A Dynamic and Autonomous Channel Selection Strategy for Interference Avoidance in 802.11

Francesco Maturi* Francesco Gringoli†, Renato Lo Cigno*
*Dept. of Information Engineering and Computer Science, University of Trento, Italy
†Dept. of Information Engineering University of Brescia, Italy
francesco.maturi.paparot@gmail.com
francesco.gringoli@unibs.it – locigno@disi.unitn.it

***Abstract*—The coexistence of different technologies within the same spectrum swaths, as well as the distributed, non-coordinated channel assignment for WLANs is becoming not only an important research topic, but also a matter of sustainability for the trend we witness in moving all last-hop communications on wireless links to unleash the users from their wires and tethers. This work proposes, implements and evaluates a strategy that allows an entire 802.11 Basic Service Set (BSS) to dynamically hop between the available channels always selecting the "best" one. The selection not only guarantees the hopping BSS performances well above what can be achieved with a static selection, but it also minimizes the interference toward other BSSs, so that the overall performance of the system maximizes resource utilization. The performance of the protocol and strategy we propose is tested with an implementation on off-the-shelf 802.11g cards; the experiments are run in labs and around the campus of the University of Trento.**

I. Introduction

Spectrum scarcity is rapidly becoming one of the major limitations in wireless communications; yet some portions of the spectrum are used sparingly. This last observation is valid both in time and space, and is one of the major drivers of the research on white spaces and cognitive radio. 802.11 (WiFi) networks are no exception: Indeed, the Industrial Scientific Medical (ISM) bands where WiFi networks operate are probably the most crowded and interfered bands of the entire spectrum, but this does not mean that they are uniformly overused, as the channel assignment to Basic Service Sets (BSSs) is rarely the outcome of a proper planning (with some rare exceptions). Rather, channel assignment to BSSs ranges from random assignment to "hypercrowding" of channel 1, the default assignment of most low cost Access Points (APs), and under-usage of other channels in the 2.4 GHz band; the 5 GHz band is far less used, even when devices can freely select to use it.

The optimal channel assignment is NP-complete [2], but efficient centralized approximations exist, and we can find them in commercial management systems. Also distributed algorithms have been explored in cooperative environments [3], and they find good solutions. However, very often APs belong to different administrations (just think about the typical high-rise condo) and BSS simply do not coordinate. Autonomously and dynamically finding a good channel is the goal of this work, solving both foundational issues as how to locally define and measure interference levels, and more technological ones, as coordinately hopping from one channel to another the AP and all the stations that form a BSS. Strangely enough, as we discuss in Section II, this problem received very little attention. Only a very recent paper [4] that appeared as we were running our measurement campaign addresses and brilliantly solves this problem: Section II discusses the relationship and the differences with our work in detail.

Our main contributions can be summarized as follows:

- We propose a simple scheme to let the AP of a BSS select on a short time-frame the least occupied WiFi channel and move the entire BSS on it without BSS disruption;
- We identify and select a set of very sensitive metrics at the Medium Access Control (MAC) and Physical Layer (PHY) protocols that empower an accurate and fast evaluation of how disruptive interference is;
- We define a simple protocol to ensure that all stations correctly follow the BSS hopping and can recover the synchronization in case they loose it;
- We implement the proposal in standard devices with Broadcom chipsets and run an extensive measurement campaign to assess the gain achievable by the system.

II. Background and Related Work

Channel assignment has been extensively studied in the literature in many different contexts, ranging from cellular access in licensed bands [5], cognitive networks [6] and Wi-Fi [7]. The impossibility of network planning and global channel assignment ISM bands, makes channel selection a tough problem, which has not yet found a definitive solution as shown by large scale analyses repeated in the last ten years [4], [8]. Despite the huge research activity in the field, the technology transfer to commercial products is still negligible: A simple method that can be adopted by consumer stand-alone APs is to sense the environment and tune on the channel with the least amount of congestion [9], but very often they simply tune on a default channel. In any case, in consumers' APs once a channel is selected, it is never changed until reboot. Channel occupation information reported by APs in large corporate networks can feed a controller that configures all channels in a centralized fashion: As pointed

This research is partially supported by the European Commission, H2020-ICT-2015 Programme, Grant Number 688768 'netCommons' (Network Infrastructure as Commons).

ISBN 978-3-901882-88-3

out by [2], the underlying graph coloring problem is NP and some heuristic is needed to find approximate solutions. Other proposals seek to extend this approach: [10] in addition to frequency can configure the bandwidth of a multichannel 802.11n/11ac network; [11] adds interference estimation to the metrics considered by the algorithm; and [12] brings the method to mesh networks. Anyway, the majority of recent works agree on the complexity of the centralized approach and suggest the adoption of distributed algorithms. [13] and [14] propose that each AP of a multiradio mesh network collects and share with the others a set of MAC level statistics, like delivery ratio, contention time or interface utilization: each AP configures autonomously its channels to improving the network throughput. Similarly, [15] estimates and shares the interference relationship among clients in neighboring BSSs to configure channel aggregation in addition to frequency in 802.11n networks. As these techniques build on Inter Access-Point Protocol (IAPP) communications, their deployment is restricted to big enterprise networks. Similarly to the work we present here, [16] does not require IAPP: This not only reduces complexity, it also makes possible the deployment of a distributed algorithm on co-located APs belonging to heterogeneous organizations/persons. The proposal, however, assigns disjoint channels with priority to neighboring APs that serve more clients and it does not take into account any metric relative to the "quality" of the transmission. Low level metrics, instead, are considered by [17] and [18], which introduce heuristics to minimize the global interference in a distributed manner. While the former considers at each AP the noise and the energy received from surrounding devices in other BSSs to make a decision, the latter can be based on any metric, such as the number of packet re-transmissions or Signal to Noise Ratio (SNR) measurement, to update the success probability of the current channel, and choose the best one.

Instead of selecting a channel when needed, [19] and [20] propose frequency hopping as a viable solution to the frequency allocation problem. With respect to previous approaches, these works do not perform any measurement on the channels they select: Simply, they use constant hopping to enforce on average equivalent network conditions to every participant. [19] distributes hopping schedules to the nodes that compose a single ad-hoc network so that different pairs of peers communicate on orthogonal channels without interfering. Similarly [20] determines a different schedule for each BSS so that on average each sees the same number of interfering neighboring BSSs. This is equivalent to random hopping, which is the reason why in experiment we use random hopping for comparison. Recently [4] presented a scheme driven by the AP in each BSS that keeps estimating the network conditions by collecting basic layer-two statistics: Then it randomly hops in different channels, staying in good channels for long time while escaping from bad ones. Our method, on the contrary, besides using different metrics, hops constantly and according to a tight pacing: To this end we follow the same low level scheme recently adopted in [21] where hopping is used to counter jamming rather than for channel selection. The goal of [4] is asymptotic optimality, which is important and proven in the paper, the goal of our work instead is following bandwidth availability in space and time, finding locally good solutions in a reactive way.

III. Channel Hopping with a BSS

Devising and developing a channel hopping protocol for a BSS requires addressing two different problems: *i)* identify a suitable set of metrics that enable to evaluate the status of the channel fast and reliably; and *ii)* guarantee that regardless of all possible scenario and situations the BSS is preserved, i.e., all stations change channel together and synchronously. Furthermore, one may want to find the channel dwell time distribution that maximize performance and a hopping strategy that ensure convergence when many BSS play the same game. Our contribution answer the first two fundamental points, and leave further optimization for future work, but experiments with two Hopping-BSS (H-BSS) not reported here for lack of space show that they coexist very well. We note once more that [4] gives a very interesting answer to the convergence problem, proposing a hopping scheme that asymptotically converges to an optimal channel selection.

The goal of H-BSS is the exploitation of "WiFi white spaces," thus we seek a solution that is fast in changing channels and does so more or less continuously, in the assumption that, in ISM bands, there is nothing like an unused channel. Moreover, we look for a very robust solution that guarantees that the system will keep working in face of non 802.11 interference (for instance possible future cellular Long Term Evolution (LTE) usage of these bands), or heavy overload of the channel. Let's clarify this key concept. The idea of white spaces is to dynamically exploit spectrum unused by some primary user or legitimate licensee. In ISM bands, however, there is nothing like a licensee or a primary user, but still there is temporarily unused spectrum, i.e., channels that are not used by any BSS for some period of time: Our goal is not only finding a low utilization (ideally free) channel, but aggressively exploiting channels that are left unused by other BSSs.

To achieve this goal, we need metrics that are very sensitive to the channel use and most of all to its occupation by signals or noise that is not intelligible to 802.11; moreover we need a protocol resilient to extreme situations to guarantee that the BSS hops to the next scheduled channel without leaving any station behind. In the following, we discuss the metrics used and describe the protocol we devised for hopping, which is and extension of 802.11h [1]. We assume that there is a BSS coordinator that perform the measures and take the decision for the BSS, and we avoid multi-point measures to keep the overhead low and the implementation in stations as simple as possible. If the BSS is structured, this is naturally the AP, while in case of an Independent BSS, it can be the station that is in charge of sending the beacons. For the sake of simplicity we call it AP in any case. The BSS continuously hops among channels and the AP takes measures on each and every visited channel, then it instructs the BSS on the next hopping cycle sequence.

A. Channel Metric Selection

To estimate the quality of a channel from a single observation point, like the AP, we need to go beyond simple metrics like delivery probability, access time, and channel idle/busy time. For instance, interference that makes frames from one or more stations impossible to receive at the AP, or interference from systems that do not correctly trigger the synchronization correlators, or trigger them without an 802.11 frame following them, are indications of a channel unsuitable for the BSS, but are not captured by delivery probability or access delay at the AP. Network cards normally do not report non-protocol events to the kernel, or they report it as long term statistics, thus disruption of the Physical Layer Convergence Procedure (PLCP) header, which has devastating impact on performance, is not captured by traditional metrics. Capturing this kind of external interference is crucial for a hopping BSS, and avoiding channels affected by it gives significant throughput gains [22].

For these reasons we build our cross-layer metrics including statistics concerning the quality perceived by the AP when *transmitting* and when *receiving* packets from other nodes, including external BSS and non intelligible interference, and measuring errors at various levels. The implementation of these metrics requires working directly in the NIC CPU: this enables to capture and react faster to a wider set of events and to recognize interference even if due to non 802.11 transmissions [23], a feature unique to this work.

Name	Description
S_T	Number of successfully transmitted data packets that were acknowledged by the receiver.
F_T	Number of transmitted data packets for which no acknowledgment was received.
S_R	Number of data packets correctly decoded including also those addressed to other BSSs.
S_R^{bss}	Number of data packets correctly decoded addressed to the BSS.
F_R^{chs}	Number of received data packets that failed the CRC/32 test.
F_R^{plcp}	Number of corrupt PLCP headers.

Table I
CHANNEL QUALITY ESTIMATION VARIABLES

Table I reports the variables that we use in building a proper channel selection metrics. Unfortunately, as the definition of best channel is complex, we need two different metrics that are combined to achieve the desired goal.

The first metric measure the overall packet error rate including packets transmitted and received by the AP. The value of the metric is the ratio between the number of *successful* and *failed* packets including transmissions originated from other BSS and spurious signals, i.e., those that occupy the medium for some time but that did not turn into 802.11 frames, and frames for which the noise did not allow to correctly decode the PLCP. Its formal definition is:

$$M_1 = \frac{S}{F} = \frac{S_T + S_R^{bss}}{F_T + F_R^{chs} + F_R^{plcp}} \quad (1)$$

where S_T and S_R^{bss} are the number of frames successfully transmitted and received by the AP (excluding packets correctly received but addressed to other BSS) during the measure interval, and F_T, F_R^{chs}, and F_R^{plcp} are the number of packets transmitted by the AP without ACK, the number of frames failing the CRC check, and the number of triggered receptions that did not pass the PLCP synchronization phase.

Metric M_1 in Equation (1) provides a *channel ranking*, but to select a good channel we need a second metric to measure the activity of neighboring BSSs. M_2 in Equation (2) describes this metric subtracting the number of correctly decoded packet addressed to the BSS from the total providing an estimate of the *external* traffic:

$$M_2 = S_R - S_R^{bss} \quad (2)$$

We use Algorithm 1 to select the best channel at the end of a hopping cycle. To determine the best channel we first rank the measured channels (i.e., all those visited during a hopping cycle and stored in set S_{prb}) by maximizing M_1 (Line 2 in Algorithm 1) and next we choose among the top three the one that minimizes M_2 (Line 3). Clearly the use of this second metric makes sense only if it is smaller than a given threshold T_{opt}, otherwise the best selection is simply the channel with the maximum M_1 (Line 5). We describe the hopping cycle and its supporting protocol in detail in Section III-B.

Algorithm 1: Algorithm used at the end of each hopping cycle to select the best channel.

```
1 Event OnEndCycle(S_prb, M_1, M_2)
2     set ch_sort[.] = RankChannel(S_prb, M_1)
3     set ch_bst = FindMin({ch_sort[1..3]}, M_2)
4     if M_2[ch_bst] > T_opt then
5         set ch_bst = ch_sort[1]
6     return ch_bst
```

B. Hopping Protocol and Signaling

We divide the BSS time in *hopping cycles*. Each cycle is composed of slots of fixed duration $\Delta T_{slot} = 4\,\text{s}$ whose channel is determined by the *hopping sequence*. As we sketch in Figure 1, the sequence $L_{hop}(n+1)$ for cycle $n+1$ is determined with the statistics collected during cycle $n-1$. $L_{hop}(n+1)$ includes always 26 slots on the best channel $ch_{bst}(n+1)$ (we identify it with number 0) and a maximum of 13 *probing slots*: we do not probe, in fact, those channels that in cycle $n-1$ were adjacent to a *crowded channel*, i.e., one with M_2 exceeding a configurable threshold T_{crw}. The rationale here is that in such channels the subtle cross-channel interference from a *crowded* channel cannot be easily detected leading to unavoidable collisions. Following considerations on the channel spectrum width $B = 20\,\text{MHz}$ and motivated by experimental observations we mark as adjacent to channel ch all channels in the set $[ch-2, ch+2]$ excluding ch. Independently of adjacency, all crowded channels are always added to hopping sequence

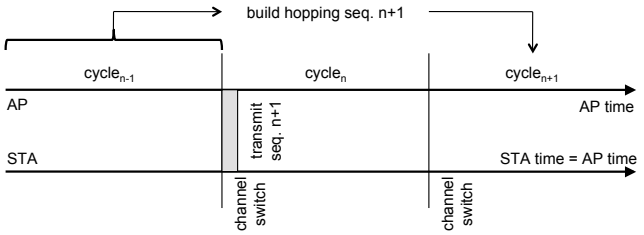

Figure 1. Hopping cycles: at the end of cycle $n-1$ the AP builds the hopping sequence for cycle $n+1$ and it immediately starts transmitting hopping sequence $L_{hop}(n+1)$ to all stations.

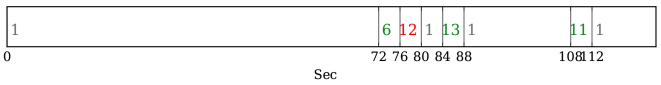

Figure 2. A cycle is divided in slots according to hopping sequence L_{hop}: slot numbers represent the channel. Black slots are on the best channel ch_{bst}. Green are the probing slots and red is the one evaluated as new best.

L_{hop}. To avoid synchronization effects between different H-BSS, L_{hop} is randomized. With the current choice of ΔT_{slot} we empirically fix T_{crw} to 500 packets: we plan to investigate the effect of shorter slots as a future work, together with the choice of the corresponding parameters.

Algorithm 2: Algorithm controlling channel hopping during BSS life.

```
1  set L_hop(1) = L_hop(2) = RND[1..13]
2  for cycle ← 1 to ∞ do
3      set S_prb = ∅
4      set S_crw = ∅
5      set L_hop^new = {0_26, j}, 1 ≤ j ≤ 13
6      for ch ← L_hop(cycle) do
7          if ch = 0 then
8              SetChannel(ch_bst(cycle))
9              Wait(ΔT_slot)
10         else
11             SetChannel(ch)
12             ClearStats()
13             Wait(ΔT_slot)
14             set S_prb = S_prb ∪ {ch}
15             Compute M_1 [ch], M_2 [ch]
16             if M_2 [ch] > T_crw then
17                 Remove [ch − 2..ch + 2] from L_hop^new
18                 set S_crw = S_crw ∪ {ch}
19     set L_hop^new = L_hop^new ∪ S_crw
20     set ch_bst(cycle + 2) = OnEndCycle(S_prb, M_1, M_2)
21     set L_hop(cycle + 2) = RND[L_hop^new]
22     Transmit([ch_bst, L_hop](cycle + 2))
```

At the beginning, when the BSS is created, the AP executes two warm-up cycles: the two hopping sequences $L_{hop}(1,2)$ are composed of 13 randomized probing slots covering all the channels (Line 1 in Algorithm 2). At the beginning of the second warm-up cycle, the AP creates the hopping sequence $L_{hop}(3)$ for the third cycle: from this point on, the mechanism repeats periodically. At each cycle we first initialize empty lists of probed and crowded channels S_{prb} and S_{crw} and a new complete hopping sequence L_{hop}^{new} (Line 3-Line 5). We then iterate on the current hopping sequence (Line 6): we set the channel either to the current best (Line 7) or to a different probing one (Line 10) where we spend ΔT_{slot} s. In the latter case we collect statistics, compute metrics M_1 and M_2 and add the channel to the list of those probed S_{prb}; if needed we add it also to the list of crowded S_{crw} (Line 11-Line 18). Finally, at the end of the cycle we compute the new best channel invoking Algorithm 1 and we finalize the new hopping sequence: we then transmit data to all stations and skip to the next iteration (Line 19-Line 22).

To enable transparent channel switching of the entire BSS, the AP distributes sequence L_{hop} and best channel ch_{bst} for a cycle at the beginning of the previous one: stations can hence switch among channels *independently of and together with* the AP for an entire cycle. Differently than [4], we did not use the 802.11h Channel Switch Announcement (CSA) feature recently included in the standard [1]: even if it scales well, in fact, it is suited only for sporadic channel switches. In a constant hopping scenario it might cause disassociation of stations after switching to a bad channel because of consecutive beacons losses. We developed instead a signalling protocol based on TCP: given the relatively long duration of cycles with respect to network packet transmission time, it reduces to zero the probability that a station does not receive the hopping sequence and, in fact, we never observed stations disconnecting even when switching to a bad probing channel[1]. With our low level approach then each NIC switches channel at the absolute time contained in the hopping sequence L_{hop}: as all NICs in the BSS share the same AP clock advertised by beacons, channel switch is perfectly synchronized.

IV. Implementation

The functions of our framework are divided between the Linux operating system, that manages those not time critical like building the new hopping sequence, and the NIC firmware, that handles time–critical operations including the actual execution of the channel hopping and collection of low–level statistics. Although this approach is feasible in the majority of the wireless NICs – they all execute a low–level firmware in their internal core – we chose the Broadcom KBFG4318 NIC as it is compatible with OpenFWWF, the only open–source firmware ever release [2]. These two elements have been recently adopted as a research platform: Even if they support only legacy 11b/g modulations, they are compatible with all the timings included in the DCF protocol, and many new features have been introduced both at the MAC and PHY layers [25]–[27].

[1] For the experiments described hereafter the signalling overhead was negligible thanks to the small size of the BSS. We can improve the scalability by moving to a solution based on reliable multicast such as 802.11aa [24]. [2] The tool can be downloaded from the official website http://netweb.ing.unibs.it/openfwwf/

For this reason we consider this to be a proof–of–concept which can be easily migrated to newer chipset once a open–firmware is available and adopted by any vendor that wish to.

With respect to the collection of the measurements, we modified some handlers in the firmware, more specifically:

- `rx_complete`: executed at the end of each reception, to measure the number of frames correctly S_R or erroneously F_R^{chs} received transmitted from inside/outside the BSS;
- `rx_badplcp`: executed when a spurious preamble is detected, to count F_R^{plcp} the number of corresponding events;
- `tx_contention_params_update`: executed at reception of the acknowledgment of a transmitted data frame, or at its expiration, to measure the number of frames correctly S_T or erroneously F_T transmitted.

The collected statistics are divided per hopping period, and stored into the *shared memory* that is later accessed from the user–space.

With respect to the channel hopping procedure, the firmware receives the hopping sequence from the user–space as a table with switching times and target channels stored in the shared memory. The main loop of the firmware *state_machine_start* keeps checking the NIC clock, that is kept synchronized to that of the BSS AP with the reception of beacons. By comparing the clock with the next switching time, the MAC CPU can switch the channel with μs accuracy. Nodes switch channel independently on the channel activity: We did not attempt to avoid or delay switch during a packet transmission/reception as nodes may detect the channel status differently according to noise, collision or their location.

Finally we developed an user–space application running at the AP side that collects statistics generated by the firmware from the shared memory, computes the metrics and decide the next channel hopping sequence: It stores the new sequence in the shared memory and sends it to all connected stations where a receiving application acts similarly.

V. Results and Discussion

A. Experimental Setup

To test the system we use three different environments at the University of Trento:

E1: A teaching lab of roughly 7×6 meters with desks and chairs where the 2.4 ISM band is not disturbed, and where we generate controlled interference. The lab is not an anechoic chamber, but it is at the second floor underground, so there is no external interference and all the University access points in the vicinity have been turned off, until no signal could be received with a network analyzer;

E2: The department library, whose topology and Electromagnetic (EM) fingerprint is reported in the upper part of Figure 3;

E3: An area of the department whose topology and EM fingerprint is reported in the lower part of Figure 3.

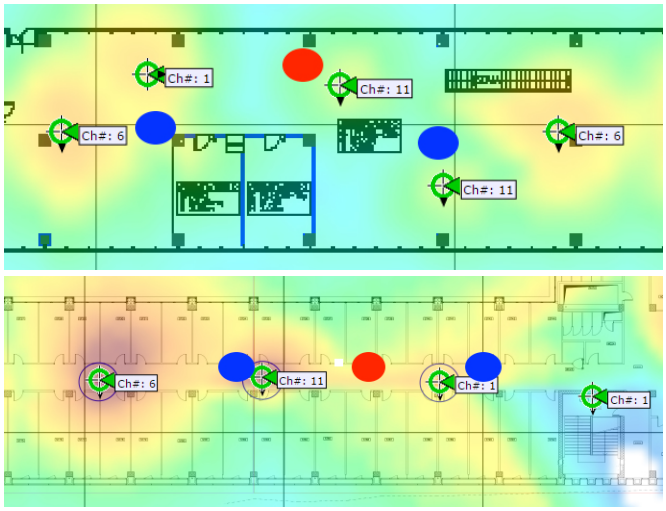

Figure 3. Experimental Environments E2 (the department library, upper plot) and E3 (a portion of the department, lower plot). The red dot is the location of the H-BSS AP while the blue ones are the stations. Other BSS APs are marked with green circles and the channel they use is reported in the adjacent label. The signal intensity form the APs increases from blue to red.

Interference in E1 can be strictly controlled enabling repeatable scientific experiments. To prevent channel capture phenomena that could potentially influence the results of the experiments, the stations were placed accurately at a fair distance between each other. The APs were at the corners of the room while the stations were distributed all around, keeping always a distance of some meters between each other. In this environment we explored many different traffic and interference scenarios, and the H-BSS was tested with up to 8 stations.

IN E2 and E3 instead we work "into the wild:" the interference cannot be controlled, and the other BSS use the channels assigned to them by the university personnel to optimize the coverage and capacity of the network. Unfortunately in these environments we were not able to measure the throughput obtained by the other BSSs, but only the one achieved by the H-BSS.

Stations join the network using the traditional *wpa_supplicant* command. Only after the connection of all the stations the experiments take place. To keep results interpretation easy and to get a better insight we fix the physical transmission speed. Results presented here include 24, 36, and 48 Mbit/s physical speeds as they represent a good mix of modulation and coding schemes. In the controlled E1 environment we have run a very large set of experiments, changing the physical transmission speed (and also checking that the system works with the legacy Direct Sequence Spread Spectrum (DSSS)), the background characteristics and intensity, the number of stations in the H-BSS, and the traffic generated in the H-BSS, collecting data corresponding to several days of operation. In E2 and E3 instead we run shorter tests with only two stations in the H-BSS. Before running any performance evaluation test we checked that the overhead of hopping and measuring is negligible, so that the performance of an H-BSS with saturated

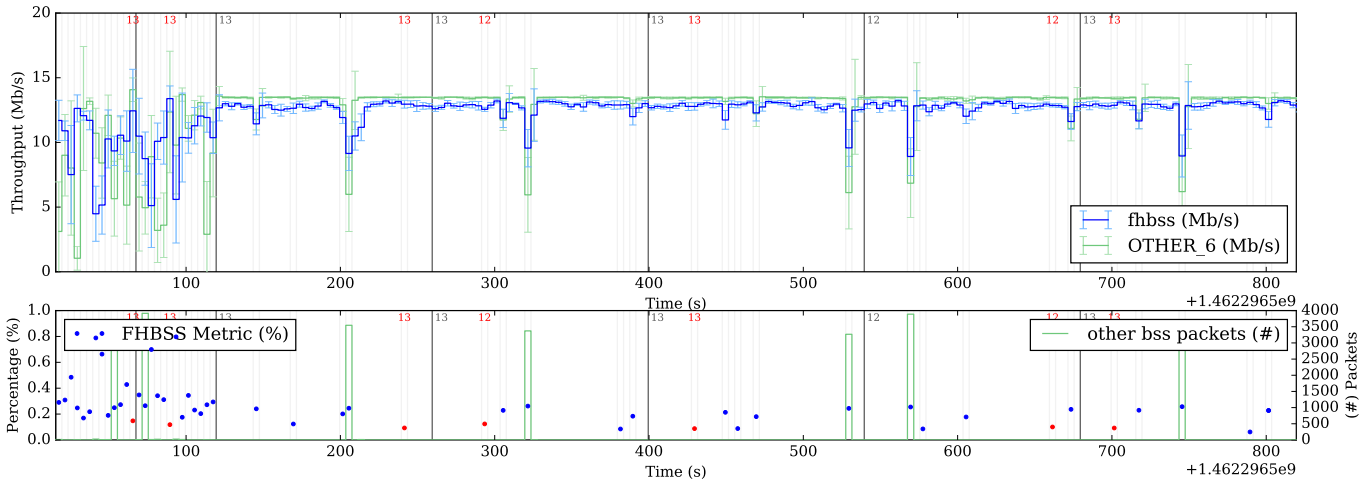

Figure 4. Throughput and metrics obtained by a H-BSS in environment E1 with a background BSS on channel 6. Both offer 14 Mbit/s and the physical rate is fixed at 24 Mbit/s

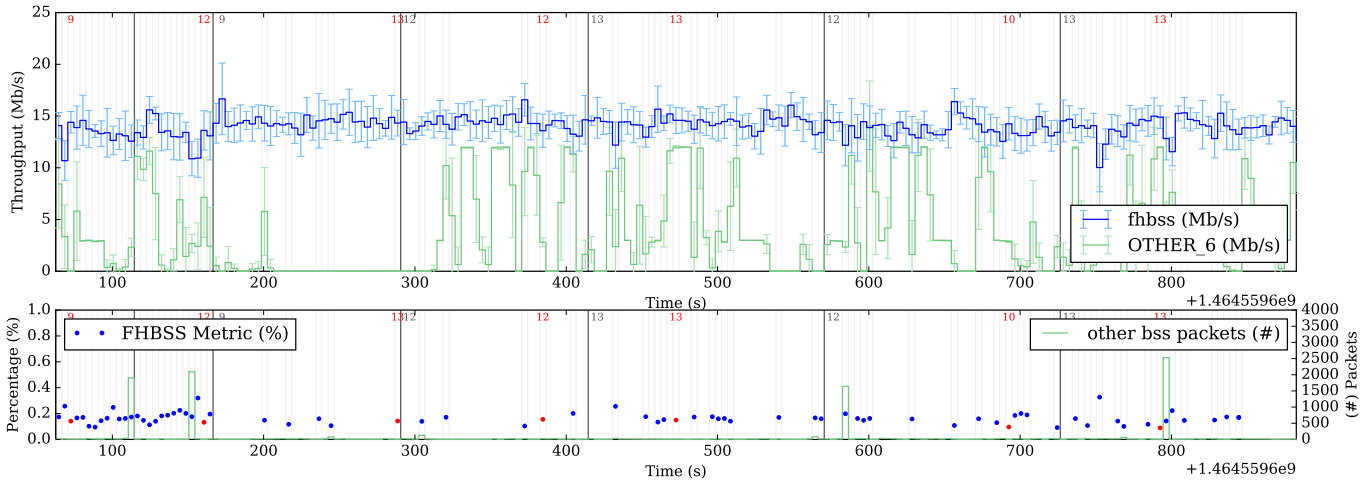

Figure 5. Throughput and metrics obtained by a H-BSS in environment E1 with a background BSS generating variable traffic on channel 6. Both offer and average of 14 Mbit/s and the physical rate is fixed at 24 Mbit/s. The H-BSS includes 8 stations.

traffic and no interference is practically identical to the one of a traditional, static BSS. Both the H-BSS traffic and the interference in E1 are generated with iperf[3]; the background in E2 and E3 is instead the standard traffic from university students and other users.

Unfortunately, space limitations forbids presenting all results; Section V-B presents a selected subset of the measures we took. As a comparison means we have implemented a H-BSS that hops following a random pattern without taking measures and trying to optimize the choice. We are aware that this comparison does not proof that our method is optimal, but at least offers a baseline. Future work include the comparison with the implementation in [4] if the authors will make the implementation available.

[3] https://github.com/esnet/iperf

B. Results

The first set of results we present is a detailed analysis of the hopping cycles and the throughput obtained by a H-BSS when the interference is represented by a single BSS on channel 6.

Figure 4 presents the throughput and the metrics measured during an experiment in saturation with the physical rate fixed at 24 Mbit/s. Both the H-BSS (blue) and the other BSS (green) try to transmit at 15 Mbit/s so that when the two BSS are on the same channel it is completely saturated. The upper plot reports the throughput. The vertical light gray bars indicate the switch from one channel to the other and the black ones delimit cycles. In the first two cycles on the left, the H-BSS randomly probes the channels, without even stopping for a longer time on any of them. It is clear that when the H-BSS stays on one of the channels from 4 to 8 both its performance and that of the other channel suffer. During these two cycles channel 13 is identified as the best one, as indicated by the red number on the top.

The black number on top at the beginning of the cycle starting from cycle three is the channel number in which the H-BSS will spend most of its time during the cycle. Short slots are the probing slots and the best one has the channel number marked with red (other numbers are not reported to avoid cluttering the picture), the best channel identified at cycle n will be used in cycle $n+2$ as explained in Section III-B. When channel 6 is probed the throughput of both BSSs is reduced. As per its design the H-BSS avoids probing channels 4, 5 and 7, 8 as they will be heavily interfered from the traffic in channel 6 with the additional drawback that CSMA/CA does not work properly. Apart from the probing slots the H-BSS remains in non-interfered channels and both BSS work correctly getting a throughput roughly equal to the traffic they offer. The lower picture in Figure 4 presents the H-BSS channel evaluation metrics. The blue dots report the value $(1/M_1)$ while the green line the metric M_2. To avoid numerical instabilities the actual implementation computes $1/M_1$ and not M_1. The red dot is the best metric overall and correspond to the selected channel.

Figure 5 presents the same performance measures and metrics of Figure 4, but in a more demanding setup. The H-BSS has 8 stations, and the background traffic is generated following a 3-state Markov model, as shown by the varying green throughput line. Even with a variable and dynamic interfering traffic, the H-BSS behaves correctly, moving among the least interfered channels, and avoiding the congested one.

Next we present aggregate results in all three the environments. Figure 6 shows the comparison between H-BSS and a random hopping BSS when the background is a BSS with 2 stations offering the same amount of traffic as the H-BSS, which however has 6 stations and not 2. The physical rate is 24 Mbit/s, and the BSS offer 4.5, 7.5, and 13.5 Mbit/s each in three experiments presented. The boxplots report the average throughput (red line), the 25th and 75th percentile of the throughput measured (the boxes themselves) and the minimum and maximum measured (the whiskers). Throughput is measured every second for several minutes and the experiments are repeated 10 times at different times to ensure that they are representative and not the outcome of a specific case. It is clear that not only the H-BSS performs much better than a random hopping sequence, but it also disturb far less the background traffic, thus really using the "white spaces" left by other legitimate traffic. It is important to notice not only the improvement in average and total throughput, but most of all the greater stability and the complete avoidance of "outage" that happen when the random and the background BSS are on adjacent channels (e.g., 5 and 6 respectively), where the interference is very high and the Carrier Sensing Multiple Access with Collision Avoidance (CSMA/CA) protocol does not work properly due to the partial channel overlapping.

Figure 7 shows the boxplot of the throughput obtained by a H-BSS and BSS that performs random hopping in E3, where the traffic offered to the background BSS is mild. Experiments are run similarly to those presented in Figure 6, measuring the throughput every second and repeating measures multiple

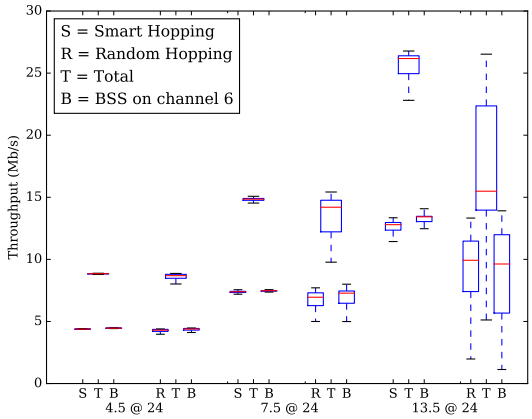

Figure 6. Throughput of hopping and background BSS increasing the load. The physical rate is 24 Mbit/s.

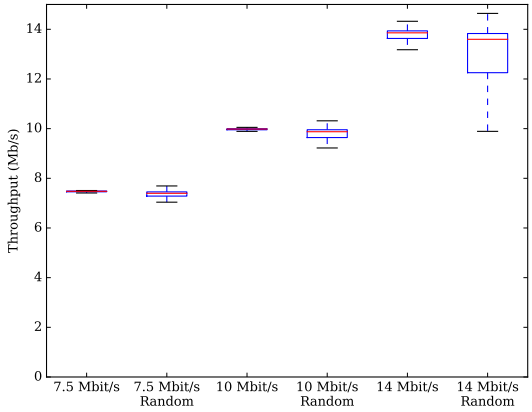

Figure 7. Throughput of H-BSS and random hopping BSS in the department offices (E3).

times at different day times. The hopping BSS offers 1.5, 10, or 14 Mbit/s using respectively a physical rate of 24, 36 or 48 Mbit/s. The better performance and stability of H-BSS is clear, even if not marked due to the low level of interference. Unfortunately we could not measure the "disruption" induced on the background as on the one hand the offered traffic is not known, and on the other hand it has proven impossible to obtain proper synchronization between our BSSs and those of the university system.

Finally, Figure 8 repeats the same experiments of Figure 7 in E2, the department library, where the background traffic is higher due to the presence of many students. In this harsher environment the difference between H-BSS and pure random hopping is striking: The average gain of H-BSS compared to random hopping ranges from 40 to 80%, and the minimum throughput of random hopping is always zero, indicating periods of outage and probably a very high impact on the background traffic.

VI. Conclusion and Future Work

The congestion in ISM bands in certain spots has reached a level that hampers the quality of WiFi communications. The

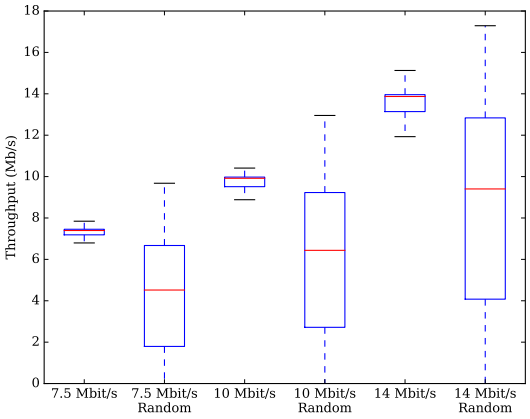

Figure 8. Throughput of H-BSS and random hopping BSS in the department library (E2).

work we presented explored the feasibility of independent, intelligent hopping by an entire 802.11 BSS so as to dynamically avoid crowded or badly interfered channels.

The system proposed has been implemented on standard hardware, and experimental results have shown a very good performance, opening the road for further research and optimization. The work ahead includes improving the channel selection strategy, specially from the dwell time point of view, so that a H-BSS that finds a free channel can stay on it for long times, reaching a more stable behavior. Additionally, the possibility of hopping across different bands (e.g., 2.4 and 5 GHz) is still to be explored, and the hopping protocol can be improved and standardized. On a more theoretical research line, one might want to explore the existence of distributed algorithms that with minimal signaling may achieve optimal dynamic channel hopping strategies.

Acknowledgments

The authors thanks the ICT Services Division (DirSISTI http://icts.unitn.it/) of the University of Trento and in particular Massimiliano Fiorazzo and Alessandro Villani for the support provided in the experimental activity that enabled this contribution. They let us "disturb" the production network, and they "cleared" the spectrum in the lab, a feature fundamental to guarantee that results are reproducible.

References

[1] IEEE Standard 802.11, " Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," Mar. 2012.

[2] K. K. Leung and B. J. Kim, "Frequency assignment for IEEE 802.11 wireless networks," in *58th IEEE Vehicular Technology Conf. (VTC 2003-Fal)*, Oct. 2003, pp. 1422–1426.

[3] M. Yu, A. Malvankar, and W. Su, "A distributed radio channel allocation scheme for WLANs with multiple data rates," *IEEE Trans. on Communications*, vol. 56, no. 3, pp. 454–465, Mar. 2008.

[4] A. Bhartia, D. Chakrabarty, K. Chintalapudi, L. Qiu, B. Radunovic, and R. Ramjee, "IQ-Hopping: Distributed Oblivious Channel Selection for Wireless Networks," in *Proc. of ACM MobiHoc '16*, July 2016, pp. 81–90.

[5] Ericsson, Qualcomm, Huawei, and Alcatel-Lucent, "Study on licensed-assisted access using LTE," September 2014, 3GPP TSG RAN Meeting #65, Edinburgh (Scotland).

[6] E. Ahmed, A. Gani, S. Abolfazli, L. J. Yao, and S. U. Khan, "Channel assignment algorithms in cognitive radio networks: Taxonomy, open issues, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 795–823, 2014.

[7] S. Chieochan, E. Hossain, and J. Diamond, "Channel assignment schemes for infrastructure-based 802.11 WLANs: A survey," *IEEE Communications Surveys & Tutorials*, vol. 12, no. 1, pp. 124–136, 2010.

[8] M. Ihmig and P. Steenkiste, "Distributed dynamic channel selection in chaotic wireless networks," in *13th European Wireless Conf., Paris, France*, 2007.

[9] J. Geier, "Assigning 802.11b access point channels," *WiFi planet*, 2002.

[10] D. Chan, M. Ahmed, A. Hedayat, and B. Hart, "Dynamic bandwidth selection for wide bandwidth wireless local area networks," Mar. 2015, US Patent 8,971,273.

[11] V. Shrivastava, S. K. Rayanchu, S. Banerjee, and K. Papagiannaki, "Pie in the sky: Online passive interference estimation for enterprise wlans." in *NSDI*, vol. 11, 2011, pp. 25–25.

[12] B. Bakhshi, S. Khorsandi, and A. Capone, "On-line joint QoS routing and channel assignment in multi-channel multi-radio wireless mesh networks," *Elsevier Computer Communications*, vol. 34, no. 11, pp. 1342–1360, 2011.

[13] V. Bhandari and N. H. Vaidya, "Channel and interface management in a heterogeneous multi-channel multi-radio wireless network," DTIC Document, Tech. Rep., 2009.

[14] A. Dhananjay, H. Zhang, J. Li, and L. Subramanian, "Practical, distributed channel assignment and routing in dual-radio mesh networks," in *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, 2009, pp. 99–110.

[15] D. Gong, M. Zhao, and Y. Yang, "Distributed channel assignment algorithms for 802.11n WLANs with heterogeneous clients," *Elsevier Jou. of Parallel and Distributed Computing*, vol. 74, no. 5, pp. 2365–2379, 2014.

[16] A. Mishra, S. Banerjee, and W. Arbaugh, "Weighted coloring based channel assignment for wlans," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 9, no. 3, pp. 19–31, 2005.

[17] B. Kauffmann, F. Baccelli, A. Chaintreau, V. Mhatre, K. Papagiannaki, and C. Diot, "Measurement-based self organization of interfering 802.11 wireless access networks," in *26th IEEE Int. Conf. on Computer Communications (INFOCOM)*, 2007, pp. 1451–1459.

[18] D. J. Leith, P. Clifford, V. Badarla, and D. Malone, "WLAN channel selection without communication," *Elsevier Computer Networks*, vol. 56, no. 4, pp. 1424–1441, 2012.

[19] P. Bahl, R. Chandra, and J. Dunagan, "SSCH: slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks," in *ACM 10th Int. Conf. on Mobile Computing and Networking (MobiCom)*, 2004, pp. 216–230.

[20] A. Mishra, V. Shrivastava, D. Agrawal, S. Banerjee, and S. Ganguly, "Distributed channel management in uncoordinated wireless environments," in *ACM 12th Int. Conf. on Mobile Computing and Networking (MobiCom)*, 2006, pp. 170–181.

[21] F. Gringoli, N. Facchi, and D. Berger, "Demo: a testbed to evaluate frequency-hopping anti-jamming techniques in IEEE 802.11," *9th ACM Int. workshop on Wireless Network Testbeds*, 2014, pp. 85–88.

[22] J.-H. Jo and H. Jayant, "Performance evaluation of multiple ieee 802.11 b wlan stations in the presence of bluetooth radio interference," *IEEE Int. Conf. on Communications (ICC)*, 2003, pp. 1163–1168.

[23] D. Croce, D. Garlisi, F. Giuliano, and I. Tinnirello, "Learning from errors: Detecting zigbee interference in WiFi networks," in *13th Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET)*, 2014, pp. 158–163.

[24] P. Salvador, L. Cominardi, F. Gringoli, and P. Serrano, "A first implementation and evaluation of the ieee 802.11 aa group addressed transmission service," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 1, pp. 35–41, 2013.

[25] P. Salvador, V. Mancuso, P. Serrano, F. Gringoli, and A. Banchs, "Voipiggy: Analysis and implementation of a mechanism to boost capacity in IEEE 802.11 WLANs carrying VoIP traffic," *IEEE Trans. on Mobile Computing*, vol. 13, no. 7, pp. 1640–1652, 2014.

[26] D. S. Berger, F. Gringoli, N. Facchi, I. Martinovic, and J. B. Schmitt, "Friendly jamming on access points: Analysis and real-world measurements," *IEEE Trans. on Wireless Communications*, vol. 15, no. 9, pp. 6189–6202, 2016.

[27] L. Sanabria-Russo, J. Barcelo, B. Bellalta, and F. Gringoli, "A high efficiency MAC protocol for WLANs: Providing fairness in dense scenarios," *IEEE Trans. on Networking*, vol. PP, No. 99, 2016.