurations are likely to have similar rewards. To exploit this similarity between neighbor configurations, we consider our configuration space as a normed space with $||\cdot||$ the L1-norm and we assume that the property given by Equation 5, which relates the spaces of arms and rewards, is verified. Although not always true, this property enables us to leverage the information collected on former trials to guide the sampling of new configurations, mostly in the neighborhood of already good configurations.

$$\forall c_i, c_j \in C, \exists \delta > 0, ||c_i - c_j|| = 1 \implies |r_i^{\cdot} - r_j^{\cdot}| < \delta \tag{5}$$

Therefore, our problem breaks down into two subproblems that must be solved concurrently: (i) sampling promising configurations, and (ii) identifying the best arm among those sampled and pulling it as much as possible. An agent called the sampler is in charge of the first task while another agent known as the optimizer accounts for the second task. Figure 3 summarizes the main principles of our solution. The remainder this section is devoted to the definition of the optimizer and the sampler agents.

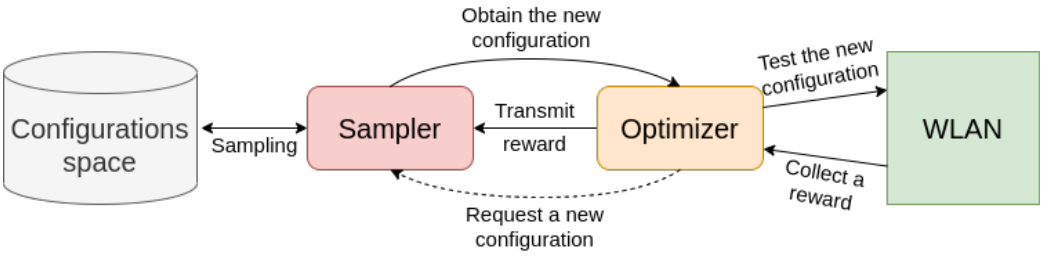

Figure 3: Outline of our solution: The optimizer requests a new configuration to the sampler, which selects and returns it to the optimizer. The optimizer tests this configuration on the real environment, obtains a reward in return, and forwards this reward to the sampler so that both agents can update their internal state.

4.2. Optimizer

The role of the optimizer is to quickly identify the best network configuration (namely $\text{argmax}_c \mathbb{E}[r|c]$) among the current reservoir of network configurations and to use it most often. In our case, the reservoir is initialized with the default configuration of APs and progressively filled with new configurations proposed by the sampler. Algorithm 1 describes the behavior of the optimizer agent. The algorithm has two main parameters: the exploration rate ϵ, which decides how often configurations out of the reservoir are tested, and the sample size n, which determines how often reward estimates of a given configuration are updated.

Algorithm 1 is based on Thompson Sampling (TS) [29], which achieves an optimal regret bound [30, 31] and derives from Bayesian principles. In the previous works [18, 19] wherein TS was used for the sake of spatial reuse of WLANs, the authors assume that $(r_i|c_i) \sim \mathcal{N}(\mu_i, 1)$, with a Gaussian prior for μ_i initialized at $\mathcal{N}(0, 1)$. Then, the prior of μ_i at step k is given by $\mathcal{N}\left(\hat{\mu}_i^k, \frac{1}{n_i^k+1}\right)$, where $\hat{\mu}_i^k = \frac{\sum_{w=1:i(w)=i}^{k-1} r_i^k}{n_i^k+1}$ ([19]) and n_i^k denotes the number of times the configuration c_i has been tested after k steps. In our solution, we make no assumptions regarding the actual value of the variance of r_i. Similarly to the mean value, the variance is progressively estimated. We assume $(r_i|c_i) \sim \mathcal{N}(\mu_i, \sigma_i^2)$ and choose normal-gamma priors for both the mean μ_i and the precision σ_i^{-2}: $\left(\mu_i, \sigma_i^{-2}\right) \sim \text{NormalGamma}\left(\hat{\mu}_i^k, \hat{\lambda}_i^k, \hat{\alpha}_i^k, \hat{\beta}_i^k\right)$. Note that a normal-gamma distribution implies that the precision (inverse to the variance) has a Gamma distribution and that the mean, once the precision is known, has a Normal distribution. Therefore, the priors at step k for μ_i are

Algorithm 1 Optimizer algorithm

Input: sample size n, exploration rate ϵ

Init reservoir E with $\emptyset$
Init step counter k with 0
loop
  if $E = \emptyset$ **or** rand() $< \epsilon$ **then**
    Get a new configuration c_i using the sampler
    Test c_i n times on the environment and collect rewards in X_i
    $k \leftarrow k + n$
    $\left(\mu_i^k, \lambda_i^k, \alpha_i^k, \beta_i^k\right) \leftarrow \left(\bar{X}_i, n, \frac{n}{2}, \frac{nVar(X_i)}{2}\right)$
    $X_i \leftarrow \emptyset$
    Add c_i to reservoir E
  else
    for c_i **in** E **do**
      Sample g_i from $\Gamma\left(\alpha_i^k, \beta_i^k\right)$
      Sample μ_i from $\mathcal{N}\left(\mu_i^k, \left(\lambda_i^k g_i\right)^{-1}\right)$
    end for
    $j \leftarrow \text{argmax}_i\, \mu_i$
    Test c_j on the environment and add reward to X_j
    $k \leftarrow k + 1$
    if $|X_j| = n$ **then**
      Update prior parameters $\left(\mu_j^k, \lambda_j^k, \alpha_j^k, \beta_j^k\right)$ according to Equation 6
      $X_j \leftarrow \emptyset$
    end if
  end if
  Send tests and rewards to the sampler algorithm
end loop

given by $(\mu_i|G) \sim \mathcal{N}\left(\hat{\mu}_i^k, \left(\hat{\lambda}_i^k G\right)^{-1}\right)$, where $G \sim \Gamma\left(\hat{\alpha}_i^k, \hat{\beta}_i^k\right)$. For a sample X_i of size n, mean $\bar{x}_i$, and variance s_i, standard calculations demonstrate that the posterior distribution is $\left(\mu_i, \sigma_i^{-2}|X_i\right) \sim \text{NormalGamma}\left(\hat{\mu}_i^{k+1}, \hat{\lambda}_i^{k+1}, \hat{\alpha}_i^{k+1}, \hat{\beta}_i^{k+1}\right)$, with Equation 6 describing how to update our parameters. These updates enable our optimizer to incorporate new measures on the network configuration into its reward estimations.

$$\begin{pmatrix} \hat{\mu}_i^{k+1} \\ \hat{\lambda}_i^{k+1} \\ \hat{\alpha}_i^{k+1} \\ \hat{\beta}_i^{k+1} \end{pmatrix} = \begin{pmatrix} \frac{\hat{\lambda}_i^k \hat{\mu}_i^k + n\bar{x}_i}{\hat{\lambda}_i^k + n} \\ \hat{\lambda}_i^k + n \\ \hat{\alpha}_i^k + \frac{n}{2} \\ \hat{\beta}_i^k + \frac{1}{2}\left(ns_i + \frac{\hat{\lambda}_i^k n\left(\bar{x}_i - \hat{\mu}_i^k\right)^2}{\hat{\lambda}_i^k + n}\right) \end{pmatrix} \tag{6}$$

Algorithm 1 has a complexity growing linearly with the number of configurations in the reservoir E. As a matter of fact, for each configuration in the reservoir, the agent must sample a Normal-Gamma distribution to find the configuration to test. Since the number of configurations in the reservoir is proportional to the current optimization step k, its computational complexity is $O(k)$.

4.3. Sampler

The role of the sampler is to explore the configuration space C, and to yield promising configurations when requested to by the optimizer. The exploration process is given by Algorithm 2.

To efficiently sample new configurations in this high-dimensional space, we build our sample distribution as a normalized sum of Gaussian distributions, which is known as a Gaussian Mixture (GM). Algorithm 2 constructs and updates this GM. Unlike uniform sampling, a GM-based sampling whose Gaussians are centered on the best-known configurations ensures that most of the new sampled configurations are located in the vicinity of the currently best-known configurations.

We allow a total of K Gaussian distributions in the mixture and we propose to define their centers as the K best configurations discovered so far, whose associated rewards are denoted by $r_1 \geq \cdots \geq r_K$. To sample in every direction without distinction, their covariance matrices will be scalar: $\Sigma_i = \lambda_i I$, $\lambda_i \in \mathbb{R}^+$. In order to find an adequate value of λ_i, we consider the hypothesis made on Equation 5. If Equation 5 is true, then $\forall c_i, c_j \in C, \exists \delta >$

Algorithm 2 Sampler algorithm

Input: K number of Gaussians, δ target parameter

if first call **then**
  Init G with $\left\{\left(\left(-82, 20, \cdots, -82, 20\right), \left(\frac{1}{\dim C}\right)^2 I\right)\right\}$
  Init weights W with $\{1\}$
  Init history H with $\emptyset$
  Init tests counter k with 0
else
  Retrieve previously built G, W, H and k
  Add pairs (conf, rew) transmitted by the optimizer
end if
Sample a new configuration c from mixture (G, W)
Transmit c to the optimizer
$k \leftarrow k + 1$
if $k = \sum_{(\mu_i, \lambda_i I) \in G} \lambda_i \dim \mu_i$ **then**
  Reset G and W
  Find K (c_i, r_i) pairs in H with largest rewards
  $target \leftarrow \delta + \max_j r_j$
  for $i \leftarrow 0$ **to** K **do**
    Add $\left\{\left(c_i, \left(\frac{target - r_i}{\delta \dim C}\right)^2 I\right)\right\}$ to G
    Add r_i to W
  end for
end if

$0, ||c_i - c_j|| = x \implies |r_i^{\cdot} - r_j^{\cdot}| < x\delta$. Targeting a new configuration with a reward of $r_1 + \delta$ for the next sample, and considering the i-th Gaussian centered on c_i with an average reward of r_i, we need to sample a new configuration c so that $||c_i - c|| \geq \frac{r_1+\delta-r_i}{\delta}$. One way to sample configurations which are, on average, away from c_i by this distance is to set $\lambda_i = \left(\frac{r_1+\delta-r_i}{\dim C * \delta}\right)^2$. Thus, parameterized by K and δ, our sampling strategy defines a mixture of K Gaussians centered on the K best configurations discovered so far; the i-th Gaussian being defined by $\mathcal{N}\left(c_i, \left(\frac{r_1+\delta-r_i}{\dim C * \delta}\right)^2 I\right)$.

The complexity of Algorithm 2 depends on the number of Gaussian distributions in the mixture K and the number of dimensions D of the WLAN configurations. Since the agent needs to chose which one of the K Gaussian distributions it will sample in a D-dimensional space, the complexity is $O(K+D)$. Note that $D = 2N_A$ since an AP is configured by two independent parameters.

Figure 4 illustrates a possible execution of Algorithm 2. For visualization purposes, we limited the configuration space to only two dimensions, which correspond to `TX_PWR` and `OBSS_PD` in a case of a WLAN made of a single AP. In practice, the algorithm is executed in a search space with many more dimensions. With the first snapshot, we can note that the mixture is initialized at the default configuration of 802.11, namely `TX_PWR` = 20 dBm and `OBSS_PD` = -82 dBm for all APs. With the two next snapshots, we remark that after a few iterations of Algorithm 2, the mixture has moved in the configuration space, sampling promising configurations along its way. Eventually, in the last snapshot, we can see that the whole sampling density is concentrated in the rectangle $[-80, -75] \times [4, 8]$, where sampled configurations reach large reward values.

5. Numerical results

5.1. Experimental settings

To evaluate the efficiency of our solution at improving the spatial reuse of a WLAN, we implemented it in the realistic discrete-event simulator ns-3 [32] and explored its performance against those of other existing strategies. The ns-3 code implementing our solution, the other strategies, as well as the considered topologies are available for download at [33].

In addition to our solution described in Section 4 and denoted by `OURS` in the following, we consider four other strategies that we also implemented

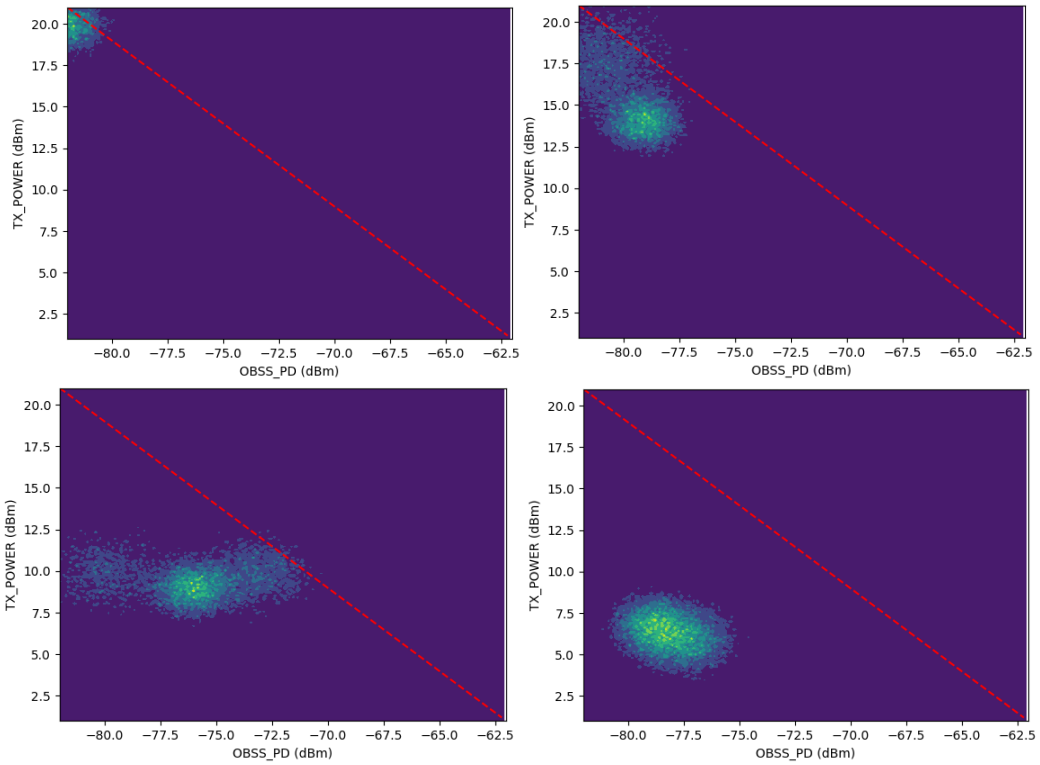

Figure 4: Four snapshots illustrating a simple execution of the sampler, with the sampling distribution density shown in colors. The frontier between authorized configurations (*i.e.* satisfying Equation 1) and unauthorized configurations is shown with a red-dashed line. The region above this line is unauthorized by the IEEE 802.11ax amendment.

in the simulator ns-3. First, we include the legacy default configuration of 802.11, which configures each AP with a sensibility threshold (`OBSS_PD`) of -82 dBm and a transmission power (`TX_PWR`) of 20 dBm. We use `DEFAULT` to refer to this strategy. Then, we consider the classical ϵ-greedy strategy [34], which, at each step, either tests a random configuration with probability ϵ, or chooses the best configuration so far with probability $1-\epsilon$. In the remainder of this section, we use ϵ`-GREEDY` to denote this simple strategy and we use $\epsilon = 0.1$. Next, we implement a solution based on Thompson Sampling but using the priors proposed by [18]. We use `TS` to refer to this solution. Note that the authors of [18] proposed the use of `TS` in a different technological context than ours (distributed version) nullifying any comparison beyond our context. Eventually, the fourth strategy is a modified version of `TS` wherein

the sampler is replaced with ours based on Gaussian Mixture (Section 4.3). We refer to this last strategy as `GM-TS`. Recall that ϵ-`GREEDY` and `TS` strategies are using uniform sampling to discover new configurations. Therefore, comparing `TS` and `GM-TS` allows us to quantify the benefits brought by our sampling algorithm while comparing `GM-TS` and our solution (including our sampler and our optimizer and denoted by `OURS`) highlights the benefits of our optimizer. Table 1 indicates the parameter values used for all the considered strategies.

In each of our experiments, the simulation runs last for a total of 120 seconds of simulated time. For the sake of accuracy, each simulation was replicated with 25 independent repetitions. The duration of a test, corresponding to the length of a trial for our solution, was set to 50 msec. Therefore, 2,400 optimization steps can be performed before the simulation ends. Because we replicate 25 times each simulation, we obtain a matrix of 25x2,400 measures for each network performance metric. In order to provide a clear visualization of this large set of data, we chose to plot the median of the metric at each optimization step, framed by its first and third quartiles. Finally, we applied an exponential moving average (EMA) with a parameter of 0.04 to the three considered quartiles, so we can notice the trends caused by the optimization. This kind of visualization gives us an insight not only into the final performance of each strategy but also on its performance during the whole optimization process. The remainder of the ns-3 simulation parameters is given by Table 2. For the sake of comparison, all strategies are evaluated using the same simulation parameters as well as the same reward function.

Table 2: ns-3 parameters.

Parameter	**Value**
ns-3 version	3.31
Number of repetitions	25
Simulation duration	120 s
Test duration	50 ms
Packet size	1,464 Bytes
Frequency band	5 GHz
A-MDPU Aggregation	4
Path loss	`LogDistancePropagationLossModel`
MCS Control	VhtMcs0
MCS Data	VhtMcs4

We present three examples out of the many we investigated corresponding to the network topologies **T1**, **T2** and **T3** depicted in Figure 5. Each of them

may correspond to a typical dense WLAN deployment. Topologies **T1** and **T2** are both composed of 6 APs, each being associated with two or three STAs. As for the topology **T3**, it is composed of 10 APs and 25 STAs. **T3** is particularly dense with an average of 5.6 conflicts per AP (when configured with the default setting of `TX_PWR` and `OBSS/PD`), and will allow us to test our solution on a larger, denser WLAN deployment. Note that the number of APs here refers to APs belonging to the same WLAN and set on the same radio channel. Given the number of independent channels (3 in 2.4GHz and 23 in 5GHz in many countries), topologies like **T1**, **T2** and **T3** could actually correspond to WLANs comprising dozens of APs.

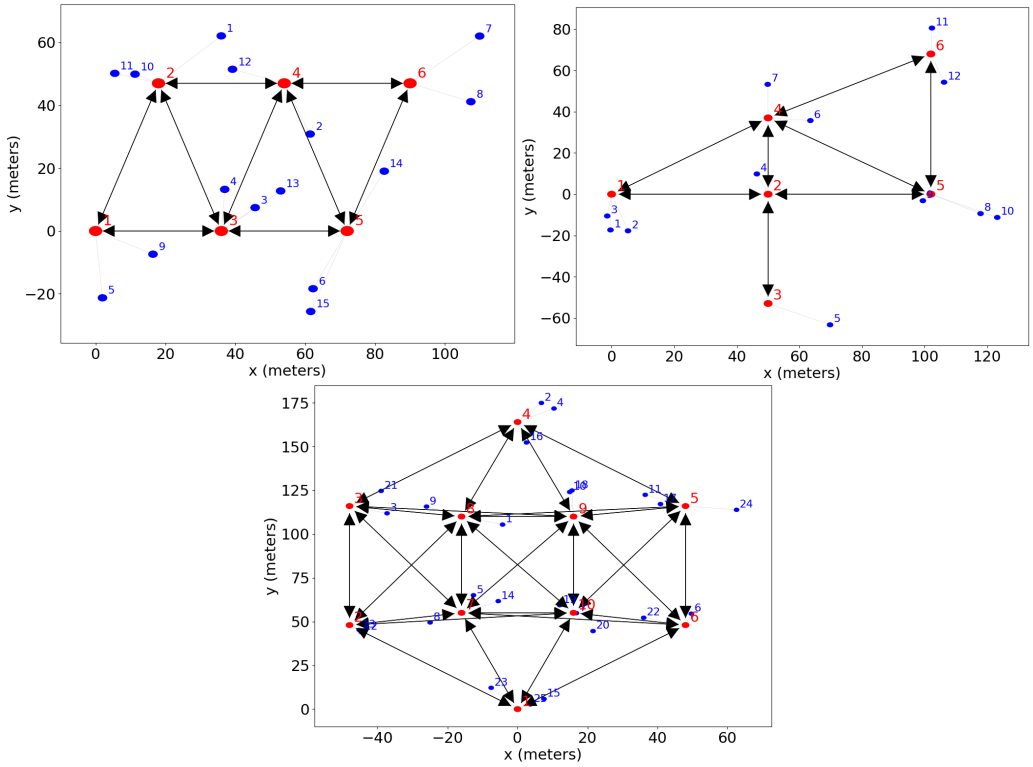

Figure 5: The three topologies **T1**, **T2** and **T3** used in the evaluation of our proposed method. The APs are shown with red circles while the conflicts between APs can be seen with black two-headed arrows. The STAs are represented with blue circles.

5.2. Simulation results

We start our performance analysis by studying the evolution of the number of starving STAs with each strategy. Recall that starving STAs represent

a major issue for WLANs and an efficient WLAN configuration should be able to remove as many starving STAs as possible. Figure 6 shows the corresponding results delivered by the simulator ns-3. We notice that with `DEFAULT` (*i.e.* with the default setting of `TX_PWR` and `OBSS/PD`), the number of STAs in starvation is in average at 8 for **T1**, 7 for **T2** and 15 for **T3**. All the other strategies manage to rapidly reduce the number of starving STAs across the three examples except for `TS` that consistently obtains worse values than `DEFAULT`. The results also show that `GM-TS` significantly outperforms `TS` suggesting the importance of the sampling process in the overall optimization. Finally, Figure 6 indicates that our solution, denoted by `OURS`, leads to the removal of a proportion of starving STAs, which goes up to 40% when compared to `DEFAULT` on **T3**.

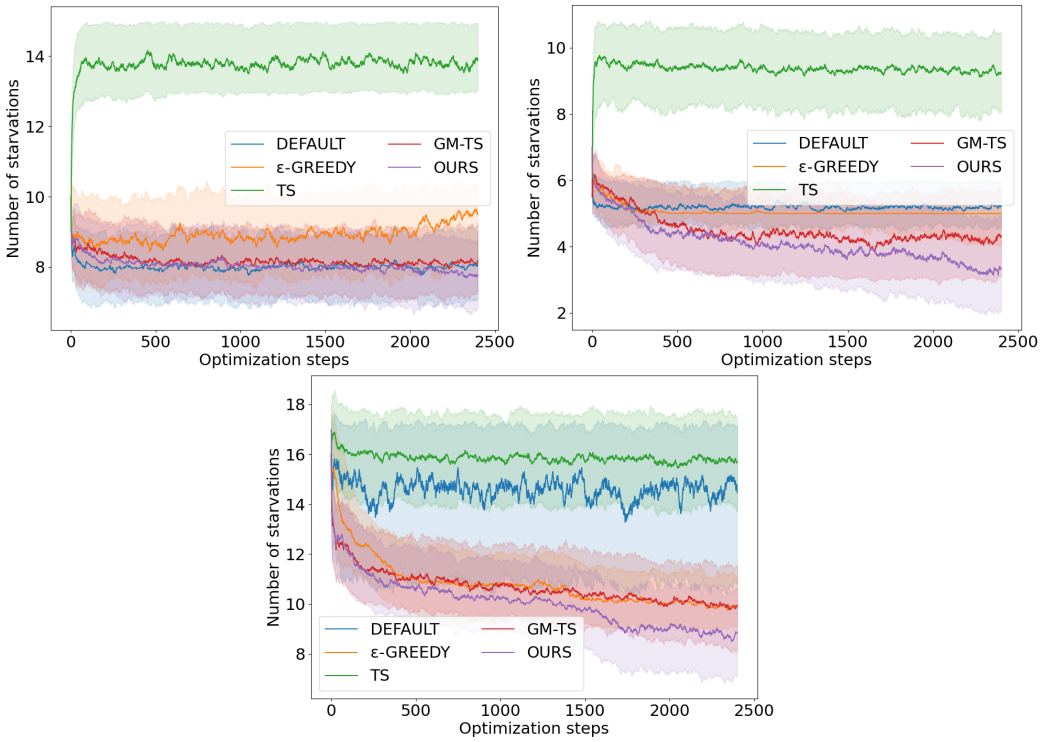

Figure 6: Evolution of the number of STAs starving of throughput for the five considered strategies on **T1**, **T2** and **T3**.

To further illustrate the gain that a better setting of the `TX_PWR` and `OBSS/PD` parameter values can have on the network, we represent in Figure 7 the throughputs of each STA for both the default configuration of 802.11ax and

the one found by our solution on **T2**. Figure 7 shows that all STAs achieve higher throughputs when using the configuration found by our solution. More importantly, as pointed by Figure 7, our solution enables most STAs to operate above the starvation threshold and only 3 of them (STA 4, STA 8 and STA 9) are occasionally experiencing starvation of throughput. Conversely, Figure 7 shows that, in the case of the default configuration, most STAs are at least periodically experiencing starvation of throughput. Similar results (not presented in this paper) were obtained for **T1** and **T3**.

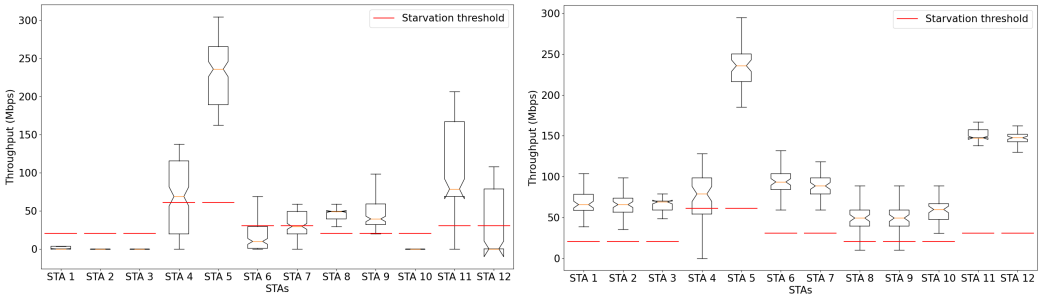

Figure 7: Throughputs obtained by STAs under the default 802.11ax configuration (left) and the configuration found by our solution (right) on **T2**. Each STA throughput distribution is shown as a boxplot, with a red horizontal bar designating the starvation threshold: if the throughput is below this bar, the STA is considered as starving.

We now explore the influence of our solution over the fairness that reflects how uniformly the throughputs are distributed among the STAs. For that purpose, we use Jain's index that tends to be negatively correlated to the number of STAs in starvation in the network. Figure 8 represents the corresponding results for each topology. We observe that our solution leads to a quick increase of the fairness during the search by 20 to 40 points when compared to `DEFAULT` and brings a substantial gain from the fairness associated with ϵ-`GREEDY` or `TS`.

For the sake of completeness, we study the influence of all strategies over the aggregate throughput, defined as the sum of the STAs throughputs (see Section 3). Figure 9 reports the corresponding results. We observe that out of the 5 considered strategies, ϵ-`GREEDY` is the one that leads to the largest improvement in terms of aggregate throughput. ϵ-`GREEDY` performs respectively around 14% and 16% better than our method on the topologies **T2** and **T3** while attaining similar values for **T1**. However, keep in mind that maximizing the aggregate throughput is only a secondary objective in

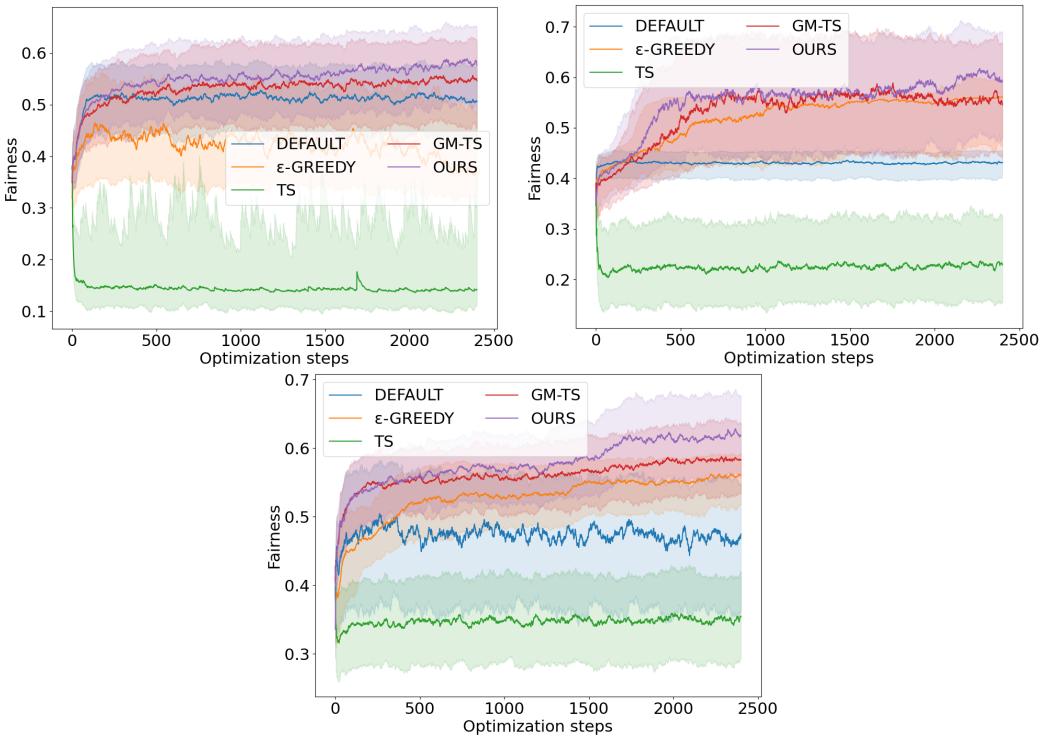

Figure 8: Evolution of fairness between the STAs throughputs for the five considered strategies on **T1**, **T2** and **T3**.

a WLAN and that it is often done at the expense of fairness and the number of starving STAs (see Figures 6 and 8). Figure 9 shows that for topology **T1** our proposed solution maintains the aggregate throughput near its value obtained with the default setting of `TX_PWR` and `OBSS/PD`. For topologies **T2** and **T3**, our solution was able to significantly increase the aggregate throughput when compared to `DEFAULT`. Overall, these results indicate that there is no downside for the aggregate throughput to the significant benefits brought by our solution.

We use Figure 10 to visualize how each strategy performed, in the sense of our reward function, at each optimization step. First, as expected, the value of the reward is negatively correlated with the number of starving STAs depicted in Figure 6. This results from the prominent role of the number of starvations in our definition of the reward (see Equation 3). We also observe that collecting a higher reward generally results in higher levels of fairness and aggregate throughput (see Figures 8 and 9). Figure 10 also shows that on

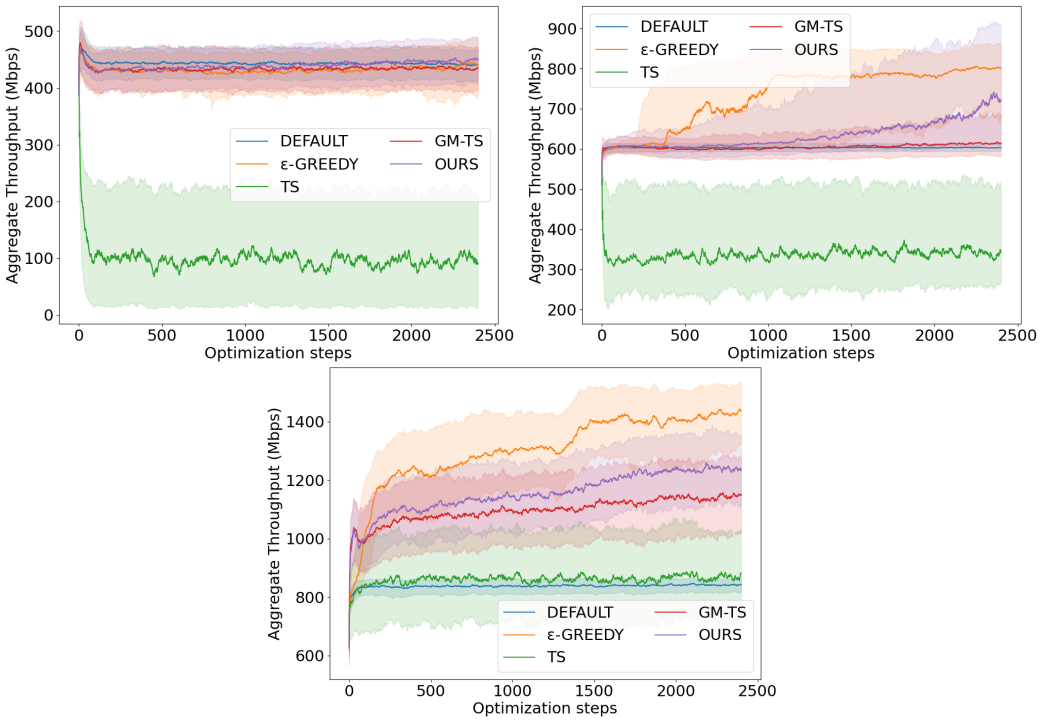

Figure 9: Evolution of the aggregate throughput for the five considered strategies on **T1**, **T2** and **T3**.

every topology and for a large majority of optimization steps, our proposed solution is the one that achieves the best reward value among the considered strategies. Note that this is always the case at the last optimization step.
Lastly, we assess the performance of our solution with regards to the standard measure of quality in MAB problems, namely the cumulative regret, which represents the sum of differences between the maximum reward and the reward obtained at each trial (as defined by Equation 4). Unlike the reward, which gives information about the performance of a strategy at a given time step, the cumulative regret is a measure of quality on the whole simulation. Figure 11 represents the cumulative regret obtained by each of the five strategies across the three topologies. This figure clearly shows that our solution is the one that provides the lowest cumulative regret on each topology during the whole optimization process. Furthermore, comparing the strategies `TS` and `GM-TS` shows the positive influence that the sampler can have while the comparison between the results of `GM-TS` and our solu-

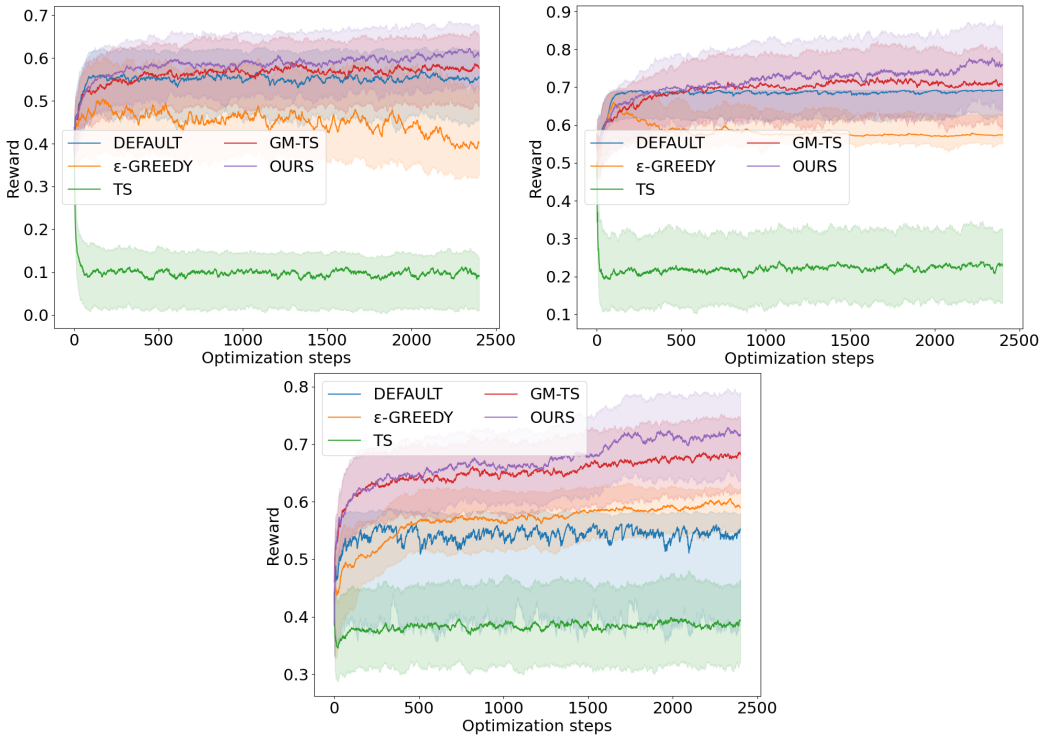

Figure 10: Evolution of the reward obtained by the five considered strategies on **T1**, **T2** and **T3**.

tion points out the importance of the optimizer and of the priors in use in a Thompson Sampling approach.

Within 2,400 iterations, representing 120 seconds of simulated time and a very limited exploration in large state spaces, our solution was always able to significantly reduce the number of starving STAs and to increase fairness between the throughputs of STAs without decreasing the aggregate throughput of the WLAN. Note that better results may be achieved with longer simulations. Those improvements are obtained in less than 900 iterations representing 45 seconds for the smaller topologies **T1** and **T2**. Overall, our solution consistently brings a significant improvement on every performance metrics when compared to the legacy default configuration of 802.11. It is able to remove 14%, 63% and 73% of the conflicts occurring with the default configuration in topologies **T1**, **T2** and **T3** respectively. We believe that these results demonstrate the capacity of a tailored MAB solution at improving the spatial reuse of radio channels in WLANs.

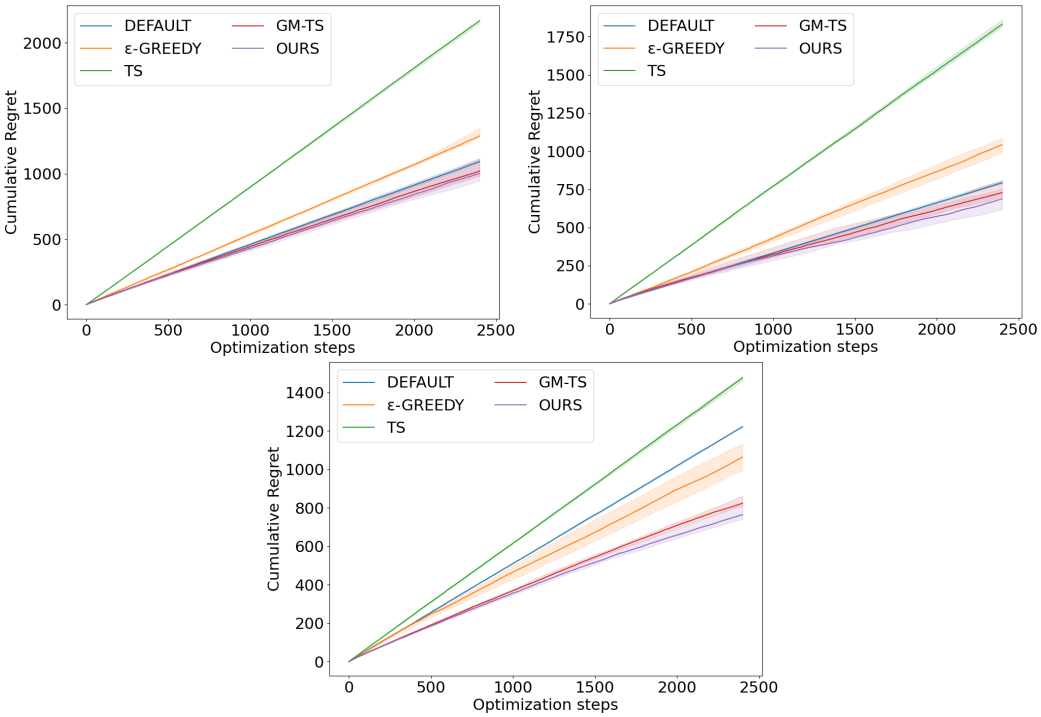

Figure 11: Evolution of the cumulative regrets for the five considered strategies on **T1**, **T2** and **T3**.

5.3. Robustness of the solution

5.3.1. Handling workload fluctuations in the WLAN

In the previous section, we showed that our solution was able to increase the spatial reuse of WLANs by finding an appropriate configuration of `TX_PWR` and `OBSS_PD` when WLANs are facing a high but constant workload. However, the workload within a WLAN may undergo variations with some STAs demanding more or less traffic to be exchanged. Thus, a good configuration of `TX_PWR` and `OBSS_PD` must be robust to these potential workload variations, by consistently bringing positive improvements to key performance metrics, when compared to the default configuration of 802.11.

To study the robustness of our solution, we consider **T3**, our densest topology, and we let its STAs run two different applications with different throughputs: (i) an application requiring as much throughput as the STA can get and (ii) an application requiring 1% of the attainable throughput of the STA. In this section, we designate STAs running application of type (i) as "active"

STAs. By controlling the proportion p of active STAs, we let the workload of the WLAN vary. In order to evaluate the robustness of a strategy, we proceed as follows. First, we deploy the considered strategy on the WLAN with $p = \frac{2}{3}$ for a first run of 90 seconds. At the end of this run, we obtain the configuration x^* recommended by the strategy. Recall that x^* refers to a set of values for `TX_PWR` and `OBSS/PD` at each AP. Then, we study the performance of x^* under different levels of workload: We configure the APs according to x^* and we run simulations on four different scenarios: $p = 0$, $\frac{1}{3}$, $\frac{2}{3}$ and 1. Each of these simulations lasts 30 seconds so that the WLAN configured with x^* can converge to a stable state. At the end of each 30 seconds simulation, the reward, the fairness, and the aggregate throughput of the WLAN are collected.

We study the robustness of three different strategies: (i) `DEFAULT`, the legacy default configuration of 802.11: (`TX_PWR`, `OBSS_PD`) = (20 dBm, -82 dBm) for each AP, (ii) `TS`, the solution proposed by [18] described in the previous section and (iii) `OURS`, the strategy described in this work.

Table 3: Confidence intervals for the reward, the fairness and the aggregate throughput (in Mbps) obtained by `DEFAULT`, `TS` and `OURS` under different proportions of active STAs.

Active STAs	**Metric**	`DEFAULT`	`TS`	`OURS`
$p = 0$	Reward	**0.98 +/- 0.01**	0.82 +/- 0.01	0.97 +/- 0.01
	Fairness	**0.95 +/- 0.01**	0.79 +/- 0.01	0.94 +/- 0.01
	Agg. Through.	**29.04 +/- 0.03**	25.22 +/- 0.15	28.91 +/- 0.17
$p = \frac{1}{3}$	Reward	0.81 +/- 0.04	0.71 +/- 0.03	**0.93 +/- 0.02**
	Fairness	0.26 +/- 0.03	0.23 +/- 0.02	**0.27 +/- 0.04**
	Agg. Through.	562.03 +/- 38.36	678.63 +/- 70.48	**716.18 +/- 80.6**
$p = \frac{2}{3}$	Reward	0.61 +/- 0.04	0.61 +/- 0.02	**0.84 +/- 0.03**
	Fairness	0.37 +/- 0.03	0.38 +/- 0.02	**0.5 +/- 0.04**
	Agg. Through.	725.13 +/- 27.4	1079.35 +/- 65.67	**1138.64 +/- 72.03**
$p = 1$	Reward	0.5 +/- 0.05	0.59 +/- 0.02	**0.86 +/- 0.02**
	Fairness	0.48 +/- 0.04	0.54 +/- 0.01	**0.75 +/- 0.01**
	Agg. Through.	842.56 +/- 10.49	**1510.89 +/- 43.8**	1468.93 +/- 26.32

The confidence intervals on these metrics are shown in Table 3. Each confidence interval has the form a +/- b, with a the center of the interval and b its radius. For each metric and each scenario, the confidence interval with the best center is shown with **bold text**.
With Table 3, we can see how a configuration obtained with $p = \frac{2}{3}$ behaves with different workloads. For the lowest workload considered ($p = 0$), the best configuration is the default configuration of 802.11, with a reward near its maximal value. The configuration recommended by `OURS` is barely inferior with relative differences of -1.0%, -1.1% and -0.4% for the reward, the fairness and the aggregate throughput, respectively. The configuration recommended by `TS` also behaves properly, although the relative differences with the `DEFAULT` are higher. With $p = \frac{1}{3}$, our solution provides better values for each performance metrics considered, although the workload is lower than the one used for the learning of the recommended configuration. When compared to `DEFAULT`, the reward, the fairness and the aggregate throughput are greater by 14.8%, 3.8% and 27.4%, respectively. Although the gap between `DEFAULT` and `TS` is reduced, the former still performs better than the latter for this workload, except on the aggregate throughput. For $p = \frac{2}{3}$, we observe the same dynamics: the relative difference between `OURS` and `DEFAULT` keeps increasing and `TS` seems equivalent to `DEFAULT` in regards to the reward and the fairness, except on the aggregate throughput where `TS` is better. Eventually, even with a higher workload than the one used for the learning of the recommended configuration ($p = 1$), our proposed solution consistently performs better than `DEFAULT` with relative differences of +72.0%, +56.3% and +74.3% for the reward, the fairness and the aggregate throughput, respectively. The configuration recommended by `TS` outperforms `DEFAULT` and provides the best aggregate throughput for this scenario. However, its reward is still significantly lower than the one obtained by `OURS`.
Based on these numerical results, we observe that the configuration recommended by our proposed solution ensures efficient performance metrics even when the WLAN's workload is moved to higher or lower levels than the one used during the learning phase. Overall, `OURS` is found to be more robust than the state-of-the-art solution `TS` and significantly better than the default configuration `DEFAULT`. Once again, we can see that, except for the case $p = 1$, having the best reward means having the best values on the other performance metrics. This suggests that our reward function is a good quality criterion. As a side note, and not surprisingly, we observe through Table 3 that, in general, the higher the workload, the lower the performance met-

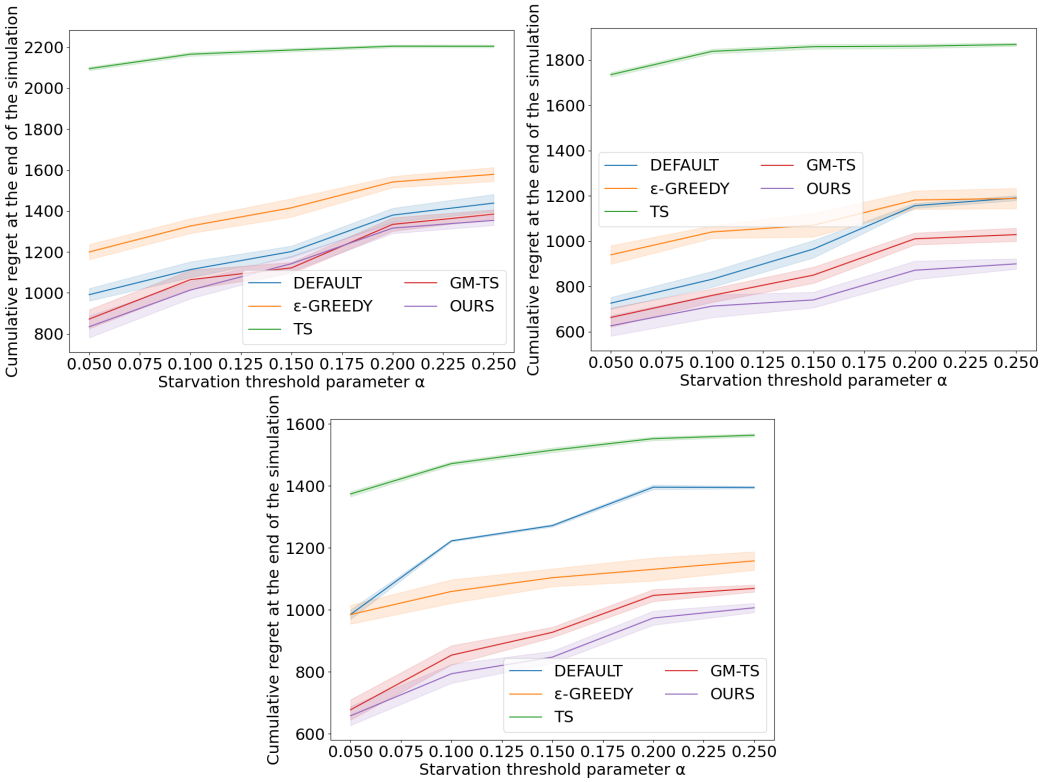

Figure 12: Cumulative regret at the end of the simulation for the five considered strategies on **T1**, **T2** and **T3**.

rics. Therefore, we recommend to perform the learning phase under levels of workload as high as possible for the studied WLAN.

5.3.2. Influence of the starvation threshold parameter, α

Eventually, we study the impact of α, the starvation threshold parameter, on the considered strategies. Recall that the value of this parameter reflects the level of requirements on STAs' performance and is decided by the network administrator. In Figure 12, we show the confidence intervals for the cumulative regret at the end of a 120 seconds lasting simulation on the three considered topologies (**T1**, **T2** and **T3**) for multiple credible values of α. Since the starvation threshold of any STA is proportional to α, the larger α, the more likely STAs are considered in a starvation situation, and ultimately, the more difficult it is for the optimizer to collect large rewards. This explains why the cumulative regret tends to increase for larger values of α as shown

by Figure 12. Overall, we observe that for all the considered topologies and values of α, our proposed solution is the most successful at minimizing the cumulative regret. This suggests that the superiority of our proposed solution is robust to changes in the starvation threshold parameter.

6. Conclusions

We have presented a new solution to improve the spatial radio reuse of 802.11ax-based WLANs by configuring two parameters at each AP, namely their transmission power and their sensitivity threshold. More precisely, we introduced a reward function specifically tailored to our purpose that quantifies the quality of a WLAN configuration. To help the exploration process at discovering promising configurations, we present a new way of sampling the state space that differs from uniform sampling, and of decoupling the search for the best configuration among those discovered so far and the discovery of new promising configurations. The obtained results on ns-3 demonstrate the large potential benefit brought by adapting the transmission power and sensitivity threshold of APs, as well as the ability of our solution to find an adequate configuration. Additionally, we showed that the configuration found by our solution is also robust to workloads variation and consistently brings positive improvements in terms of spatial reuse when compared to the default configuration of 802.11.
A natural follow-up to our work is to implement our solution in a WLAN to test our solution in a real-world environment. Other future works include the extension of our approach to a distributed context making it applicable to WLANs that do not include a network controller.

Acknowledgements

This work was supported by the LABEX MILYON (ANR-10-LABX-0070) of Université de Lyon, within the program "Investissements d'Avenir" (ANR-11-IDEX- 0007) operated by the French National Research Agency (ANR).

The authors wish to thank the anonymous referees for their thorough and constructive review of an earlier version of this paper.

References

[1] Cisco, Cisco annual internet report (2018–2023) white paper, White paper 1 (1) (2018).

[2] IEEE standard for information technology–telecommunications and information exchange between systems - local and metropolitan area networks–specific requirements - part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications, IEEE Std 802.11-2020 (Revision of IEEE Std 802.11-2016) (2021).

[3] A. Bardou, T. Begin, A. Busson, Improving the spatial reuse in ieee 802.11ax wlans: A multi-armed bandit approach, in: Proceedings of the 24th International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM'21), 2021, pp. 135–144.

[4] E. Khorov, A. Kiryanov, A. Lyakhov, G. Bianchi, A tutorial on IEEE 802.11 ax high efficiency WLANs, IEEE Communications Surveys & Tutorials (2018).

[5] G. Anastasi, M. Conti, E. Gregori, A. Passarella, Saving energy in Wi-Fi hotspots through 802.11 PSM: an analytical model, in: 2nd Workshop on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt'04), 2004, pp. 227–236.

[6] B. Badihi, L. F. Del Carpio, P. Amin, A. Larmo, M. Lopez, D. Denteneer, Performance evaluation of IEEE 802.11 ah actuators, in: IEEE 83rd Vehicular Technology Conference (VTC Spring), IEEE, 2016, pp. 1–5.

[7] Y. Daldoul, D.-E. Meddour, A. Ksentini, Performance evaluation of OFDMA and MU-MIMO in 802.11ax networks, Computer Networks 182 (2020) 107477. doi:https://doi.org/10.1016/j.comnet.2020.107477.
URL `https://www.sciencedirect.com/science/article/pii/S1389128620311531`

[8] A. Mishra, V. Brik, S. Banerjee, A. Srinivasan, W. Arbaugh, A client-driven approach for channel management in wireless LANs, in: IEEE International Conference on Computer Communications, 2006.

[9] J. Herzen, R. Merz, P. Thiran, Distributed spectrum assignment for home wlans, in: IEEE International Conference on Computer Communications, 2013.

[10] S. Lee, T. Kim, S. Lee, K. Kim, Y. H. Kim, N. Golmie, Dynamic channel bonding algorithm for densely deployed 802.11ac networks, IEEE Transactions on Communications (2019).

[11] D. Leith, P. Clifford, V. Badarla, D. Malone, Wlan channel selection without communication, Computer Networks (2012).

[12] S. Barrachina-Muñoz, F. Wilhelmi, B. Bellalta, Online primary channel selection for dynamic channel bonding in high-density WLANs, IEEE Wireless Communications Letters (2020).

[13] A. López-Raventós, B. Bellalta, Concurrent decentralized channel allocation and access point selection using Multi-Armed Bandits in Multi BSS WLANs, Computer Networks (2020).

[14] J. Zhu, X. Guo, L. Lily Yang, W. Steven Conner, S. Roy, M. M. Hazra, Adapting physical carrier sensing to maximize spatial reuse in 802.11 mesh networks, Wireless Communications and Mobile Computing (2004).

[15] Y. Kim, J. Yu, S. Choi, SP-TPC: a self-protective energy efficient communication strategy for IEEE 802.11 WLANs, in: IEEE Vehicular Technology Conference (VTC'04), 2004.

[16] T. Ropitault, N. Golmie, ETP algorithm: Increasing spatial reuse in wireless LANs dense environment using ETX, in: IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 2017.

[17] I. Selinis, K. Katsaros, S. Vahid, R. Tafazolli, Control OBSS/PD sensitivity threshold for IEEE 802.11ax BSS color, in: IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 2018.

[18] F. Wilhelmi, S. Barrachina-Muñoz, B. Bellalta, C. Cano, A. Jonsson, G. Neu, Potential and pitfalls of multi-armed bandits for decentralized spatial reuse in wlans, Journal of Network and Computer Applications

127 (2019) 26–42. doi:10.1016/j.jnca.2018.11.006.
URL https://doi.org/10.1016/j.jnca.2018.11.006

[19] F. Wilhelmi, C. Cano, G. Neu, B. Bellalta, A. Jonsson, S. Barrachina-Muñoz, Collaborative spatial reuse in wireless networks via selfish multi-armed bandits, Ad Hoc Networks 88 (2019) 129–141. doi:https://doi.org/10.1016/j.adhoc.2019.01.006.
URL https://www.sciencedirect.com/science/article/pii/S1570870518302646

[20] F. Wilhelmi, J. Hribar, S. F. Yilmaz, E. Ozfatura, K. Ozfatura, O. Yildiz, D. Gündüz, H. Chen, X. Ye, L. You, Y. Shao, P. Dini, B. Bellalta, Federated spatial reuse optimization in next-generation decentralized IEEE 802.11 WLANs (2022). doi:10.48550/ARXIV.2203.10472.
URL https://arxiv.org/abs/2203.10472

[21] M. Gast, 802.11 wireless networks: the definitive guide, O'Reilly Media, Inc., 2005.

[22] M. Gast, 802.11n: the survival guide, O'Reilly Media, Inc., 2012.

[23] Ieee standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks–specific requirements part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications amendment 1: Enhancements for high-efficiency wlan, IEEE Std 802.11ax-2021 (Amendment to IEEE Std 802.11-2020) (2021) 1–767doi:10.1109/IEEESTD.2021.9442429.

[24] M. Garetto, T. Salonidis, E. Knightly, Modeling per-flow throughput and capturing starvation in CSMA multi-hop wireless networks, IEEE/ACM Transactions on Networking (2008).

[25] M. Stojanova, T. Begin, A. Busson, Conflict graph-based model for IEEE 802.11 networks: A divide-and-conquer approach, Performance Evaluation (2019).

[26] Y. Wang, J.-Y. Audibert, R. Munos, Algorithms for infinitely many-armed bandits, in: D. Koller, D. Schuurmans, Y. Bengio, L. Bottou (Eds.), Conference on Neural Information Processing Systems (NeurIPS'09), Vol. 21, Curran Associates, Inc., 2009.

[27] Y. David, N. Shimkin, Infinitely Many-Armed Bandits with Unknown Value Distribution, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD'14), 2014.

[28] M. Aziz, J. Anderton, E. Kaufmann, J. Aslam, Pure exploration in infinitely-armed bandit models with fixed-confidence, in: International Conference on Algorithmic Learning Theory (ALT'18), Vol. 83 of Proceedings of Machine Learning Research, PMLR, 2018, pp. 3–24.

[29] W. R. Thompson, On the Likelihood that One Unknown Probability Exceeds Another in View of the Evidence of Two Samples, Biometrika (1933).

[30] S. Agrawal, G. Navin, Analysis of Thompson Sampling for the Multi-Armed Bandit problem, in: Proceedings of Machine Learning Research, 2012.

[31] S. Agrawal, N. Goyal, Further optimal regret bounds for thompson sampling, in: C. M. Carvalho, P. Ravikumar (Eds.), International Conference on Artificial Intelligence and Statistics (AISTATS'13), Vol. 31 of Proceedings of Machine Learning Research, PMLR, Scottsdale, Arizona, USA, 2013, pp. 99–107.
URL `https://proceedings.mlr.press/v31/agrawal13a.html`

[32] The Network Simulator ns-3, `https://www.nsnam.org/` (2021).

[33] A. Bardou, T. Begin, A. Busson, Online repository for code, `https://github.com/abardou/IMAB-SR-STA-802.11ax` (2021).

[34] R. Sutton, A. Barto, Reinforcement learning: An introduction, MIT press, 2018.