Mitigating Starvation in Dense WLANs: A Multi-Armed Bandit Solution

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

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

Abstract

With the recent 802.11ax amendment to the IEEE standard commercialized as Wi-Fi 6, WLANs have the potential to greatly improve the spatial reuse of radio channels. This resorts to the new ability for APs (Access Points) to dynamically modify their transmission power as well as the signal energy threshold beyond which they consider the radio channel to be free or busy. In general, selecting adequate values for these parameters is complex because of (i) the high dimensionality of the problem and (ii) the uncertainty of the radio environment. To overcome these difficulties, we frame this problem as a MAB (Multi-Armed Bandit) problem and propose an efficient and robust solution using Thompson sampling, an original sampling of WLAN configurations, and a tailor-made reward function. We evaluate the efficiency of our solution as well as several other ones with scenarios inspired by real-life WLANs' deployments using the network simulator ns-3. The numerical results show the ability of our solution along with its superiority over the others at finding adequate parameterization at each AP thereby significantly improving the overall performance of WLANs.

Keywords: WLANs, Spatial Reuse, Fairness, Reinforcement Learning, Thompson Sampling, Power Control, Clear Channel Assessment

1. Introduction

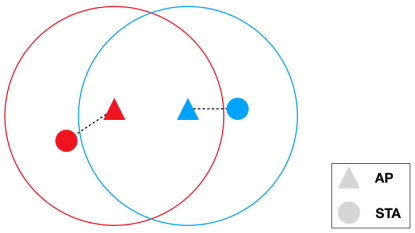
Over the last decade, access to WLANs has gradually come to be regarded as a basic service by many, much in the same way as running water, electricity, and heating. Yet for all of its importance, WLANs rarely run at its maximum efficiency. This is because setting up WLANs is complicated as they need to address the specific requirements of their STAs (Stations) while

accommodating the constraints of their physical environment. In addition, when the area to be connected is large, WLANs are often comprised of a fleet of APs (Access Points), each of them serving as a gateway for its associated STAs. For the sake of practicality, all APs are typically managed by a single controller hosted on a server of the wired network.

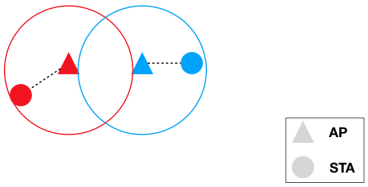
Due to the relative scarcity of space on the radio spectrum and the increasing need of STAs for throughput, current deployments of WLANs tend to contain a dense number of APs. This can have strong implications on the WLANs' performance because of the listen-before-talk principle, which governs the behavior of the 802.11 standard behind the Wi-Fi protocol (e.g., [1]). For instance, if two APs are far enough away from each other (beyond each other's detection range), they can successfully exchange their frames at the same time despite operating on the same radio channel. This is referred to as the "spatial reuse of the channel bandwidth". Conversely, if they are too close (within each other's detection range), as soon as one of them transmits, it automatically blocks (or freezes) the other. While spatial reuse can greatly increase the throughput capacity of a WLAN, it has remained, until now, far from optimal because: (i) APs are fixed and are not easily movable; and (ii) the physical properties of the radio channels are rather unpredictable at the deployment of WLANs' APs.

The recent amendment to the 802.11 standard, namely 802.11ax [2] (the interested reader can refer to [3] for a good tutorial) has the potential to be a game-changer as it enables WLANs to dynamically modify the transmission power of APs (a.k.a. `TX_PWR`), and it allows the signal energy threshold to go beyond the level which APs consider the radio channel to be free or busy (a.k.a. the Overlapping Basic Service Set Packet Detect or `OBSS/PD`). When done properly, the tuning of these two parameters can greatly increase the performance of WLANs (e.g., [4]).

To illustrate the potential of 802.11ax, let us consider a simple scenario where two APs are each serving one STA. First, assume that with their default configuration for `TX_PWR` and `OBSS/PD`, each AP falls in the detection range of the other, as shown in Figure 1a. This makes it virtually impossible for the two APs to transmit simultaneously. By gradually decreasing `TX_PWR` at each AP, we can obtain a configuration where each AP lies out of the other's detection range as depicted in Figure 1b. Now, both can exchange data with



(a) By default, the APs are located in the communication range of each other. Assuming they are on the same radio channel, they cannot transmit simultaneously.



(b) By decreasing the transmission power of each AP, they end up out of the communication range of each other so that, despite being on the same radio channel, they can transmit simultaneously (and successfully).

Figure 1: Illustrating the potential of 802.11ax at improving the spatial reuse of a channel bandwidth.

their STAs at the same time. Another way of doing this is to increase the `OBSS/PD` threshold at each AP so that the energy received at each AP (due to the other's transmissions) becomes insufficient to block its own transmissions. It is worth noting that although these two approaches may be viewed as similar (in the sense that both allow the two APs to transmit simultaneously), the performance of each STA may differ as they will have different RSS (Received Signal Strength). Indeed, changing `TX_PWR` directly affects the value of RSS for the STAs associated with this AP whereas changing `OBSS/PD` affects the AP's opportunism in its transmissions.

In general, selecting adequate values for `TX_PWR` and `OBSS/PD` for a given WLAN is far from trivial. First, according to the 802.11ax amendment [2], each parameter can take 21 values. Values for `TX_PWR` and `OBSS/PD` range from 1 to 21 dBm and from -82 to -62 dBm, respectively. The number of combinations grows exponentially with the number of APs composing the WLAN. Denoting the number of APs by N_A, we have a total of 21^{2*N_A} combinations. Even in the simple case of Figure 1 with only two APs, this leads to 194,481 combinations. Second, the search for adequate values must be conducted quickly since WLANs will be typically maintained in operation during the search process. Third, the radio environment is complex and uncertain. This virtually precludes any optimization approach that relies on a constructive theoretical model to translate surrounding radio properties into high-level performance (for instance STAs' throughput). This is because

most constructive models are either coarse-grained representations that do not permit a fine tuning of the TX_PWR and OBSS/PD parameters (e.g., [5, 6]), or are so detailed that they do not easily scale with the number of APs and STAs (e.g., [7]).

Conversely, data-driven approaches are often designed to deal with large dimensionality and uncertainty, making them natural candidates to address the problem of finding adequate values for the TX_PWR and OBSS/PD. For this study, we chose to frame this problem as a MAB (Multi-Armed Bandit) problem and proposed an efficient and robust solution using Thompson sampling, an original sampling of WLAN configurations, and a tailor-made reward function. More precisely, this paper's contributions are:

- A solution that uses an objective (reward) function suited to the peculiarities of WLANs, a Bayesian optimizer, and an original sampler to determine adequate values for the TX_PWR and OBSS/PD parameters that will improve the spatial reuse in WLANs.
- An evaluation of the efficiency of the above solution with scenarios inspired by real-life WLANs' deployments. We also incorporated important aspects that were ignored in previous works such as the rate adaptation, letting the speed of the wireless links between APs and STAs depend to their actual RSS and the presence of upstream traffic (from STAs to the APs) and downstream traffic, instead of solely downstream traffic. To the best of our knowledge, we are the first to evaluate the efficiency of a solution based on machine learning techniques for the TX_PWR and OBSS/PD problem using the well-established network simulator ns-3 [8].
- A comparison of the performance of our solution with several previously proposed ones and a demonstration of the superiority of ours at finding adequate parameter values for TX_PWR and OBSS/PD at each AP. The simulation results show that our solution is able to quickly discover WLAN configurations that significantly reduce the number of STAs suffering from poor performance, improve the fairness among STAs and increase the cumulated throughput.

The remainder of the paper is organized as follows. The next section discusses the related works. Section 3 describes the objective function in use while

Section 4 presents the optimizer and sampler agents. We relate the numerical results in Section 5. Section 6 concludes this paper.

2. Related Work

For a detailed explanation of how the IEEE 802.11ax standard implements the adaptation of the transmission power (`TX_PWR`) or the clear channel assessment threshold (`OBSS/PD`) for nodes of WLANs, we refer the reader to the recent work of Wilhelmi et al. [4]. The paper also illustrates, through simple scenarios, the benefit of adapting these latter parameters in terms of spatial reuse of the channels and throughput.

Before the adaptation of the `TX_PWR` and `OBSS/PD` parameters was officially enabled by the IEEE 802.11ax standard, some researchers had explored this idea. A case in point is Zhu et al. in [9] who present an analytical model to derive the optimal value for `OBSS/PD` in Wi-Fi-based mesh networks. The selected values for `OBSS/PD` are dynamically tuned on each node depending on the current conditions of the radio channel. In [10], Kim et al. show how adapting the `TX_PWR` parameter of nodes can increase the throughput of nodes while reducing the energy consumption of communications.

In 2020, Qiu et al. [11] cast the issues of AP positioning and their power allocation as a single optimization problem. Their solution addresses the initial positioning of APs, the channel allocation, and the configuration of the parameter `TX_PWR` on each AP. However, it delivers a static configuration for the sole `TX_PWR` parameter of every AP (the threshold parameter `OBSS/PD` is not considered) that does not account for the number of STAs, nor for the amount of traffic exchanged between STAs and APs.

More practical approaches have been proposed to adapt the `TX_PWR` and `OBSS/PD` parameters in real time. Afaqui et al. in [12] studied the problem of configuring the `OBSS/PD` parameter on the STAs. They proposed a distributed algorithm, called Dynamic Sensitive Control algorithm (DSC), in which STAs regularly measure the RSS of the beacons they receive from their associated AP. Recall that beacons are small messages that APs periodically broadcast to advertise their WLAN. In a nutshell, with DSC, each STA gradually decreases its value of `OBSS/PD` to favor concurrent transmissions from nearby nodes (APs or STAs) while keeping this value high enough to

ensure the proper reception of beacons. In [13], Lee et al. proposed another distributed algorithm referred to as Link-aware Spatial Reuse (LSR). With LSR, each AP chooses a concurrent AP that will be allowed to transmit simultaneously and then prescribes the value of its `TX_PWR`. Like DSC, these choices are based on the RSS. More precisely, each AP chooses its concurrent AP as the one having the lowest RSS while the prescribed value for `TX_PWR` is based on the frame error rate when the AP and its concurrent AP transmit simultaneously. Overall, both DSC and LSR algorithms [12, 13] represent practical and effective solutions that can typically increase the throughput of a WLAN by a factor of up 20-30%. However, none of them fully exploit the potential benefits offered by the 802.11ax standard. DSC only modifies the value of the `OBSS/PD` parameter for STAs so that performance improvements mostly apply to the small portion of traffic that goes upstream. In the case of LSR, the solution applies to APs and downstream traffic, but LSR limits to one the number of concurrent APs allowed to transmit simultaneously. In practice, this limitation is further increased since concurrent APs do not always have frames to send at once.

Several researchers made use of machine learning methods to address the issue of tuning the `TX_PWR` and `OBSS/PD` parameters in WLANs. Ak and Canberk in [14] propose a two-scale solution that relies on the learning capabilities of ANN (Artificial Neural Networks). In their framework, STAs and APs first locally adjust their value of `OBSS/PD` to decrease interference. Then, thanks to an ANN whose parameters have been set thanks to offline simulations, they mitigate potential unfairness among STAs in terms of attained throughput, which may otherwise occur due to the various locations of STAs. The authors use the ns-3 simulator to show that their solution can improve the throughput and fairness of WLANs up to 36% and 82%, respectively. As far as we know, there is no performance dataset recognized by the networking community whose content captures the large diversity of scenarios that can be encountered in WLANs. Because of this lack of representative dataset, the predefined parameters of the ANN may not be accurate for all WLAN topologies. Therefore, an online approach, where the agent sequentially learns how to optimize the performance metrics, seems better suited here. Among the online approaches, the Multi-Armed Bandit (MAB) framework is the one that best fits our problem since the agent can perform an action (choosing a configuration) without causing any change in the environment (the network is assumed to be static). This rationale led

multiple researchers (including ourself), to rely on the MAB framework to tackle the issue of configuring the TX_PWR and OBSS/PD parameters of APs in WLANs. Indeed, in 2017 [15] and then in 2019 [16], Wilhelmi et al. framed this problem as a MAB problem and proposed a distributed algorithm to be executed on each AP. The two works from 2017 and 2019 mostly differ by their definition of the reward, which represents the quality of a configuration. In [15], the reward at each AP is computed as its own throughput, which can be described as a "selfish" solution since each AP aims at optimizing its own reward regardless of the others. Conversely, in [16], the authors introduce a new definition for the reward based on a max-min function of the throughputs of the AP and its neighboring APs. Using a self-made simulator, the authors show that their solution significantly outperforms the default configuration of TX_PWR and OBSS/PD and that the selfish reward in [15] may lead to unfair situations between APs or STAs. In [17], Bardou et al. proposed another solution based on MAB to solve the problem of configuring TX_PWR and OBSS/PD parameters. They introduced an original way of sampling new configuration from the search space (as opposed to an uniform sampling as is the case with the classical ϵ-GREEDY strategy) and they use the realistic ns-3 simulator to validate the efficiency of their solution.

It is worth noting that, with the exception of [13], all aforementioned works assume a constant Modulation and Coding Scheme (MCS) for the nodes. However, in a real-life, every STA and AP dynamically adapt their MCS (determining the data rate at which frames are exchanged) as a function of the RSS and/or the frame error rate (e.g., [18]). This dynamic in the selection of MCS is further enhanced when the values of the TX_PWR and OBSS/PD parameters of nodes may vary. Although the interplay between these two adaptive mechanisms brings further complexity, evaluating the performance of an adaptive algorithm of TX_PWR and OBSS/PD in the presence of an automatic MCS selection mechanism can only strengthen the accuracy of any study.

To summarize, machine learning-based solutions methods appears more capable than others at finding efficient parameter settings for the sake of spatial reuse, especially when the number of APs and STAs comprising the WLAN is large. The MAB framework seems particularly well suited for that purpose as it has the capacity to learn from the current environment without prior knowledge of the network, unlike pretrained ANN. As compared to

existing MAB approaches addressing the same problem, our solution makes no assumption regarding the variance of the reward function unlike [15, 16], and introduces a novel and more efficient way of sampling the search space as compared to [17]. The combined effect of these contributions allows our solution to quickly find better configurations for the WLAN as shown in Section 5.

We use Table 1 to review the characteristics of the different approaches discussed so far for the problem of tuning `TX_PWR` and `OBSS/PD` in WLANs. This table shows that, except for [17], no machine learning-based solutions have been tested on realistic scenarios with the help of a detailed simulator such as ns-3, let alone the MCS automatic selection. Also, only the solution presented in [14] considered bidirectional (upstream and downstream) traffic in its performance evaluation. In this paper we are the first, to the best of our knowledge, to consider a machine learning-based solution for which the performance evaluation features (i) a realistic simulator such as ns-3 or Opnet, (ii) an MCS automatic selection on nodes, and (iii) the coexistence of upstream and downstream traffic.

Table 1: Comparison of the the state-of-the-art approaches. The last column refers to the number of APs and the number of orthogonal radio channels used in the performance evaluation. For instance, 100/11 indicates that the simulations consider a total of 100 APs distributed over 11 orthogonal channels.

Proposed solutions	Tuning of `OBSS/PD`	Tuning of `TX_PWR`	Dynamic MCS	Traffic Up/Down	Centralized	Simulator	APs / channels
WCMC'04 [9]	✓			Up/Down		Opnet	100/1
VTC'04 [10]		✓		Up		Self-made	8/1
Infocom'20 [11]		✓		Up/Down	✓	Self-made	100/11
WCNC'15 [12]	✓			Up		Self-made	100/3
WCNC'21 [13]	✓	✓	✓	Down		ns-3	6/1
Globecom'20 [14]	✓			Up/Down	✓	ns-3	3/1
ADHOC'19 [15]	✓	✓		Down		Self-made	8/1
JNCA'19 [16]	✓	✓		Down		Self-made	8/1
MSWiM'21 [17]	✓	✓		Down	✓	ns-3	10/1
Our work	✓	✓	✓	Up/Down	✓	ns-3	180/18

3. Objective Function

In order to properly frame our model as an optimization problem, we need to define an objective function that evaluates the quality of a network configuration for any WLAN. Recall that, in our case, a configuration consists in N_A pairs of parameters (`TX_PWR`, `OBSS/PD`). We want the objective function to return a score between 0 and 1, with 0 being the lowest score a configuration can get, and 1 the best possible score. From the standpoint of a network administrator, a good configuration must (i) guarantee enough throughput for each AP and STA, (ii) ensure a fair share of throughput among the APs and STAs, and (iii) maximize the WLAN overall throughput. Each of these three criteria may be associated to specific performance metrics. Criterion (i) relates to the number of starving STAs in the WLAN. In our work, a STA is said to be in starvation when its attained throughput is too low. Criterion (ii) is fulfilled by maximizing a definition of fairness such as Jain's index [19]. Finally, criterion (iii) corresponds to the system overall throughput, obtained by summing the individual throughputs of STAs. However, these criteria are most often not correlated and, in our case, criterion (i) prevails. For example, assume that a new configuration leads most STAs to significantly improve their attained throughput and thus the WLAN overall throughput. If this configuration also leads a single STA to become in starvation, then, as good as this solution may be, it should be viewed as less efficient than the current one.

We denote by T_i the throughput attained by STA i and by T_i^A the attainable throughput of STA i. The latter refers to the throughput STA i would have in the absence of all other competing devices in the WLAN. Clearly, we have $T_i \leq T_i^A$. Note that both T_i and T_i^A are not constant since their values depend on the tested configuration (due to the competition with other STAs and APs for the former and to the RSS and MCS in use for the latter). Throughout this work, STA i is labeled as starving of throughput whenever T_i/T_i^A is less than a given threshold γ (say $\gamma = 10\%$). Table 2 presents the principal notation used for the definition of the objective function.

We consider the proportional fairness (PF), which is simply obtained by multiplying the normalized throughputs of STAs, i.e., $\prod_i T_i/T_i^A$, to obtain a natural tradeoff between criteria (ii) and (iii). To account for criterion (i), which is the most critical concern, we partition the set of STAs into two

Table 2: Principal notation for the proposed method.

Parameter	Description
C	Configuration space
N_A	Number of APs
N_S	Number of STAs
T_i	Attained throughput of STA i
T_i^A	Attainable throughput of STA i
T^-	STAs in starvation situation
T^+	STAs not in starvation situation
γ	Starvation threshold for a STA

subsets: T^- the set of starving STAs and T^+ the set of non-starving STAs, and we compute the PF for each subset before combining them through a weighted average. This leads us to define the objective function f as follows:

$$f\left(c\right) = \frac{|T^-| \prod_{j \in |T^-|} \frac{T_j^-}{\gamma T_j^A} + |T^+| \left(N_S + \prod_{j \in |T^+|} \frac{T_j^+}{T_j^A}\right)}{N_S(N_S+1)} \tag{1}$$

where c represents any given network configuration and $|X|$ denotes the cardinality of X.

By definition of T^- and T^+, we have: $|T^-| + |T^+| = N_S$. Moreover, the numerator in Equation 1 verifies that any configuration causing more STAs to be in starvation than another will be associated to a lower returned value (thanks to the extra N_S term in the right component). Finally, the returned value of f is always in the range of 0 and 1 thanks to the denominator of Equation 1where N_S is used to normalize the weights $|T^-|$ and $|T^+|$, while $N_S + 1$ normalizes the two other terms (since we have: $\prod_{j \in |T^-|} \frac{T_j^-}{T_j^A} \leq 1$ and $N_S + \prod_{j \in |T^+|} \frac{T_j^+}{T_j^A} \leq N_S + 1$). Finally, let us remark that all variables involved in Equation 1 are fully observable. Of course, there are other ways of defining starvations. For instance, this could depend on the STA requirements. If the administrator is able to determine the required throughput for every STA of its WLAN, then it is straightforward to adapt the reward function of Equation 1 to account for this new definition of starvation.

4. Multi-Armed Bandit Solution

We model the optimization task as a MAB problem. Each WLAN configuration is viewed as an arm that the network controller can pull to observe its reward (the value returned by the objective function). Because of the large cardinality of the set of arms (21^{2N_A}), the network controller cannot realistically test all the arms in a reasonable amount of time. Therefore, although the set of configurations is finite, we argue that the optimization task is more precisely described as an Infinitely Many-Armed Bandit (IMAB) problem.

When dealing with IMAB problems, a common approach to circumvent the issue of dimensionality is to restrain the optimization to a subset of (randomly drawn) arms referred to as the reservoir (e.g., [20, 21, 22]). If no assumptions can be made between the arms and their rewards, then a natural way of making up the reservoir is to uniformly sample the arms. In our case, close configurations, say in the sense of the L_1 distance, are likely to obtain similar rewards. We express this hypothesis through Equation 2, which states that when the distance between two configurations is 1 (which is the smallest distance between two different configurations as the parameters `TX_PWR` and `OBSS/PD` are discrete and which corresponds to a change of parameter on a single AP), the difference between their rewards does not exceed δ.

$$\forall c_i, c_j \in C, \exists \delta \in [0,1], ||c_i - c_j||_1 = 1 \implies |f(c_i) - f(c_j)| < \delta \qquad (2)$$

Although Equation 2 cannot be ensured for every pair of neighboring configurations in the configuration space, our empirical study suggests that, for a value of $\delta = \frac{1}{N_S+1}$ (recall that N_S denotes the number of STAs in the topology), Equation 2 is verified by a large majority of configuration pairs. More details on this empirical study can be found in Appendix A.

Taking Equation 2 for granted, we can break down the optimizing task into two subproblems that must be solved concurrently: (i) Sampling promising configurations from the large configuration space, and (ii) Quickly identifying the best configuration out of the current reservoir. In [17], Bardou et al. showed the relevance of having a sampler agent to address subproblem (i) and an optimizer agent for subproblem (ii) when searching for an effective configuration for a WLAN. In the current paper, we adapt the definition of the two agents to better perform in realistic conditions (namely, with a larger number of APs and STAs, the presence of bidirectional traffic, and the

presence of rate adaptation letting STAs and APs dynamically select their MCS according to the quality of the radio channel).

4.1. Optimizer

Similarly to [15, 16, 17], our optimizer is based on Thompson sampling [23], which was proven to achieve a very good regret bound [24, 25] and generally has better empirical performance than other MAB algorithms such as UCB [26]. We assume that, for any configuration c_i, the reward is distributed according to a Gaussian distribution of unknown parameters (μ_i, σ_i^2). We use a Normal-gamma conjugate prior with parameters $\left(\hat{\mu}_i, \hat{\lambda}_i, \hat{\alpha}_i, \hat{\beta}_i\right)$ to describe the uncertainty on the pair (μ_i, σ_i^2). Thompson sampling [23] is used to quickly identify the best configuration c^* out of the current reservoir that verifies $c^* = \text{argmax}_c \, \mathbb{E}\left[R|c\right]$, where $R : C \to [0, 1]$ is a random variable representing the reward.

To cope with the complex interplay between the mechanism of selecting the values of `TX_PWR` and `OBSS/PD` and that of the dynamic MCS selection, we proceed as follows. Whenever a new configuration c_i is tested by the optimizer, the agent performs a series of n tests before considering a new sample $X_i : C^n \to [0, 1]^n$. Then it updates its belief on c_i using Equation 3, which is a classical result of Bayesian inference [27]. Here, $\bar{x}_i$ represents the empirical mean of the sample and s_i its empirical variance.

$$\begin{pmatrix} \hat{\mu}_i^{k+1} \\ \hat{\lambda}_i^{k+1} \\ \hat{\alpha}_i^{k+1} \\ \hat{\beta}_i^{k+1} \end{pmatrix} = \begin{pmatrix} \frac{\hat{\lambda}_i^k \hat{\mu}_i^k + n\bar{x}_i}{\hat{\lambda}_i^k + n} \\ \hat{\lambda}_i^k + n \\ \hat{\alpha}_i^k + \frac{n}{2} \\ \hat{\beta}_i^k + \frac{1}{2}\left(n s_i + \frac{\hat{\lambda}_i^k n \left(\bar{x}_i - \hat{\mu}_i^k\right)^2}{\hat{\lambda}_i^k + n}\right) \end{pmatrix} \tag{3}$$

Algorithm 1 details the behavior of our optimizer agent, parametrized by the desired sample size $n \in [2, +\infty[$ and an exploration rate $\epsilon \in \left[0, \frac{1}{n}\right]$. With probability $n\epsilon$, the optimizer will ask for a new promising WLAN configuration to the sampler. Otherwise, it will try to find the best arm in the current reservoir using Thompson sampling. This leads the agent to sample a Gamma and a Gaussian distributions for each configuration in the reservoir whose cardinality is proportional to the number of optimization steps k. Therefore, at step k, its computational complexity is $\mathcal{O}\left(k\right)$. In order

to execute properly, the agent needs to store all the configurations (of size $2N_A$) in the reservoir as well as four parameters by configuration $(\hat{\mu}, \hat{\lambda}, \hat{\alpha}, \hat{\beta})$. Therefore, its memory cost is $\mathcal{O}(kN_A)$ at step k.

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() $< n\epsilon$ **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 n times on the environment and collect rewards in X_j
    $k \leftarrow k + n$
    Update prior parameters $\left(\mu_j^k, \lambda_j^k, \alpha_j^k, \beta_j^k\right)$ according to Equation 3
    $X_j \leftarrow \emptyset$
  end if
  Send tests and rewards to the sampler agent
end loop

4.2. Sampler

The goal of the sampler is to discover promising candidates in the configuration space to build up the optimizer's reservoir. To outperform a simple uniform sampling of the configuration space, we take advantage of the assumed similarity in the score obtained by similar configurations. For two

given configurations, and based on Equation 2, we can cap the distance between their rewards with an upper bound, which is proportional to the L_1 norm of the difference between the two configurations. Equation 4 details this upper bound.

$$\exists \delta \in [0,1], \forall c_i, c_j \in C, |f(c_i) - f(c_j)| \leq \delta ||c_i - c_j||_1 \tag{4}$$

Proof. Assume two arbitrary configurations $u, v \in C$ and suppose that $||u - v||_1 = p + 1$. Then, we can find a set of p configurations $\{z_1, \cdots, z_p\}$ so that $||u - z_1||_1 = ||z_1 - z_2||_1 = \cdots = ||z_{p-1} - z_p||_1 = ||z_p - v||_1 = 1$. By triangle inequality, $|f(u) - f(v)| \leq |f(u) - f(z_1)| + |f(z_p) - f(v)| + \sum_{i=1}^{p-1} |f(z_i) - f(z_{i+1})|$. Applying Property 2, we have $|f(u) - f(z_1)| + |f(z_p) - f(v)| + \sum_{i=1}^{p-1} |f(z_i) - f(z_{i+1})| \leq \delta(p+1)$. Therefore, we have $|f(u) - f(v)| \leq \delta ||u - v||_1$, which is Property 4. Note that this property would translate into a Lipschitz property if our configuration space C was continuous.

We can use Equation 4 to obtain a lower bound for the distance between two configurations based on the distance between their rewards:

$$\exists \delta \in [0,1], \forall c_i, c_j \in C, |f(c_i) - f(c_j)| \geq l\delta \implies ||c_i - c_j||_1 \geq l \tag{5}$$

Proof. Let $\delta \in [0,1]$ for which Property 4 holds. Combining the assumption on the left of Equation 5 and Property 4, we get $l\delta \leq |f(c_i) - f(c_j)| \leq \delta ||c_i - c_j||_1$ which in turn leads to $||c_i - c_j||_1 \geq l$.
We can now determine how far the sampler should search from the current configuration to get a significant gain in the reward function for the next configuration. Let c_i be the current configuration and r_i its associated reward. We compute the distance between c_i and the next configuration to be sampled as follows:

$$d_i = \frac{r^* - r_i}{\delta} \tag{6}$$

where r^* denotes the best reward observed so far and $\delta \in [0,1]$. Throughout this work, and based on our empirical study, we use $\delta = \frac{1}{N_S+1}$. Note that this choice can also be supported by the fact that our reward function (see Equation 1) exhibits threshold effects that can be of magnitude $\frac{1}{N_S+1}$.

Unlike [17] that leverages Equation 6 using a multivariate Gaussian distribution, we opt for the use of a multivariate hypersphere distribution to explore the configuration space. The use of multivariate Gaussian distributions is

hampered by their relative sensitivity to the number of dimensions. When the number of dimensions is small, most of their density is centered around the mean value (i.e., within the standard deviation-ellipsoid) and, in our case, this will translate to exploring configurations that are very akin to the configuration c_i. On the contrary, in high dimensions (which can quickly occur given the exponential growth of our configuration space), a large part of the probability density is shifted to areas that are far away from the mean value [28]. To overcome these inconveniences, we let our sampler draw new samples from the surface of a hypersphere of radius d_i and centered on c_i. Equation 7 shows how to sample a random variable Z_{c_i,d_i} uniformly distributed on the surface of a hypersphere centered on c_i and of radius d_i.

$$Z_{c_i,d_i} = c_i + \frac{v}{||v||_2} d_i \tag{7}$$

where v is a standard normal random vector ($\forall i \in [1, 2N_A], v_i \sim \mathcal{N}(0, 1)$).

In order to concurrently explore multiple promising areas within the configurations space, we consider a mixture of K hypersphere distributions, wherein each hypersphere is centered on one of the K best configurations explored so far by the sampler. At first, the mixture is initialized with two hyperspheres, each of them representing a starting point for our algorithm. The first hypersphere is initially centered on the default configuration of 802.11, namely (`TX_PWR`, `OBSS/PD`) = (20, -82) dBm for all APs. For the second starting point, its location is based on the conflict graph between APs. Note that two APs are said to be in conflict when they cannot transmit at the same time (see [29]). Then, we simply decrease the `TX_PWR` of APs in a round-robin fashion until the conflict graph of APs reaches an average degree of 0.5. By doing so, we ensure spatial diversity between the two starting points with the aim of speeding up the search when adequate parameter settings are mostly far from the default configuration. Figure 2 illustrates a possible execution of the sampler in a two-dimensional space with two starting points and three hyperspheres.

Algorithm 2 summarizes the behavior of the sampler agent, with P starting points as parameters. As discussed before, we recommend $P = 2$ starting points: the default configuration of 802.11 and one that minimizes the average degree of the WLAN's conflicts graph. To sample a new configuration, Algorithm 2 needs to chose one of the K hyperspheres, sample a configuration

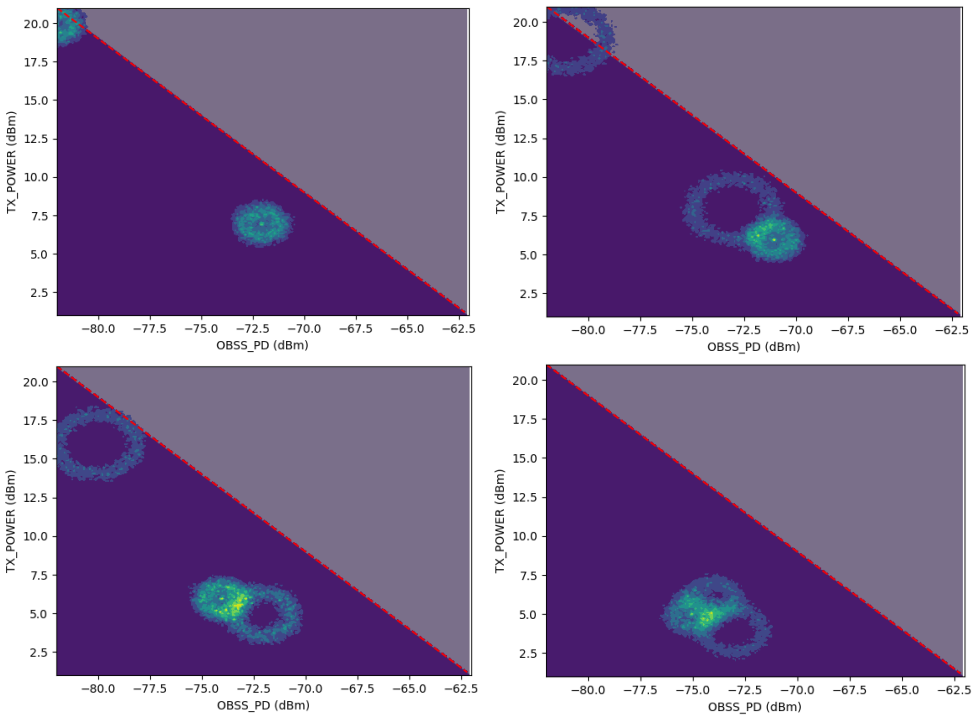


Figure 2: Four snapshots taken from a plausible execution of our sampling algorithm in a two-dimensional space with P=2 starting points and K=3 hyperspheres. The distribution density is shown in colors and the dashed line draws the frontier between authorized and unauthorized configurations according to 802.11ax (see Equation 8). From left to right, top to bottom: (i) The mixture is initialized with two starting points and the exploration begins; (ii) The mixture is updated with the three best distributions so far. The higher the reward, the more concentrated the distribution is; (iii) Another update of the mixture after a few iterations; (iv) Eventually, the three hyperspheres gather around the configuration (5,-75) dBm, which is relatively far from from the default configuration (20,-82) dBm.

of size $2N_A$, and update the K hyperspheres. Therefore, its computational complexity and its memory cost are $\mathcal{O}(KN_A)$.

Note that the 802.11ax standard [2] rules that the values of `TX_PWR` and `OBSS/PD` for each node must satisfy the following constraint:

$$\texttt{OBSS/PD} \leq \max(-82, \min(-62, -82 + (20 - \texttt{TX\_PWR}))). \tag{8}$$

Including this constraint in our sampler is straightforward as it simply reduces the size of the configuration space by a factor of 2^{N_A}.

Algorithm 2 Sampler algorithm

Input: number of hyperspheres K, target parameter δ, starting points P

if first call **then**
  Init mixture M with $\{(P_i, 1) \,|P_i \in P\}$
  Init weights W with $\{1\}$
  Init history H with $\emptyset$
  Init tests counter k with 0
else
  Retrieve previously built M, W, H and k
  Add pairs (conf, rew) transmitted by the optimizer
end if
Sample a new configuration c from mixture (M, W) that verifies Equation 8
Transmit c to the optimizer
$k \leftarrow k + 1$
if $k = \sum_{(c_i,d_i)\in G} d_i \dim c_i$ **then**
  Reset M 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 $\left\{\left(c_i, \frac{target - r_i}{\delta}\right)\right\}$ to M
    Add r_i to W
  end for
end if

Finally, Figure 3 illustrates a possible example of a dense WLAN in its default configuration with multiple conflicts among its APs; the interactions between the sampler agent, the optimizer agent and the experiments in the optimization process; and finally the WLAN after optimization in which many conflicts have been removed thanks to the configuration of `TX_PWR` and `OBSS/PD` at each AP.

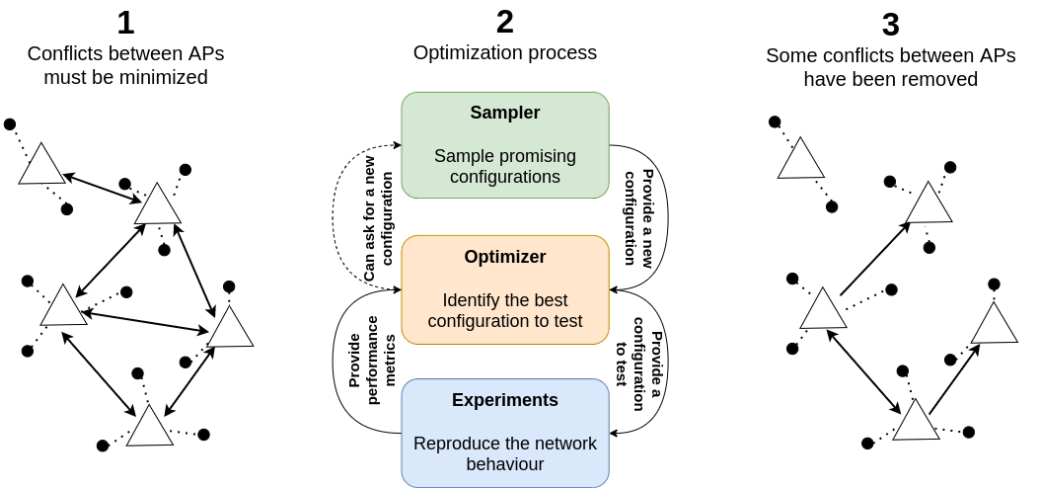


Figure 3: Schematic representation of our solution: (left) the initial set of conflicts between the APs of a WLAN in its default configuration (APs are represented by the white triangles, STAs by the black dots, and conflicts by the black arrows) ; (middle) the interactions between the sampler agent, the optimizer agent and the experiments, (right) the set of conflicts left in the WLAN after the application of our solution

5. Numerical Results

5.1. Experimental settings

To evaluate the performance of our solution, we consider three examples of WLAN that we denote by **T1**, **T2** and **T3**, respectively. Figure 4 illustrates the location of APs and STAs for each of them. **T1** is a simple example composed of six APs and a dozen of STAs. **T2** mimics the topology of the highly-dense WLAN deployed by Cisco in its offices in San Francisco [30]. To account for APs from lower and upper floors, we replicate the Cisco deployment on three floors and we run a simple channel allocation algorithm before selecting the most crowded channel to obtain **T2**. The resulting WLAN has

a total of 10 APs with an average of 5 STAs per AP, which are uniformly distributed within its vicinity (*i.e.* the intersection between its radio range at 20 dBm and its Voronoï cell in the WLAN). **T3** is very similar to **T2** as it only differs by the locations of STAs. On average, STAs from **T3** are much further away from their AP than in **T2** (the mean distance between an AP and its associated STAs is 12.93 meters in **T3** instead of 4.03 meters for **T2**). It follows that, on average, STAs in **T3** are about 3 times closer to their AP than to the closest competing AP. In the case of **T2**, this ratio is exceeding 9. Said differently, the relative distance between an AP and its associated STA is much larger in **T3** than in **T2**, and this significantly compounds the complexity of **T3**. In a sense, **T3** can be seen as a case in which the association between APs and STAs is far from optimal, or alternately, a case in which the number of available radio channels is too limited. In practice, this may turn to be the case thus making **T3** an interesting example but harder than **T2** to study.

We set the rate of data streams between every AP and each of their STA at 50 Mbps downstream and 3.33 Mbps upstream. This asymmetry reflects that STAs are typically much more downloading than uploading. Additionally, with this level of workload, **T1**, **T2**, and **T3** are all guaranteed to be in saturation: their APs are unable to properly serve all the needs of their STAs. Hence, we resort to the configuration of APs through the setting of their `TX_PWR` and `OBSS/PD` parameters to increase the QoS (Quality of Service) of the WLANs. Eventually, saturation can be seen as the worst-case scenario for each example. If our solution is effective in discovering an adequate tuning of the `TX_PWR` and `OBSS/PD` parameters under these circumstances, it can only do better in less saturated cases.

We compare the performance of our solution with those attained by five other strategies: (i) `DEFAULT`: the default configuration (namely (`TX_PWR`, `OBSS/PD`) = (20, -82) dBm for all APs), (ii) `WCNC'15`: a dynamic sensitivity threshold solution described in [12] applied to APs only, (iii) ϵ-`GREEDY`: the classical solution for MAB problems that, at each iteration and with probability $1-\epsilon$, uses the best configuration observed so far, and otherwise, uniformly draws a new configuration from the whole search space, (iv) `JNCA'19`: a centralized version of a MAB solution described in [16], and (v) `MSWiM'21`: another solution based on a MAB version described in [17] but with different optimizer and sampler agents. Note that all these strategies were amended to use

the reward function defined in Equation 1 and to apply their optimization only for the settings of the APs. Table 3 reports the numerical values of the parameters for each strategy. The starvation threshold γ is involved in the computation of the reward function and thus required by all strategies. The exploration rate, denoted by ϵ, is common to all the strategies but `DEFAULT` and `WCMC'04` and set to 0.1, a classical value for the exploration parameter. Finally, n, K and δ are specific parameters for [17] and our proposed solution, for which the values recommended in Table 3 provided very good empirical results. Let us recall that the choice for δ is discussed in Appendix A.

Table 3: Principal numerical values for the parameters of the strategies evaluated in the simulations.

Parameter	**Value**	**Description**
γ	0.1	Starvation threshold
ϵ	0.1	Exploration rate for strategies
n	3	Sample size in Algorithm 2
K	6	Number of distributions in Algorithm 2
δ	$\frac{1}{1+N_s}$	Hypothesis parameter in Algorithm 2

We implemented all the aforementioned strategies in the open-source network simulator ns-3 [8]. ns-3 is one of the few simulators that implements all the network protocols, mechanisms, and aspects from the Physical layer up to the Application layer and therefore is often regarded as among the most realistic simulators for WLANs. Table 4 reports the settings we used for ns-3. All our simulations last 120 seconds. To account for the potential high variance of the studied performance parameters, we replicate each simulation 22 times and we represent in the corresponding figures the first, second, and third quartiles. If a metric is subject to large variations, we extract and represent its trend using an exponential moving average in place of the raw data. Finally, to enable the reproducibility of all our results, we made available (in open-source) all the code we used for this section (including the three WLANs topologies and the implementations of the different strategies) [31].

In the ns-3 simulator, we instrument, collect and compute multiple performance parameters. On one hand, there are performance parameters that provide information on the goodness of the current state of the WLAN such

Table 4: ns-3 parameters.

Parameter	**Value**
ns-3 version	3.31
Number of repetitions	22
Simulation duration	120 s
Duration of an iteration	75 ms
Packet size	1,464 Bytes
Downlink traffic	50.0 Mbps
Uplink traffic	3.33 Mbps
Channel size	20 MHz
Frequency band	5 GHz
A-MDPU Aggregation	4
Path loss	`LogDistancePropagationLossModel`
Wi-Fi Manager	`IdealWifiManager`

as (i) the number of STAs in starvation, (ii) the Jain's fairness index [19] measuring how evenly STAs are being served by the APs, and (iii) the aggregate throughput that simply corresponds to the sum of all STAs' throughputs. On the other hand, we have performance parameters relating to the quality of the optimization such as (iv) the reward, computed using Equation 1, (v) the (instantaneous) regret which is obtained by subtracting the current reward to 1 (the upper bound of our reward function), and (vi) the cumulative regret which is computed as the sum of the regrets obtained at each optimization step. The latter is often viewed as the reference metric for machine learning problems as it reflects the overall good behavior of a solution over the whole simulation. This is why we will focus the most on it in our numerical results.

We now detail how the WLAN controller behaves in our simulation and applies the selected strategy to configure the APs. Every 75 ms, the performance parameters relating to the current state of the WLAN are collected by the WLAN controller that, in return, computes the value of the reward function associated with the current WLAN configuration. Afterward, the controller resorts to its strategy to determine the next configuration to be applied during the upcoming period of 75 ms. Note that with APs transmitting a single packet that reports the throughput of their associated STAs towards the controller on the wired network every 75 ms, the traffic overhead of our solution is small.

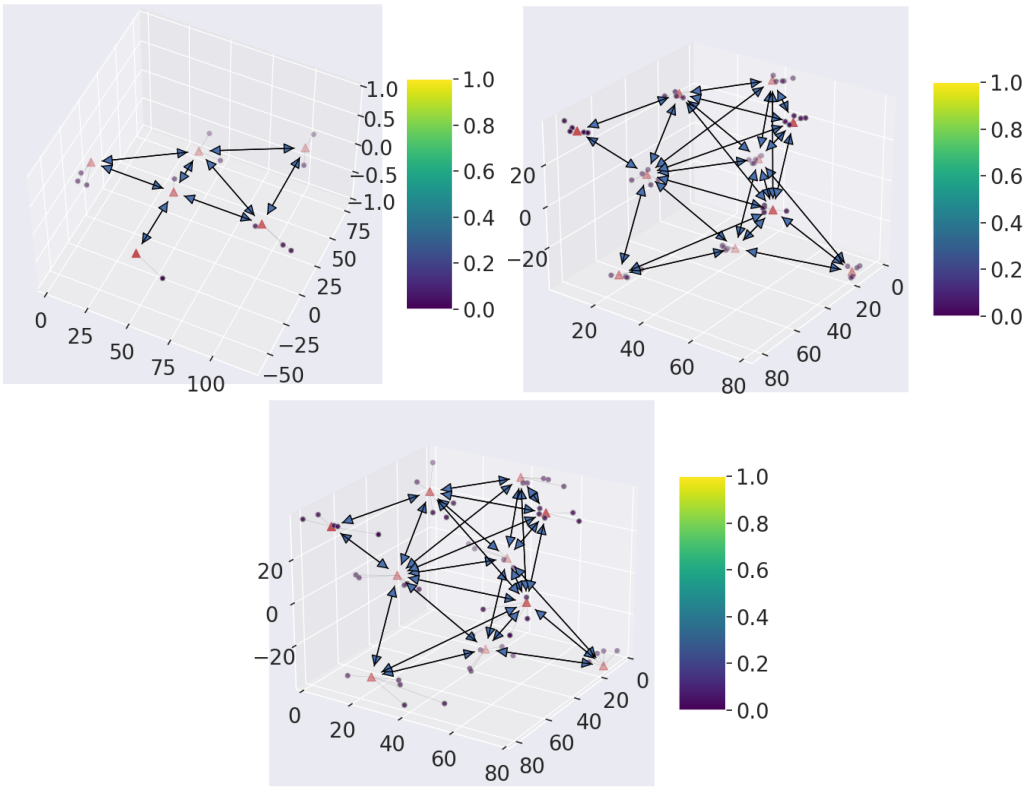


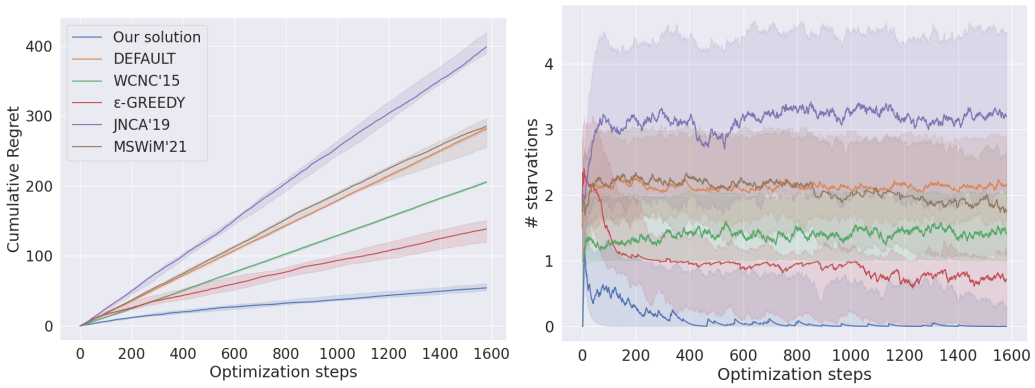
Figure 4: From left to right: topologies **T1**, **T2** and **T3**. APs are shown as red dots, conflicts between APs as double-headed arrows, and STAs as colored dots. Their color shows how frequently a given STA has a reasonable throughput (*i.e.* is not in starvation) with the 802.11ax default configuration (`OBSS/PD` = -82 dBm and `TX_POWER` = 20 dBm). A cool color means that the STA is often in starvation, while a warm color denotes a STA that is never in such a situation. All the scales are expressed in meters.

5.2. Simulation results

We start the analysis of the simulation results with **T1**. Figure 5a represents the cumulative regret for each strategy. First, we observe that our solution is able to reduce this metric by more than 80% compared to the `DEFAULT` strategy (which simply keeps the default 802.11 configuration) and outperforms all the other strategies. It is also worth noting that `JNCA'19`, and to a lesser extent our previous solution `MSWiM'21`, both struggle to do better than the `DEFAULT` strategy. These relatively poor performances can be explained by the presence in the simulator of the dynamic selection of MCS. Upon each new configuration of the WLAN, APs and STAs may change their MCS and

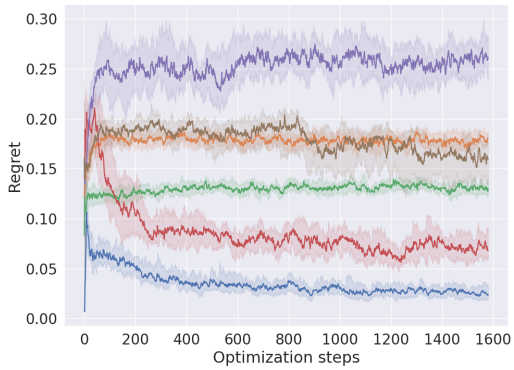
this selection process may take several dozens of milliseconds. During this period (i.e., before the selection of MCS has converged), the collected performance parameters can be unstable. Therefore, the most aggressive strategies (e.g., `JNCA'19` and `MSWiM'21` that change the WLAN configuration more often than the others) are more prone to suffer from these instabilities than ϵ-`GREEDY` and our new solution, which was designed to circumvent this bias by only allowing series of n consecutive tests for the same configuration before switching to another one. Figure 5c depicts the evolution of the (instantaneous) regret. We notice that, by the end of the 120 seconds of simulation, our solution is able to reduce its value by a factor of 80% as compared to its initial value. Looking at Figure 5b, we observe that only our solution is able to find a WLAN configuration in which no STAs experience levels of throughput corresponding to throughput starvation. At the same time, the WLAN configuration found by our solution leads to an increase of 40% for the fairness among STAs (see Figure 5d) while the aggregate throughput at the end of the 120-second simulation undergoes an increase of 66% (see Figure 5e). At the end of the simulation, the recommended configuration for the APs in **T1** is (in dBm): (-80,16), (-77,15), (-78,16), (-78,15), (-79,16), (-79,15).

We now consider a more realistic example corresponding to the dense topology **T2** illustrated in Figure 6. By the end of the simulation, the use of our strategy has led to a significant decrease of 87% of the cumulative regret as compared to `DEFAULT` configuration (see Figure 6a). This improvement mainly results from the ability of our solution to nearly remove all situations of starvation whereas the `DEFAULT` configuration keeps as much as half of its STAs in starvation (25 out of 50 STAs) as shown by Figure 6b. Looking at Figures 6d and 6e, we see that our solution also manages to find a fairer sharing of resources, improving the fairness by 125%, while still increasing the aggregate throughput by 133%. This demonstrates the ability of our solution to find an efficient trade-off between fairly shared resources and high system throughput in real-life dense WLAN deployments. The efficiency of our proposed strategy is particularly eloquent in this example where we can notice a significant gap between our solution and the second best strategy. At the end of the simulation, the recommended configuration for the APs in **T2** is (in dBm): (-76,12), (-78,12), (-76,9), (-72,9), (-75,10), (-79,14), (-77,8), (-75,11), (-76,9), (-75,13).

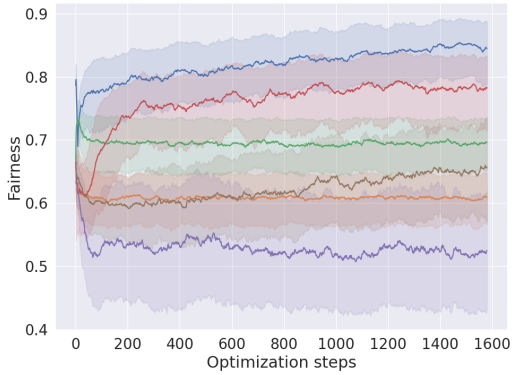


(a) Cumulative Regret

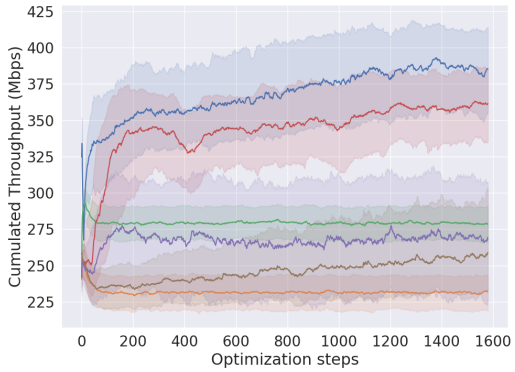
(b) Number of STAs in starvation



(c) Regret



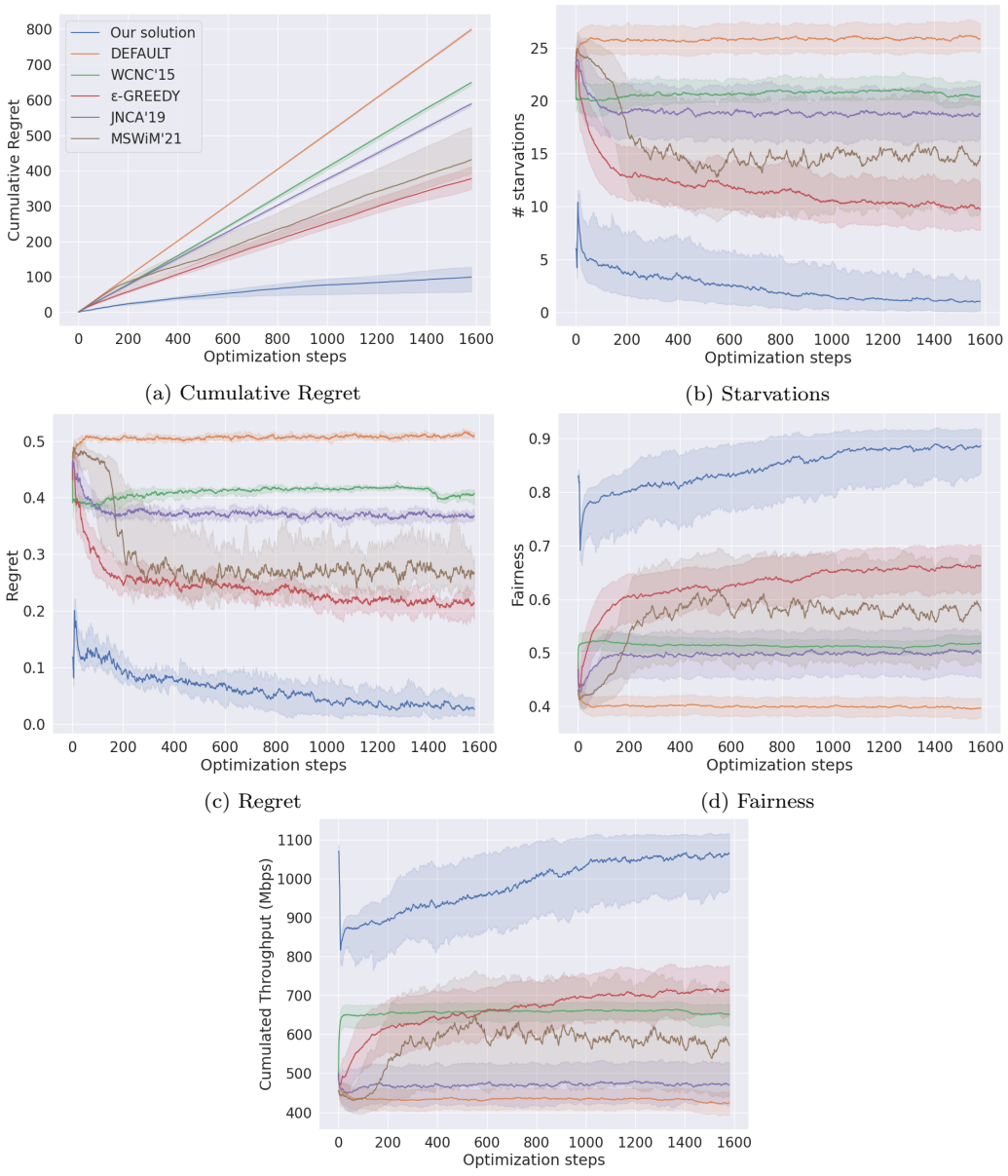
(d) Fairness



(e) Aggregate Throughput

Figure 5: Performance parameters on **T1** for each strategy.

We end this section with the example illustrated by the topology **T3**. Recall that this topology shares the same APs' locations as **T2** but represents a



(a) Cumulative Regret

(b) Starvations

(c) Regret

(d) Fairness

(e) Aggregate Throughput

Figure 6: Performance parameters on **T2** for each strategy.

willingly difficult scenario due to the STAs' locations that tend to be more in between their AP and the closest competing AP. Figure 7 shows the simula-

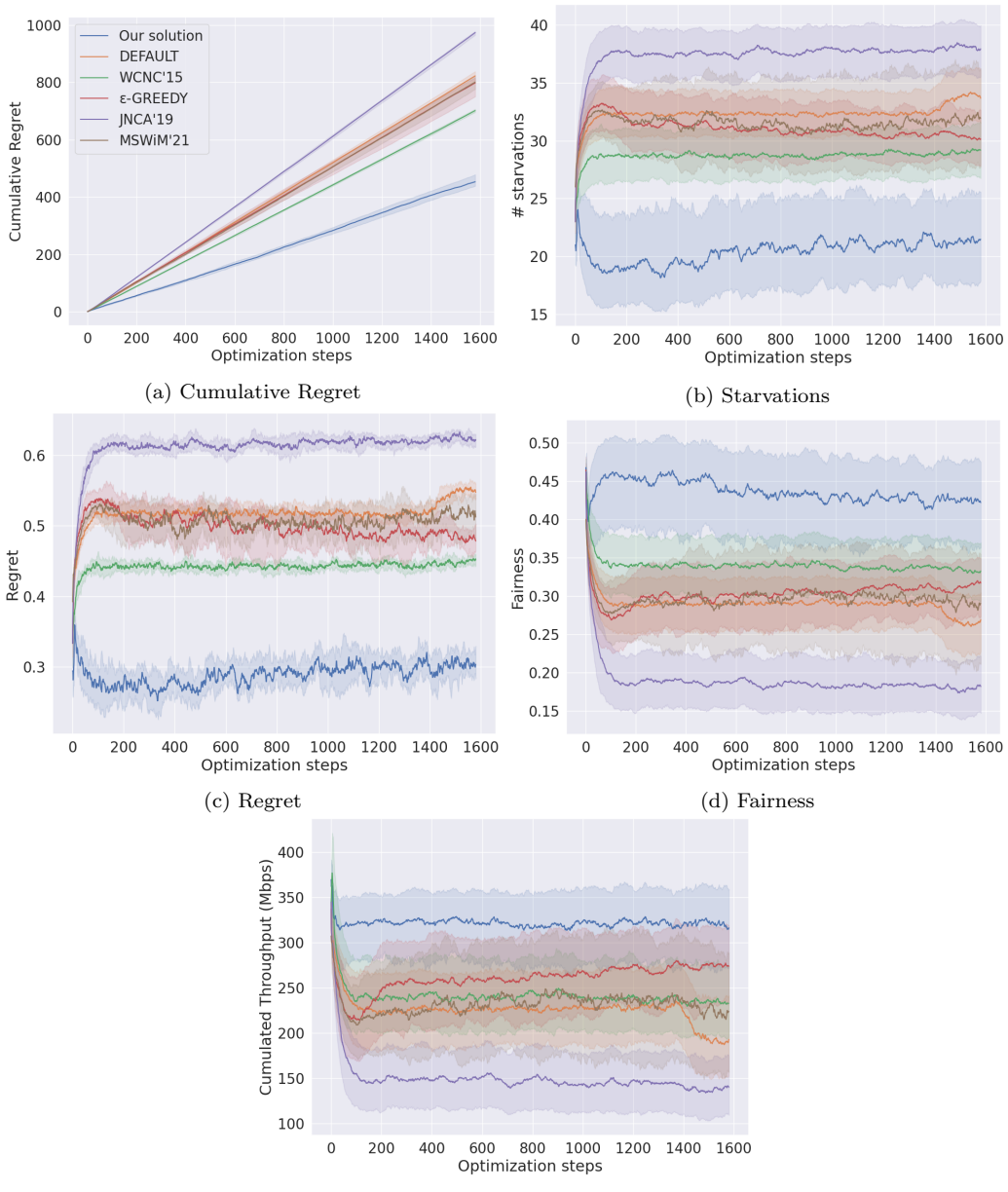
tion results obtained for this example. As expected, looking at the different performance parameters, we observe that, despite the help of our solution, the WLAN performance on **T3** remain relatively poor as compared to those attained by its counterpart **T2**. Indeed, by the end of the simulations, in the case of **T3**, the number of starving STAs is 22 (Figure 7b), the fairness is 0.45 (Figure 7d) and the aggregate throughput is 325 Mbps (Figure 7e) whereas for **T2**, our solution managed to prevent all starvations (Figure 6b), reach a fairness of 90% (Figure 6d), and raise the aggregate throughput above 1.2 Gbps (Figure 6e). Nonetheless, it is worth pointing out that, even in this example, our solution consistently reaches the best value for each considered performance parameter outperforming all the other tested strategies. Furthermore, our strategy brings significant gains as compared to the `DEFAULT` configuration, with an improvement of 48% on the cumulative regret, a 37% prevention of starvations, a fairness increased by 48%, and a 60% improved aggregate throughput. At the end of the simulation, the recommended configuration for the APs in **T3** is (in dBm): (-78,15), (-79,16), (-78,14), (-78,16), (-79,14), (-78,13), (-78,14), (-79,17), (-79,15), (-78,15).

5.3. Discussion

Overall, in all the examples we explored, including the three topologies presented in this paper, our solution never failed to quickly discover a configuration of `TX_PWR` and `OBSS/PD` that significantly reduces the number of STAs suffering from poor performance, improves the fairness among STAs and increases the cumulated throughput. Interestingly, the configuration found by our solution consists, in general, in decreasing `TX_PWR` and increasing `OBSS/PD` at each AP. To put it simply, at the end of the optimization, most APs tend to emit at a lower level but allow themselves to emit in more noised conditions (owing to the interferences of concurrent APs). This dual change favors spatial reuse of the radio channel with more simultaneous transmissions from nearby APs.

6. Conclusions

In this study, we have proposed a solution that improves the spatial reuse problem in radio channels of WLANs by taking advantage of elements introduced by the latest Wi-Fi protocol amendments. More precisely, our solution is able to configure two key parameters of APs so that the sharing of the radio channel among WLAN APs is significantly improved. The configuration



(a) Cumulative Regret

(b) Starvations

(c) Regret

(d) Fairness

(e) Aggregate Throughput

Figure 7: Performance parameters on **T3** for each strategy.

of these parameters is complex because of the high dimensionality of the state space and the interplay between multiple factors related to internal

Wi-Fi mechanisms and radio propagation aspects. To address this, we have framed our problem as a MAB problem to which we have proposed: (i) an objective (reward) function specially tailored to account for the principal performance parameters of interest in a WLAN, (ii) a Bayesian optimizer based on Thompson sampling, and (iii) an original sampler taking advantage of the peculiarities of our problem.

Using the ns-3 simulator that implements the full network stack with all its layers as well as advanced Wi-Fi mechanisms such as MCS automatic selection, we ran simulations on several realistic examples of WLANs (with regards to their topology and traffic) to evaluate the effectiveness of our solution. We then compared the performance of our solution with those delivered by other existing on examples inspired by real-life WLANs' deployments. Our solution significantly outperforms these approaches and, for a dense WLAN, its application can drastically reduce the number of STAs suffering from poor performance.

A natural follow-up would be to design a distributed version of our work. But this may prove to be extremely difficult because the definition of a relevant objective function in the sense of WLAN performance tends to push for a centralized computation. We plan to overcome this difficulty by using consensus algorithms. As part of our future works, we also plan to extend our solution to handle the case of mobile STAs.

7. 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 would like to express their sincere thanks to the anonymous referees for their remarks and comments.

References

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

[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 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.

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

[4] F. Wilhelmi, S. Barrachina-Muñoz, C. Cano, I. Selinis, B. Bellalta, Spatial reuse in ieee 802.11ax wlans, Computer Communications 170 (2021) 65–83. doi:https://doi.org/10.1016/j.comcom.2021.01.028.
URL https://www.sciencedirect.com/science/article/pii/S0140366421000499

[5] R. Laufer, L. Kleinrock, The capacity of wireless CSMA/CA networks, IEEE/ACM Transactions on Networking 24 (3) (2015) 1518–1532.

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

[7] G. Bianchi, Performance analysis of the IEEE 802.11 distributed coordination function, IEEE Journal on selected areas in communications 18 (3) (2000) 535–547.

[8] The Network Simulator ns-3, https://www.nsnam.org/, accessed: 2021-09-30.

[9] 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, IEEE Wireless Communications and Networking Conference (WCNC'04). (2004).

[10] 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.

[11] Q. Shuwei, C. Xiaowen, L. Yiu-Wing, J. Kee-Yin Ng, Joint access point placement and power-channel-resource-unit assignment for 802.11ax-based dense wifi with qos requirements, in: IEEE Conference on Computer Communications (INFOCOM'20)., 2020, pp. 2569–2578. doi:10.1109/INFOCOM41043.2020.9155490.

[12] M. S. Afaqui, E. Garcia-Villegas, E. Lopez-Aguilera, G. Smith, D. Camps, Evaluation of dynamic sensitivity control algorithm for ieee 802.11ax, in: IEEE Wireless Communications and Networking Conference (WCNC'15)., 2015, pp. 1060–1065. doi:10.1109/WCNC.2015.7127616.

[13] L. Hyunjoong, K. Hyung-Sin, B. Saewoong, Lsr: Link-aware spatial reuse in ieee 802.11ax wlans, in: IEEE Wireless Communications and Networking Conference (WCNC'21), 2021, pp. 1–6. doi:10.1109/WCNC49053.2021.9417353.

[14] E. Ak, B. Canberk, Fsc: Two-scale ai-driven fair sensitivity control for 802.11ax networks, in: IEEE Global Communications Conference (GLOBECOM'20)., 2020, pp. 1–6. doi:10.1109/GLOBECOM42002.2020.9322153.

[15] 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

[16] 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

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

[18] R. Grünblatt, I. Guérin-Lassous, O. Simonin, Simulation and performance evaluation of the intel rate adaptation algorithm, in: ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM'19), 2019, pp. 27–34.

[19] R. Jain, D. M. Chiu, H. WR, A quantitative measure of fairness and discrimination for resource allocation in shared computer systems, CoRR cs.NI/9809099 (01 1998).

[20] 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.

[21] 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.

[22] 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.

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

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

[25] 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`

[26] P. Auer, Using confidence bounds for exploitation-exploration trade-offs 3 (null) (2003) 397–422.

[27] J. Bernardo, A. Smith, Bayesian Theory, Wiley, 2000.

[28] B. Wang, W. Shi, Z. Miao, Confidence analysis of standard deviational ellipse and its extension into higher dimensional euclidean space, PLoS One 10 (03 2015). doi:10.1371/journal.pone.0118537.

[29] L. Abdelwedoud, A. Busson, I. Guérin-Lassous, M. Foare, M. L. Diakite, M. Farouk Nanne, Inference of a clear channel assessment based conflict graph, Internet Technology Letters (2020) 1–6doi:10.1002/itl2.227.
URL https://hal.inria.fr/hal-03124090

[30] High density wi-fi deployments, https://documentation.meraki.com/Architectures_and_Best_Practices/Cisco_Meraki_Best_Practice_Design/Best_Practice_Design_-_MR_Wireless/High_Density_Wi-Fi_Deployments, accessed: 2021-09-30.

[31] A. Bardou, T. Begin, A. Busson, Online repository for code, https://github.com/abardou/IMAB-RING-SR-AP-802.11AX (2021).

Appendix A. Setting the hyperparameter δ

In this work, we choose a value of $1/(N_S+1)$ for the hyperparameter δ. This choice is supported by the following reasoning. In general, incrementing or decrementing TX_PWR or OBSS/PD at a single AP in a dense network has a limited effect on the SINR, which causes little change to the reward value. However, due to the specific definition of our reward function, it may occur that an increment or decrement of TX_PWR or OBSS/PD at a single AP causes the throughput of a given STA to move above or below the starvation threshold. If that is the case, then the reward is changed by approximately $1/(N_S+1)$. This explains why we selected $\delta = 1/(N_S+1)$.
To further support our choice of having $\delta = 1/(N_S+1)$, we conducted an empirical study. Figure A.8 shows the empirical distribution of the distance in terms of their reward between two neighboring configurations in the configuration space. For the topology **T1** in which $N_S = 12$ STAs, we randomly sample 270 pairs of neighboring configurations for which we compute the distance between their reward. Then, we use Figure A.8 to show how many of these random pairs (c, c') verify the inequality $|r(c) - r(c')| \leq \frac{1}{N_S+1}$. It appears that a large majority (more than 93 %) of neighboring configuration pairs are meeting this criterion. Therefore, by selecting $\delta = \frac{1}{N_S+1}$, we can reasonably ensure that Equation 2 will be verified in most cases.

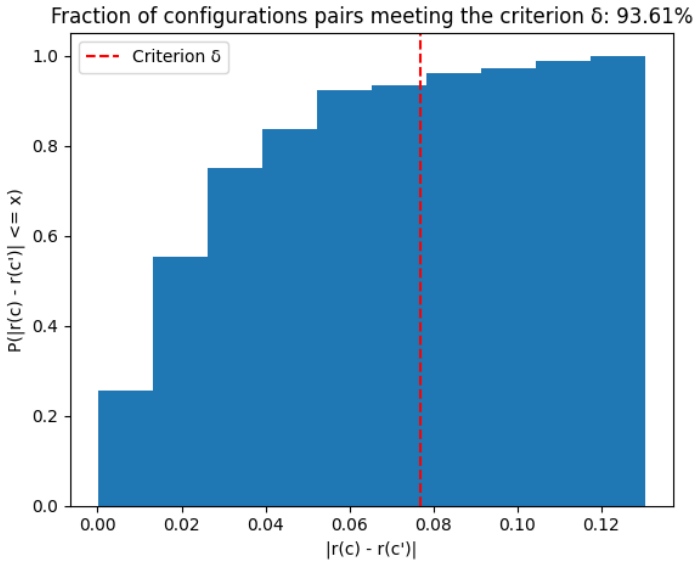


Figure A.8: Empirical cumulative distribution function of the absolute difference $|r(c) - r(c')|$ between the rewards of two neighboring configurations c and $c' = c + h$, with h being a one-hot vector. 270 pairs of configurations uniformly distributed in the configuration space of topology **T1** were collected to generate this empirical distribution. Recall that **T1** has $N_S = 12$ STAs so that $\delta = \frac{1}{N_S+1} = \frac{1}{13} \approx 0.077$ in this experiment.