

# Imola: A Decentralised Learning-driven Protocol for Multi-hop White-Fi

N. Facchi<sup>a</sup>, F. Gringoli<sup>b</sup>, D. Malone<sup>c</sup>, P. Patras<sup>d,\*</sup>

<sup>a</sup>*Department of Information Engineering and Computer Science, University of Trento*

<sup>b</sup>*Department of Information Engineering, CNIT / University of Brescia*

<sup>c</sup>*Hamilton Institute, National University of Ireland Maynooth*

<sup>d</sup>*School of Informatics, University of Edinburgh.*

---

## Abstract

In this paper we tackle the digital exclusion problem in developing and remote locations by proposing Imola, an inexpensive learning-driven access mechanism for multi-hop wireless networks that operate across TV white-spaces (TVWS). Stations running Imola only rely on passively acquired neighbourhood information to achieve scheduled-like operation in a decentralised way, without explicit synchronisation. Our design overcomes pathological circumstances such as hidden and exposed terminals that arise due to carrier sensing and are exceptionally problematic in low frequency bands. We present a prototype implementation of our proposal and conduct experiments in a real test bed, which confirms the practical feasibility of deploying our solution in mesh networks that build upon the IEEE 802.11af standard. Finally, the extensive system level simulations we perform demonstrate that Imola achieves up to  $4\times$  more throughput than the channel access protocol defined by the standard and reduces frame loss rate by up to 100%.

*Keywords:* decentralised medium access, reinforcement learning, multi-hop White-Fi, IEEE 802.11af, prototype implementation

---

## 1. Introduction

Pundits have hailed the recent decisions by communications regulators to allow the unlicensed access of wireless devices to ultra-high frequency (UHF) spectrum as the next Internet disruption [1]. Subsequent pilots have demonstrated that TV white-space technology, despite employing narrow channels (6-8 MHz) often interleaved with digital terrestrial television (DTV) and wireless audio equipment, has significant potential to bridge the digital divide in developing countries [2], as well as in remote and sparsely populated rural areas [3].

Considering the wide communication ranges achievable in sub-gigahertz frequencies due to the superior propagation characteristics these exhibit [4, 5], in this paper we make the case for multi-hop White-Fi networks that build upon the IEEE 802.11af standard [6], as an effective and inexpensive solution for providing broadband coverage to remote and underdeveloped locations. To this end, we propose Imola, a **decentralised learning-driven networking paradigm** that overcomes the performance limitations facing carrier sensing based access mechanisms in multi-hop topologies, **especially when operating in TVWS** [7]. Precisely, as base stations often compete with different numbers of neighbours, while encountering hidden and exposed terminal problems [8, 9], their performance degrades severely and unfairness arises in the absence of appropriate link scheduling.<sup>1</sup> For example, consider a typical multi-hop wireless network where traffic flows traverse two hops to an Internet gateway, as illustrated in Fig. 1. In this topology, stations 1 and 5 do not overhear each other's transmissions, while node 3 carrier senses both of them and regards the channel as mostly busy (exposed). As a result, flow #2 is throttled down at the first hop. Similarly, nodes 2 and 6 are hidden

---

\*Corresponding author. Address: The University of Edinburgh, Informatics Forum, 10 Crichton Street, Edinburgh, EH8 9AB, UK.

*Email addresses:* nicolo.facchi@unitn.it (N. Facchi), francesco.gringoli@unibs.it (F. Gringoli), david.malone@nuim.ie (D. Malone), paul.patras@ed.ac.uk (P. Patras)

<sup>1</sup>Although we particularly focus on White-Fi networks, the protocol we propose could also substitute CSMA-based channel access in multi-hop wireless networks that operate with other technologies.

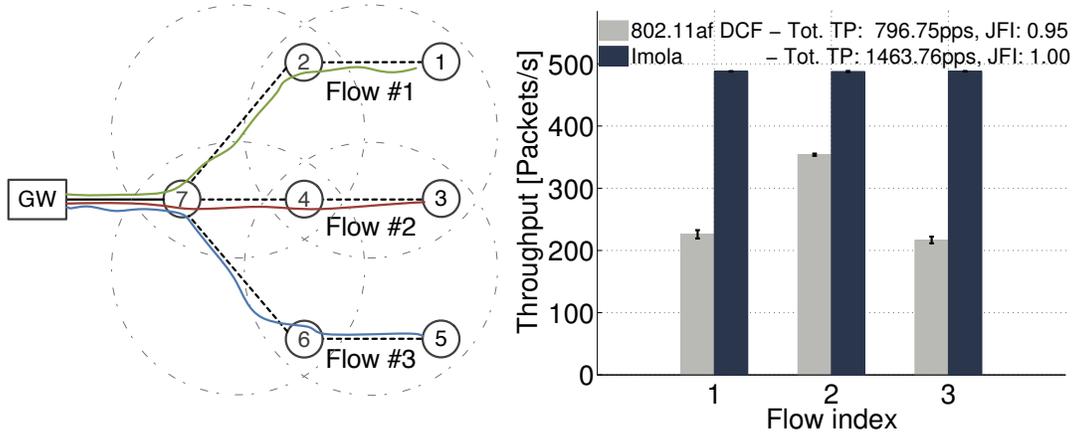


Figure 1: Multi-hop White-Fi network with seven stations (left). Base station 7 has a wired gateway (GW) connection. Nodes within carrier sensing range shown inside dashed-line contours. Three backlogged flows originate at nodes 1, 3, and respectively 5, and terminate at gateway (GW). Per flow throughput (right) when stations run the default 802.11af DCF protocol (light bars) and respectively the scheme proposed in this paper (dark bars). Total throughput and Jain’s Fairness Index (JFI) labelled for each approach. Simulation results.

to one another, but transmit to the same destination (node 7), whereas node 4 carrier senses all of these and thus avoids collisions. As a result, the performance attained by flow #2 becomes superior. We observe indeed that, when channel access is regulated by the 802.11af distributed coordination function (DCF), the throughputs of flows #1 and #3 are each 65% of that of flow #2 (shown with light bars). Also plotted in Fig. 1 with dark bars is the individual performance attained with the scheme introduced in this article, which demonstrates its effectiveness in distributing network resources fairly among competing flows and increasing the total throughput.

Resource allocation in multi-hop wireless networks has been extensively scrutinised in the context of legacy 802.11 networks. Previous works, however, suffer from at least one of the following limitations that challenge their practicality: (i) they assume the network size is known by each node or topology information could be easily propagated throughout the network [10, 11]; (ii) they require precise node synchronisation [12, 13]; and (iii) they rely on cross-layer knowledge of flow paths and/or queue lengths [14–17]. Further, the applicability of multi-channel solutions that seek to alleviate flow starvation (e.g. [18, 19]) is questionable in White-Fi deployments where channel availability is limited and restricted to different geographic locations.

Imola –the scheme we propose– tackles the aforementioned drawbacks of prior works and, to the best of our knowledge, is the first multi-hop White-Fi solution that is designed to avoid chronic unfairness among flows without unnecessarily wasting capacity. Our work makes the following key contributions:

1. We design a link scheduling paradigm not requiring global topology knowledge, control signalling or tight synchronisation. Instead, **Imola only relies on locally available neighbourhood information** acquired passively. We circumvent fairness issues by substituting CSMA with a reinforcement learning algorithm that identifies collision-free transmit opportunities to achieve scheduled-like operation;
2. We devise a simple cycling strategy that ensures base stations do not drift into collisions in steady state operation, when observing different numbers of neighbours;
3. We give a formal analysis of Imola’s convergence properties and outline an effective technique to adapt to network dynamics, i.e. nodes joining/leaving the network;
4. We detail a proof-of-concept implementation of Imola on off-the-shelf hardware and validate its practical feasibility by means of real experiments in simple, yet pathological network conditions;
5. We evaluate the performance of our scheme in multi-hop White-Fi topologies by means of system level simulations, demonstrating significant throughput and fairness gains over the standard 802.11af DCF protocol. Specifically, our proposal yields up to  $4\times$  higher throughput and reduces frame loss rate by up to 100%.

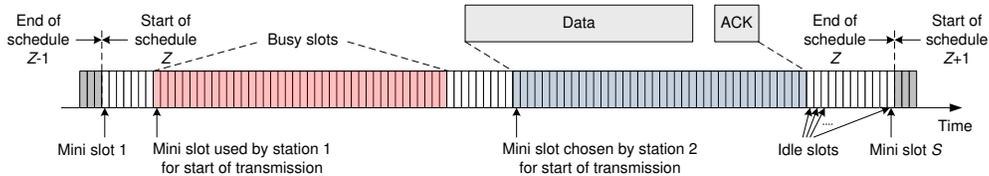


Figure 2: Channel observed by station 2 in a basic network with two Imola nodes. The channel time is regarded as a sequence of schedules of a fixed length, comprising a number of mini slots  $S$ . Stations transmit deterministically at the beginning of a chosen mini slot.

## 2. System Model

We focus on multi-hop White-Fi networks that provide wireless connectivity to underserved locations. We consider infrastructure deployments without structured planning in unlicensed frequency bands, where access points<sup>2</sup> are equipped with a single wireless interface that utilises a single channel or can aggregate multiple TVWS channels for transmission and reception. Envisioned use cases include rural broadband, community networks, temporary networks for emergency rescue operations, and sensing infrastructure for monitoring industrial processes.

To capture the interference between simultaneous transmitters, we consider a protocol model similar to [20]. Although this captures PHY-MAC interactions with some accuracy, it conceals the complexity of interference characterisation and was shown to be adequate if augmented with “reality checks” [21]. Thus we assume two nodes are within the same interference set if their simultaneous transmissions could collide at a destination. This includes not only the cases where stations are within transmission range of one another, but also those where nodes share a common neighbour. We refer to the latter as *two-hop neighbours* and note that this concept covers both hidden and exposed node scenarios.

To eliminate the performance issues associated with carrier sensing listed above, we do not employ this mechanism in our system. Instead, we consider the channel divided into *mini slots* of equal duration,  $\sigma$ , as exemplified in Fig. 2. Any contender  $i$  observes a schedule  $S_i$  that consists of a fixed number of mini slots, approximately proportional to an estimate of their number of (one- and two-hop) neighbours  $n_i$ .

A transmission begins at the start of a mini slot and spans  $T$  slots on average, depending on the duration of the packet being sent. This includes the reception of an acknowledgement, which follows immediately after the packet, in the absence of collisions or channel errors. Within a schedule, each node transmits once and listens for potential receptions during the remaining period, while we note that a generalisation allowing multiple transmissions is also possible. We treat schedules in a circular fashion, i.e. we allow a transmission that starts near the end of a schedule to carry over the slots at the beginning of the following one, meaning that nodes do not need to agree on the beginning of the cycle. This mechanism is illustrated in Fig. 2 for a simple scenario with two stations.

Next, we introduce a networking paradigm that enables nodes to identify mini slots suitable for non-interfering transmissions, and achieve scheduled-like operation.

## 3. Imola: Decentralised Learning-driven Access

We propose Imola, a decentralised carrier sensing free access scheme, that employs reinforcement learning to thwart performance asymmetries in multi-hop White-Fi networks. Imola’s operation involves the following key steps:

- Step 1:** A node  $i$  joining the network listens for transmissions for a duration  $T_{scan}$  and estimates the number of direct and two-hop neighbours,  $n_i$ , by examining the source and destination addresses of overheard frames.
- Step 2:** Given  $n_i$  and the expected transmission duration,  $T$ , the station computes a schedule length,  $S_i$ , that would accommodate all transmissions within the neighbourhood with a minimum idle time in each cycle.

<sup>2</sup>We use the terms access point, base station, and node interchangeably.

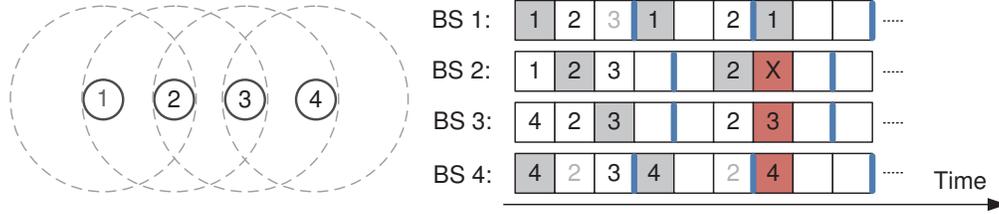


Figure 3: Chain topology example where nodes observing different schedule lengths reach conflicting states. Here  $S_1 = S_4 = 3T$  and  $S_2 = S_3 = 4T$ , as marked right with thick lines, where sequences of transmissions observed by each stations are shown. Shaded boxes depict a station’s action. Transmissions of one- and two-hop neighbours marked with black and respectively grey font on white. Colliding transmissions shown on dark background.

- Step 3:** A slot probability vector  $\mathbf{p}$ ,  $|\mathbf{p}| = S_i$ , is initialised with values  $1/S_i$  and the station chooses randomly one slot to commence transmission.
- Step 4:** If the transmission fails, the probabilities of choosing the same or adjacent slots are decreased; the probabilities of attempting in farther slots are increased.
- Step 5:** The procedure at step 4 is repeated until a successful transmission. Subsequently, the node will transmit in the same set of slots every cycle. Thus the network will reach a state where it operates deterministically, in a scheduled-like fashion.
- Step 6:** Upon a failed transmission starting at a previously successful slot, the probability of that slot is marginally reduced (to accommodate eventual new nodes), while the other probabilities are increased.
- Step 7:** To avoid air time wastage when nodes are switched off, stations attempt to reduce their schedules periodically. Likewise, if new transmitters tamper with existing schedules and cannot be accommodated within a settling period  $T_{set}$ , the schedule length is increased.

In what follows we detail the steps listed above.

### 3.1. Schedule Length Computation

For single-hop topologies where each base station is within transmission range of all neighbours (fully connected networks), the optimal schedule length can be easily computed by ensuring the transmissions of all base stations can fit into a schedule, packed as closely as possible. Thus, for a network with  $N$  nodes and given the expected duration of a transmission  $T$ , the necessary schedule length is  $S = N \cdot T$ .

Choosing the schedule length in multi-hop scenarios, however, poses **two important challenges**:

- (i) base stations that are not within transmission range of one another can collide if their packets arrive simultaneously at the same receiver (two-hop neighbours), and
- (ii) neighbouring stations that transmit deterministically may still collide periodically if they observe different schedule lengths (due to having different numbers of interferers) that are not multiples of each other.

To tackle the first problem, a station joining the network will listen to the medium for a  $T_{scan}$  period and inspect the source and destination addresses of the overheard frames. This will facilitate acquiring not only the list of direct, but also two-hop neighbours that can be identified by the destination MAC addresses present in the overheard acknowledgement frames sent by one-hop neighbours. Thus a node  $i$  having  $n_i - 1$  direct and two-hop neighbours could follow a schedule length  $S_i = n_i T$  that can accommodate all the transmissions within the neighbourhood, without overlaps.<sup>3</sup> A practical question that arises is how to configure  $T_{scan}$ . Although this is a design choice, we work with  $T_{scan} = 10S_{max}$ , where  $S_{max}$  is the maximum schedule

<sup>3</sup>We assume most neighbours will transmit during the discovery phase, though a joining station that settles on a short schedule due to a small estimate, can subsequently double the schedule length to find a set of collision free mini slots for its transmissions.

length that ensures certain one-way delay bounds are met, as we explain below. Typically  $T_{scan} < 1s$ , which guarantees rapid bootstrap.

Choosing a schedule length  $S_i = n_i T$ , however, may be problematic when neighbours  $i$  and  $j$  ( $n_j > n_i$ ) observe

$$n_j \neq \kappa n_i, \kappa \in \mathbb{N}, \kappa > 0,$$

which is formally the second challenge listed above. To understand this better, consider the simple chain topology shown in Fig. 3. In this scenario, since station 1 observes  $S_1 = 3T$  and its direct neighbour (node 2) observes  $S_2 = 4T$ , the transmissions of 1 will not occur with the same frequency as those of 2. Thus, although initially all nodes transmit in non-conflicting slots, 1 will soon transmit to 2, while a transmission of 3 is in progress, which results in a collision. Similarly, the transmission of 3 and 4 will eventually overlap as well.

To overcome this problem, Imola seeks to choose schedule lengths that are a multiple of each other. Since establishing schedules that satisfy this condition precisely across the network would require global topology knowledge, which we want to avoid, stations running Imola compute their schedule lengths as below.

$$S_i = 2^{\lceil \log_2 n_i \rceil} (T + \epsilon), \quad (1)$$

where  $\epsilon > 0$  is a small integer number of mini slots used as guard interval. By the above, a station follows a schedule that can accommodate a maximum number of transmissions that is the power of two value closer to its neighbourhood size  $n_i$ . In addition, if station  $i$  does not settle into collision-free operation, it can double its schedule length  $S_i$  up to a maximum value  $S_{max}$  which we choose such that the one-way delay in the mesh is upper bounded by 150ms, which follows the ITU-T G.1010 recommendation for voice traffic [22]. For typical mesh networks with  $M \leq 5$  hops to the gateway [23],  $S_{max} \leq 30ms$ , which can accommodate up to  $N = 31$  two-hop neighbours transmitting MTU size packets (1500B) over 8MHz channels at 16Mb/s on average. This corresponds to a 5.58 mean node degree, which is inline with previous scalability studies [23].

Notwithstanding that this approach may leave a fraction of the available air time idle, it eliminates the overhead associated with periodic topology information exchanges. Instead, **our solution only relies on passively acquired local information.**

Our aim is to produce an allocation of slots so that for the two-hop neighbour graph there is no interference. If this was a simple vertex colouring problem, then the maximum node degree would be an upper bound for the chromatic number, and so we could calculate a schedule length simply as the product of the maximum node degree and the length of a successful transmission.

If all stations use the same schedule length, then time can be divided into a series of non-overlapping periods, each of which can accommodate a transmission. The number of these periods is greater than the degree, as ensured by (1). In this case, we now see that the problem is a colouring problem, and there are enough periods to ensure a correct colouring exists.

If there is a mix of schedule lengths, then from the perspective of stations with longer schedules, nodes with shorter schedules seem to claim multiple slots. However, stations doing this are in low-degree neighbourhoods of the graph, and so have more freedom to choose slots. In practice, we find that the values of  $S_i$  by (1) result in a feasible allocation for all interference graphs that we tested. Based on results on local node density [24] and Vizing's theorem [25] we conjecture that given a small number of extra periods beyond the maximum node degree, a successful allocation will always be possible.

### 3.2. Channel Access Protocol

We propose a channel access procedure based on reinforcement learning principles, that achieves scheduled-like operation in a decentralised fashion. The underlying idea is to explore, probabilistically, different mini slots for transmission, until successful. Afterwards, the access to the channel is performed deterministically, choosing the same mini slot to trigger transmissions in each schedule. Whenever an attempt fails due to a collision with a (partially) overlapping transmission of a neighbour, the slot probabilities are updated to reduce the likelihood of selecting again that same slot for the start of transmission. The approach we take is similar to [26] as it preserves some channel state information to improve convergence, but differs in that it also reduces the likelihood of attempting to transmit in slots that are adjacent to the one where competition was detected. Note that since we do not employ carrier sensing, nodes are not required to maintain slot

alignment, which reduces protocol complexity when stations that are not within transmission range of each other interfere.

In this protocol, once a station has determined the schedule length  $S$  to be followed<sup>4</sup> (see Sec. 3.1), it initialises the probabilities of all mini slots to the same value, i.e.

$$p_k(0) = \frac{1}{S}, \quad \forall k \in \{1, \dots, S\}$$

and chooses randomly one for the first transmission.

Assume the number of slots  $S$  is even and upon an unsuccessful transmission started in a slot  $j$ , the probabilities are updated as follows:

$$p_k(t+1) = \alpha p_k(t) + (1-\alpha) \frac{2^{\|k-j\|}}{3(2^{S/2}-1)}, \quad (2)$$

where  $\alpha \in (0, 1)$  controls the learning strength and  $\|k-j\|$  is the distance from  $j$  to  $k \bmod S$ . The term with powers of two in the above is to weight slots further from the failure more highly. The rationale is that a transmission spans multiple slots and thus some of those around  $j$  may have also overlapped with the packet of another station. Therefore, we increase the likelihood of exploring a slot,  $k$ , farther from  $j$  upon the next attempt, while reducing the probability of selecting again one of those around  $j$ . We work with an exponentially increasing weight to ensure a reasonably small number of adjacent slots are penalised and use a power of 2 function to meet the constraints imposed by the limited computational capabilities available *inside* commodity network interface cards, as we explain in Sec. 5, where we present a real prototype. Note that, after a collision, the probability of using any slot is at least

$$\gamma = \frac{1-\alpha}{3(2^{S/2}-1)} \quad (3)$$

and observe that  $\gamma > 0$ .

It is essential to verify that all slot probabilities sum to one with the proposed update. Thus we give the following result.

**Lemma 1.** *The probabilities  $p_k$  updated by (2) sum to one.*

*Proof.* Since we know that

$$\alpha \sum_k p_k(t) = \alpha,$$

we need to check that

$$(1-\alpha) \sum_k \frac{2^{\|k-j\|}}{3(2^{S/2}-1)} = 1-\alpha.$$

Focusing on the sum, note that in any cycle there is one slot at distance 0 from  $j$ , two slots at distance 1 from  $j$ , two slots at distance 2 from  $j$ , ..., two slots at distance  $S/2-1$  from  $j$ , and one slot at distance  $S/2$  from  $j$ . Thus, the sum becomes

$$\begin{aligned} & \frac{2^0 + 2(2^1 + 2^2 + 2^3 + \dots + 2^{S/2-1}) + 2^{S/2}}{3(2^{S/2}-1)} = \\ & = \frac{1 + 2(2^{S/2}-2) + 2^{S/2}}{3(2^{S/2}-1)} = \frac{3 \cdot 2^{S/2} - 3}{3(2^{S/2}-1)} = 1. \quad \square \end{aligned}$$

Finally, if a node transmits successfully in slot  $j$  and the subsequent ones, the probabilities become

$$p_j(t+1) = 1; \quad p_k(t+1) = 0, k \neq j.$$

These values are maintained until the node experiences a collision, which triggers a restart of the learning phase. Note that while we aim to decrease the probability of choosing the same slot where a failure occurred,

---

<sup>4</sup>To avoid notation clutter, in what follows we drop the base station index  $i$  when there is no scope for confusion.

we also maintain a lower bound to prevent certain slots from becoming unselectable. Following success in slot  $j$ , after a failed transmission  $p_j$  decreases since following (3),  $\gamma < 1 - \alpha$ . One may also require that after a collision, the probability of all other slots is less than that of the colliding slot, which is achieved by setting  $\alpha \leq 0.5$ . This is also the case in our prototype. Specifically, since we implement the algorithm within the network interface card’s firmware, we work with shift operations instead of multiplications and use  $\alpha = 2^{-K}$ ,  $K > 0$ .

In Sec. 4 we prove formally that, using the above decentralised learning procedure, stations converge to non-interfering slots for transmission.

### 3.3. Schedule Length Adaptation

To ensure the available airtime within a neighbourhood is not underutilised when some APs are switched off (e.g. to save energy in the absence of users), Imola reduces the schedule length by half periodically, thereby exploring the possibility of employing shorter and more throughput efficient schedules.

Executing this operation too frequently may however harm performance. If the new schedule cannot accommodate all stations currently active, especially when neighbourhoods are large and require longer periods to re-converge to collision free sets of slots. To avoid such degradations, we configure the frequency  $f_i$  with which a station  $i$  explores a schedule length  $S'_i = S_i/2$ , such that the opportunity-cost of trying the shorter  $S'_i$  without success remains below a certain percentage of the throughput achieved under the previous setting  $S_i$ .<sup>5</sup> In our design we choose this to be 5%, though other values can be accommodated. If the station settles in  $S'_i$  without collision, it will continue using this length until disturbed (or until subsequently finding a shorter schedule). Otherwise, it reverts to using  $S_i$ .

Suppose a station is expected to settle in a collision free schedule within  $T_{set} > S_i$  and denote  $P$  the number of packets it could transmit in steady state during such a period,<sup>6</sup> if not exploring a shorter schedule, but instead working with the current one  $S_i$ . Then  $P = T_{set}/S_i$  and recall that a station transmits one packet per schedule.

Now assume the worst case situation where a station probes a schedule of length  $S'_i = S_i/2$  after  $zP$  cycles, but cannot find collision free slots after  $T_{set}$ , i.e. another  $P$  cycles. The station can potentially only transmit  $zP$  packets before exploring  $S'_i$ , instead of transmitting  $(z + 1)P$  packets while operating with the longer schedule  $S_i$ . To ensure this cost is below, say, 5% of the throughput achieved with the current  $S_i$ , we need to set  $z$  such that  $zP/(z + 1)P = 95\%$ . That is  $z = 19$  and thus a station will explore a halved schedule once every  $f_i = 19P = 19T_{set}/S_i$  schedules.

## 4. Convergence Analysis

In this section we prove formally that stations running Imola converge to scheduled operation in a decentralised fashion, without message passing, but only relying on the positive acknowledgement mechanism specified by the standard [6]. Specifically, we will show that, regardless of the state, the network can converge to a non-interfering allocation, using the lower bound on slot probabilities after a failure given in (3).

Let  $V$  be the set of nodes and let  $G'$  be the directed interference graph, where  $(i, j)$  is an edge of  $G'$  if  $i$ 's transmissions interfere with  $j$ 's. Let  $G$  be the undirected graph where  $(i, j)$  is an edge if  $i$  interferes with  $j$  or vice versa. We aim to assign to  $i$  a slot  $x_i \in \{1, \dots, S\}$ , where  $i$ 's transmission uses the mini slots  $X_i := \{x_i, x_i + 1, \dots, x_i + T/\sigma\}$ , where recall  $T$  is the expected length of a transmission, and addition is modulo  $S$ . Transmissions from  $i$  and  $j$  overlap if  $X_i \cap X_j \neq \emptyset$ . If transmissions from  $i$  and  $j$  overlap and  $i$ 's transmissions interfere with  $j$ , then  $j$  will detect this as a collision.

Our convergence proof follows that given in the appendix of [27], however, here we are not directly solving a colouring problem. Using the same notation, when a graph colouring problem is phrased as a constraint satisfaction problem, the constraints for an edge  $(i, j)$  become  $\Phi_{(i,j)}(\vec{x}) = 1$  if  $x_i = x_j$  and zero otherwise. We must replace these constraints with  $\Phi_{(i,j)}(\vec{x}) = 1$  if  $X_i \cap X_j \neq \emptyset$  and zero otherwise.

---

<sup>5</sup>Exploring halved schedule lengths ensures that any schedule in the network can always accommodate a power of 2 number of transmissions, in line with (1). Therefore, even with reduced schedules lengths, any two schedules remain multiples of each other, while nodes can avoid periodically drifting into collisions without knowledge of the schedule lengths observed by their neighbours.

<sup>6</sup>Note that the number of packets a station can send in steady state does not depend on  $T_{set}$ , which merely imposes the time allowed for the protocol to reach convergence. This is a design parameter, which in our implementation we set to  $T_{set} = 10S_{max}$ .

**Theorem 1.** *Suppose Imola is run on a network where  $G'$  is strongly connected and at least one feasible allocation exists, then Imola converges almost surely in finite time.*

*Proof.* Let  $A$  be the set of allocations of slots  $\vec{x} = (x_1, x_2, \dots)$  which result in no collisions. Since this set is non-empty, we choose any one of them  $\vec{a}$  and show that there is a positive probability that we arrive at this allocation in a finite number of steps. Note that since this state has no collisions, it will be an absorbing state for Imola.

Let  $\vec{x}(t)$  be the allocation of slots at time  $t$ . Let  $F(t) := \{i : x_i(t) = a_i(t)\}$  be the stations that currently have been allocated their final slot. Let  $U(t)$  be the set of stations that are currently experiencing collisions. We will show that there is a set of events of positive probabilities that lead to  $F(t)$  increasing unless no nodes are in  $U(t)$ . If  $U(t)$  is empty, we have reached an absorbing state with no collisions, so assume  $U(t)$  is non-empty.

At time  $t$  consider two possibilities  $F(t)' \cap U(t) \neq \emptyset$  and  $F(t)' \cap U(t) = \emptyset$ . If  $F(t)' \cap U(t) \neq \emptyset$  then there are unsatisfied nodes who have not chosen their final slot. Let these nodes choose  $x_i(t+1) = a_i$ , while all other nodes choose  $x_i(t+1) = x_i(t)$ . This happens with probability greater or equal to  $\gamma^N$ , and increases the size of  $F(t)$ .

Alternatively, if  $F(t)' \cap U(t) = \emptyset$ , but  $U(t) \neq \emptyset$  we can find a node  $i \in F(t) \cap U(t)$  that is having a collision due to some node  $j \in F(t)'$  (generalising Lemma 4 of [27]). Since  $G'$  is strongly connected, we can find a path in the directed interference graph  $i = v_1, v_2, \dots, v_m = j$  where  $v_k \in V, m \leq N$ . In less than  $m^2$  steps, by propagating the unsatisfied nodes around this loop, we can assign slot  $a_j$  to node  $j$  while leaving all other nodes with their assignment at time  $t$  with probability greater or equal to  $\gamma^{N^3}$ . Thus, we have increased the size of  $F(t)$  by one with positive probability (generalisations of Lemmas 5–7 of [27]).

We conclude that if there are unsatisfied nodes, in at most  $m^2$  steps, we increase  $F(t)$  by one, keeping at least one node, with probability  $\gamma^{N^3}$ . Once all nodes are in  $F(t)$ , we are in an absorbing state. Combining these, a short technical argument gives that Imola converges with probability 1.  $\square$

We note that while the proof above exhibits a specific sequence of events to arrive at a specific schedule, in practice there are multiple paths to multiple feasible schedules.

## 5. Prototype Implementation and Experiments

In this section we present a prototype of our solution, which we developed using commodity wireless hardware, and validate its performance through experiments in a real test bed.

### 5.1. Hardware Platform

For the implementation of Imola we use the Broadcom BCM4318 wireless cards. The motivation behind this hardware choice is twofold: (i) a MAC central processing unit (CPU) controls the card's behaviour in real time by running a dedicated firmware, which is an essential feature given the strict timing requirements of our protocol; and (ii) the device is deployable with the OpenFirmware open-source firmware,<sup>7</sup> which provides enhanced programmability and access to the card's internals, as documented by recent research [28–30]. Although these cards do not support sub-gigahertz operation, which are characteristic to TVWS, our proof-of-concept implementation provides sufficient insights into the feasibility of running Imola in practical multi-hop wireless networks.

In a nutshell, the MAC CPU co-ordinates the internal modules of the device by running the firmware code, i.e. a closed loop that observes various *hardware conditions* and executes the corresponding *handlers* for setting up the circuitry according to the logic of the underlying MAC protocol. For our prototype we exploit two key features of the card that allow to (i) interrupt an ongoing reception and (ii) start the transmission of a frame at a precise time instance, independently of the channel state [31]. Next, we explain how we modified the OpenFirmware DCF code to obtain Imola compliant behaviour.<sup>8</sup>

<sup>7</sup><http://www.ing.unibs.it/openfirmware/>

<sup>8</sup>The source code of our implementation is available at <http://netweb.ing.unibs.it/openfirmware/IMOLA/>

---

**Algorithm 1:** Imola’s firmware level operation.

---

```
1 while device is on do
2   switch condition do
3     case ACK timeout do
4       state ← failure;
5       update  $p_k$  | failed at slot  $j$ ;
6        $j$  ← extract slot according to  $p_k$  distribution;
7        $tx_{time} \leftarrow j \cdot \sigma$ ;
8     case RX complete do
9       if ACK received then
10        if state == failure then
11          state ← success;
12           $p_j \leftarrow 1; p_k \leftarrow 0, \forall k \neq j$ ;
13        else
14          if  $\text{mod}(t_{now} + \Delta_{ACK}, S) < tx_{time}$  then
15            transmit ACK;
16        case  $\text{mod}(t_{now}, S) == tx_{time}$  do
17          if receiving then
18            stop receiver;
19          transmit frame;
20          set ACK timeout;
```

---

## 5.2. Firmware Modifications

The pseudo-code in Algorithm 1 details the three main handlers that underpin Imola, together with the triggering conditions. Besides adapting existing DCF functionality (lines 3 and 8), we implement the *deterministic transmission* mechanism from scratch. The first handler is activated upon the expiration of the acknowledgement (ACK) timeout counter, which forces the system to enter the *failure* state (line 4). Here, Imola also updates the mini slot probability distribution according to (2), taking into account that the last slot used for transmission resulted in failure (line 5). Then it extracts the slot for the next attempt and computes the exact transmission time (lines 6–7). The second handler is executed once the reception of a frame has completed. In case the received frame was an expected ACK following the transmission of a data frame, the system enters the *success* state, assigning the probability 1 to the slot used last and thus continuing to use this in the following schedule (lines 11–12). Otherwise, if a data frame was received, it schedules the transmission of an ACK, but only if the ACK transmission time would not exceed the start of the next scheduled data transmission ( $tx_{time}$ ), i.e.  $t_{now} + \Delta_{ACK}$  comes before  $tx_{time}$  inside the corresponding schedule of duration  $S$ . Finally, we check whether the scheduled transmission time for the next frame in the output queue was reached, by verifying that the clock value  $t_{now}$  matches the precomputed transmission time  $tx_{time}$  within the current schedule (line 16). If the condition is met, we request immediate transmission, interrupting the ongoing reception (if any), and starting the ACK timeout timer (lines 17–20).

In addition to the above functionalities, we implement from scratch the procedures for maintaining and updating the slot probability distribution, and identifying the transmission times. As the MAC CPU has a modest 88MHz clock and a fixed-point arithmetic logic unit (ALU) capable only of addition and subtraction, we configure the relevant parameters as power of two values. This allows to obtain both the current mini slot number  $j$  and the transmission time indicator from the real time hardware clock register, using *shift and mask* instructions. Further, the approach enables us to implement the probability updates as in (2). Finally we transformed the division by 3 either into a left or right shift and mask operation (depending on the sign of  $\|k - j\| - 16$ ) with the following technique:

$$2^{\|k-j\|} / 3 = 2^{[\|k-j\|-16]} 2^{16} / 3 \approx 2^{[\|k-j\|-16]} \cdot 0x5555.$$

Next we summarise the features implemented in the kernel.

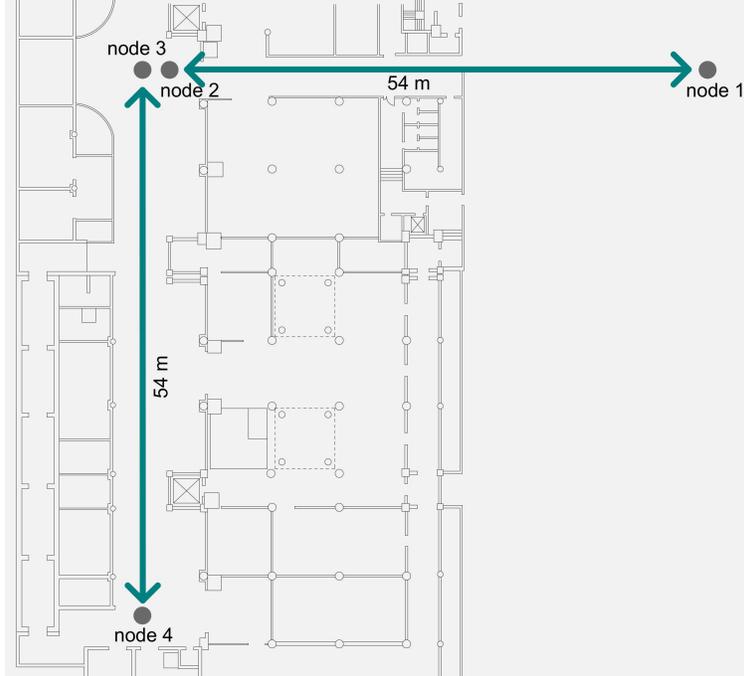


Figure 4: Wireless test bed with four embedded PCs used for experimentation. Bidirectional backlogged UDP traffic is transferred between nodes 1 and 2, and respectively 3 and 4. Nodes 1 and 4 are hidden, 2 and 3 are exposed.

### 5.3. Kernel Space Modifications

At kernel level we alter the `mac80211` and `cfg80211` modules to enable frame exchanging between nodes in a direct-link fashion. We also force devices to operate on channel 14, to ensure our experiments are not subject to external interference. Finally, we modify the `b43` driver to reset the Logical Link Control (LLC) headers of incoming data frames before passing these to upper layers, since our firmware uses the LLC for debugging and tracing purposes, embedding information such as the chosen slot and the frame transmission time within the protocol headers.

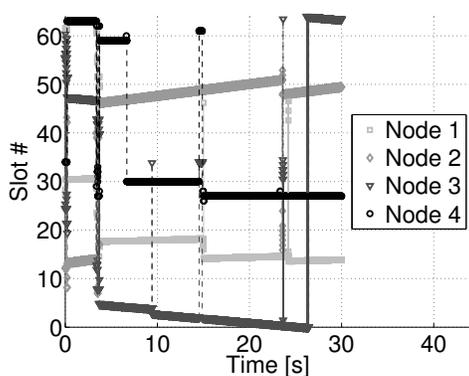
### 5.4. Experiments

To validate our implementation, we conduct experiments in a small test bed that, despite its reduced size, presents problematic hidden and exposed terminal circumstances. More specifically, we deploy four battery powered Alix 2D2 embedded PCs equipped with Broadcom BCM4318 wireless adapters and running Linux kernel 3.14, at the basement level of a university building with concrete walls, as illustrated in Fig. 4.

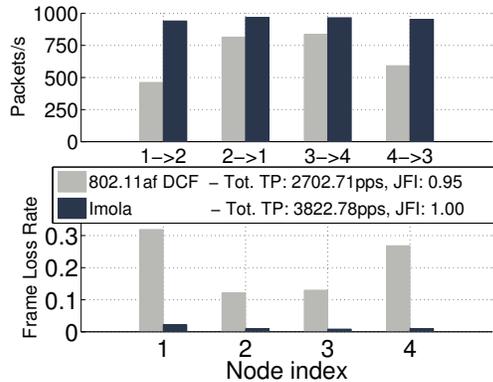
In these experiments each node transmits backlogged UDP traffic using the `iperf` tool, node 1 sending packets to 2, node 4 to 3, and vice-versa. We use a set of Perl scripts to ensure all flows start simultaneously and run several tests, each for a duration of 30 seconds, with stations running both the standard DCF and our Imola prototype.

First we study the convergence properties of our protocol. To this end, we modify the LLC header where we insert at each node the mini slot chosen for the start of transmission and use `tcpdump` to capture the transmitted packets. Then we combine all traces and examine the time evolution of each station's transmission time within a schedule, as seen in the snapshots shown in Fig. 5a. We conclude that, despite clock drift and lack of precise synchronisation, Imola reaches convergence quickly and maintains steady collision free operation, only occasionally changing slots due to clock drift.

Second, we assess the gains of our proposal over the standard protocol. As seen in the upper sub-plot of Fig. 5b, Imola successfully overcomes the hidden and exposed terminal problems, doubling the individual throughputs in some circumstances and ensuring fairness is preserved (JFI=1). In addition, in this set-up our prototype almost eliminates the frame loss rate inherent in CSMA approaches (shown in the lower sub-plot of Fig. 5b). From these experiments, we conclude that Imola is amenable to implementation with off-the-shelf hardware and deployable in White-Fi networks.



(a) Collision-free mini slots chosen by Imola nodes for start of transmission. Clock drift visible after settling.



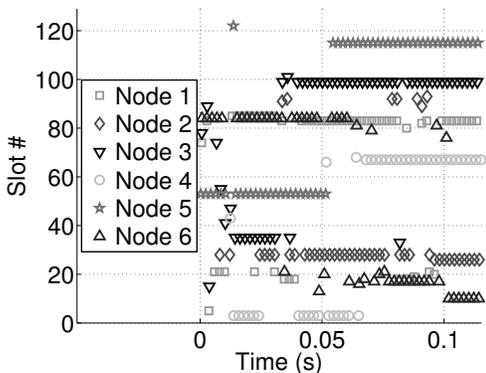
(b) Imola vs 802.11af DCF: per flow (above) and total (labelled) throughput; frame loss rate (below).

Figure 5: Performance analysis with the network test bed shown in Fig. 4, where both hidden and exposed terminals are present. Experimental results.

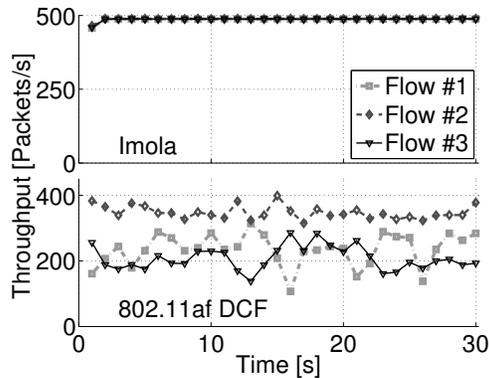
## 6. Performance Evaluation

To evaluate Imola’s performance gains in more complex network scenarios, we implement our protocol in the NS-3 network simulator, which we extensively modify.<sup>9</sup> The key changes we introduce to the simulator’s wireless stack target the `MacLow` and `DcfManager` classes, to enable the channel access management for both Imola and DCF protocols. We implement the deterministic packet transmission mechanism in a dedicated `ImolaManager` class, and the mini slot probabilities update procedure within the newly introduced `ImolaAlgorithm` class. To create a White-Fi network comprising Imola compliant node, we specify the `ImolaWifiMac` class and corresponding helpers.

For all the simulations conducted, we fixed the duration of a mini slot to  $16\mu\text{s}$  and the duration of a frame exchange to 15 mini slots. Thus every station has a transmission budget  $T = 240\mu\text{s}$  per schedule and always has 1000-byte packets to send at 54Mb/s. Note that the same budget could alternatively be used to transmit multiple shorter packets at lower bit rates, or left partially utilised under light loads. We considered losses occur only due to collisions and clock skews are negligible.



(a) Mini slot chosen by each Imola node for their start of transmission.



(b) Evolution of flow throughput with Imola and 802.11af DCF.

Figure 6: Protocol convergence analysis for the tree network topology depicted in Fig. 1. Simulation results.

<sup>9</sup>Vanilla NS-3 is available at <https://www.nsnam.org/>. The patch that enables Imola in NS-3.23 is available at <http://netweb.ing.unibs.it/openfwwf/IMOLA/>.

### 6.1. Protocol Convergence

First we verify that nodes running the Imola protocol converge to collision free slots distributively in larger networks, within a short period of time. To this end, we consider the topology shown in the example of Fig. 1, where three backlogged UDP flows originate at stations 1, 3, and respectively 5, and terminate at the gateway (node 7). To illustrate convergence, in Fig. 6a we plot the time evolution of the mini slot chosen by each node for the start of their transmission.

Observe from the figure that with our proposal all nodes converge to non colliding sets of slots within 100ms of their operation and continue to operate stably after this phase, as reflected in the evolution of the individual flow throughputs shown in the upper sub-plot of Fig. 6b. To add perspective, in the lower sub-plot of Fig. 6b we plot the flow throughputs in the same network, where nodes run the standard 802.11af DCF protocol. Notice that DCF not only exhibits higher variability, but is also unfair as flow #2 attains significant better performance. We conclude that Imola converges quickly to scheduled-like operation and is fair in steady state.

### 6.2. Performance Comparison

Next we illustrate the throughput gains achievable with Imola and the reduction of the frame loss rate that arises due to carrier sensing when hidden and exposed nodes are present, as typically encountered in multi-hop wireless networks. To this end, we consider two multi-hop White-Fi topologies comprising 13 and 9 nodes respectively, as illustrated in Figs. 7a and 8a. The first network is an extended star topology with 3 branches, where the nodes located one hop away from the gateway serve three other base stations each. The second is characteristic for tree-like deployments, where the maximum hop distance to the gateway is

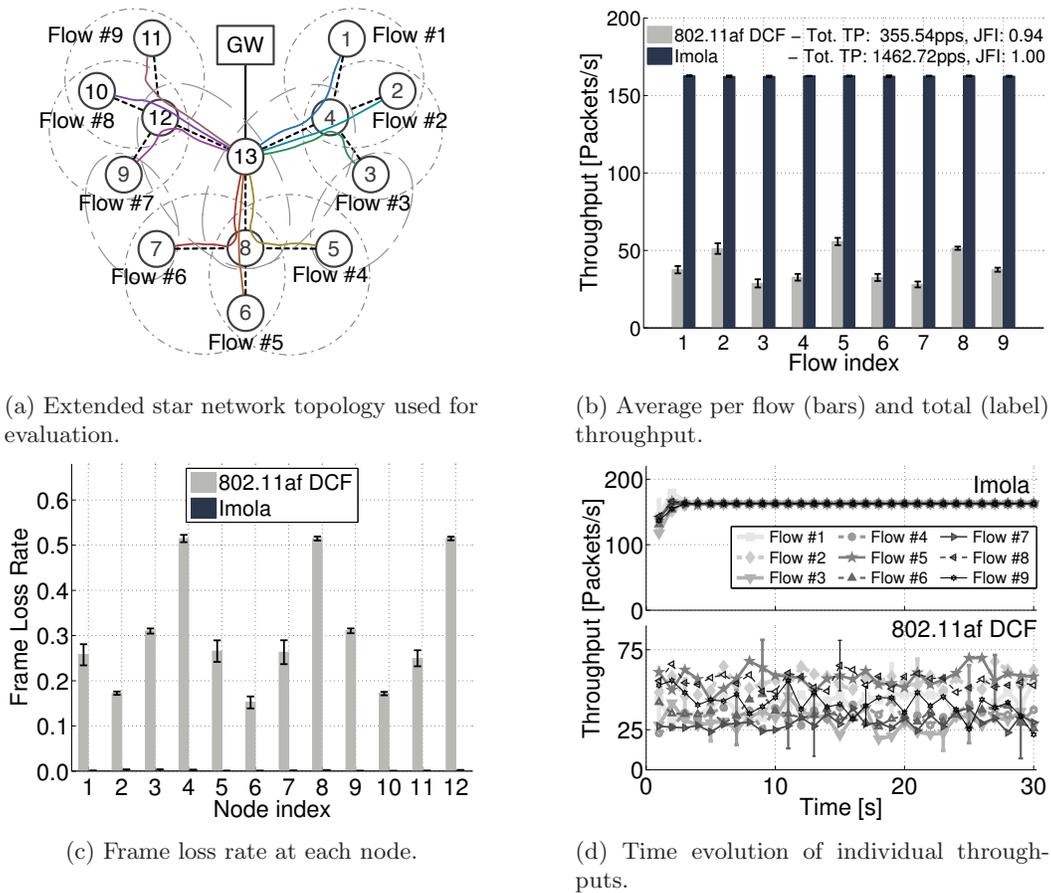


Figure 7: Performance comparison between Imola and standard 802.11af DCF on an extended star topology (nodes within carrier sensing range inside dashed-line contours). All flows are backlogged and are transmitted over UDP. Simulation results.

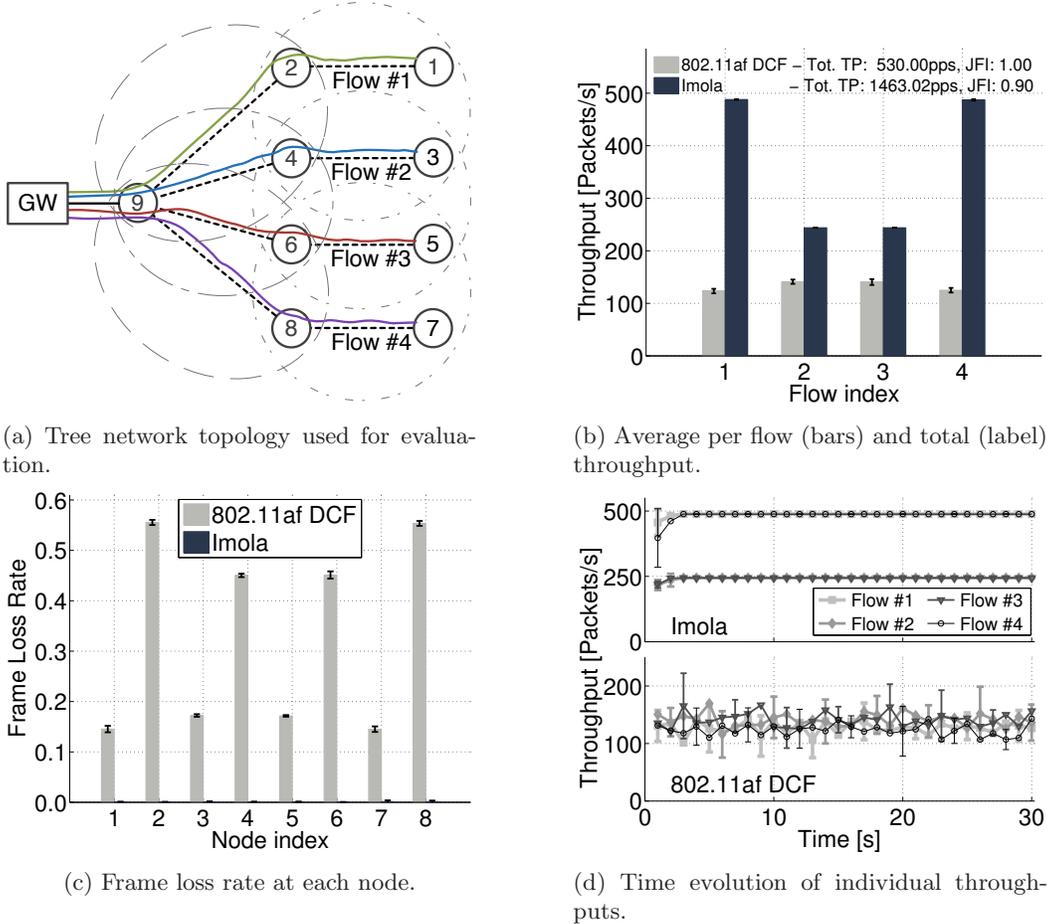


Figure 8: Performance comparison between Imola and standard 802.11af DCF on a 4-branch tree topology (nodes within carrier sensing range inside dashed-line contours). All flows are backlogged and are transmitted over UDP. Simulation results.

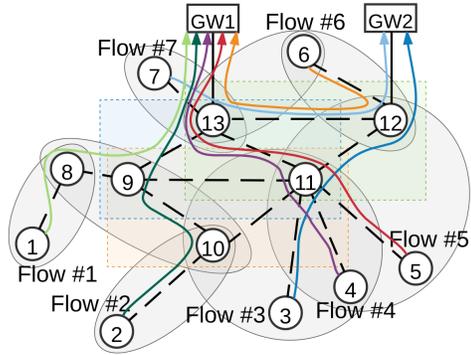
two. Traffic consists of saturated UDP sessions flowing from the “leaf” base stations to the gateway, and forwarded by intermediary nodes.

We compare the performance of individual flows as well as network wide performance, in terms of throughput and fairness (quantified using Jain’s Fairness Index) when the stations run our proposal and the standard DCF protocol respectively. In addition, we examine the frame loss rate experienced by each node in the network, when transmitting to the intended next hop. We run each experiment for a 2-minute duration and repeat all tests 10 times, seeding the pseudo-number generator with different values, to obtain average values of the metrics of interest with good statistical significance.

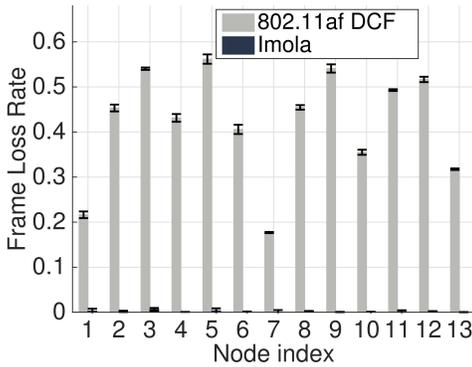
As shown in Figs. 7b and 8b, Imola increases individual flow throughputs significantly and equalises these when stations observe similar numbers of two-hop neighbours (and thus schedule lengths), yielding a JFI of 1 (Fig.7b). In contrast, with 802.11af DCF the JFI drops to 0.94 in the first scenario, which is reflected in the individual throughputs that are almost twice that of competing flows (e.g. flow #5 vs. flow #7). From a network wide perspective, in the extended star topology tested, Imola achieves a fourfold increase in total throughput.

On the other hand, when the neighbourhood size of some nodes is smaller (as in the case of Fig. 8), our approach avoids unnecessarily wasting channel time, which is seen in this case in the superior throughput values attained by flows #1 and respectively #4 (Fig. 8b). Note here that, although flow throughputs are not equalised with Imola in such circumstances, the performance of flows throttled by DCF is increased by 60% when using our approach. Further, Imola attains 175% gains in terms of total network throughput.

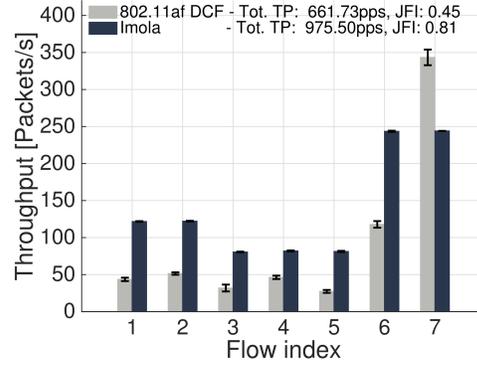
We attribute the significant performance gains Imola exhibits in both scenarios to the fact that our proposal overcomes the conundrums inherent in carrier sensing based access protocols, eliminating almost



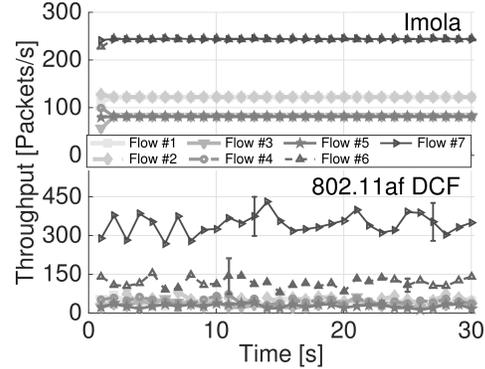
(a) Mesh network topology used for evaluation.



(c) Frame loss rate at each node.



(b) Average per flow (bars) and total (label) throughput.



(d) Time evolution of individual throughputs.

Figure 9: Performance comparison between Imola and standard 802.11af DCF on a mesh topology with 2 gateways (nodes within carrier sensing range inside shaded areas). All flows are backlogged and are transmitted over UDP. Simulation results.

completely the frame loss rate at each node, as depicted in the 30 second snapshots of Figs. 7c and 8c. Notice that with DCF the loss rate exceeds 50% at some nodes, which is noteworthy, while our scheme reduces this by 100% in all cases. In addition, the scheduled like operation we achieve eliminates throughput variability and boosts performance, as illustrated in Figs. 7d and 8d.

We now turn attention to a more general multi-hop topology which we obtain using the Cerdà-Alabern model [32]. This is known to produce graphs whose properties have been derived from real wireless community networks. The network we consider in our case is depicted in Fig. 9a, where observe the complex overlap of the stations carrier sensing ranges, and the 7 flows routed to two different gateways. In this scenario, Imola doubles the throughput performance of all flows, except for that of Flow #7, which experiences a 25% reduction with our approach. However, flow #7 maintains a superior level of performance as compared to other flows traversing nodes with more neighbours, while we improve fairness as we ensure #7 fairly shares resources with #6 (Fig. 9b). Again, the network sees an almost 50% overall throughput gain, which is largely due to the fact that Imola reduces dramatically frame loss, as shown in Fig. 9c. Moreover, our approach converges within a couple of seconds to scheduled-like operation, circumventing throughput variations inherent in random access protocols (see Fig. 9d).

From these simulation results, we conclude that Imola attains substantial performance gains in multi-hop White-Fi networks, while only operating with local neighbourhood information that can be acquired passively.

### 6.3. TCP Traffic

Lastly, we assess Imola's performance gains over 802.11af DCF under TCP traffic. To this end, we consider the simpler network topology in Fig. 1, this time with backlogged TCP flows originating at stations

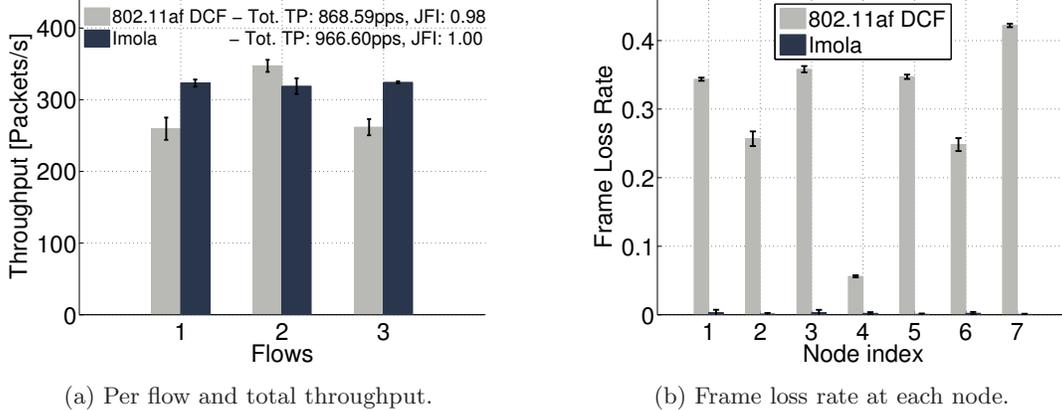


Figure 10: Performance comparison between Imola and 802.11af DCF when backlogged data flows are transmitted over TCP in the network topology depicted in Fig. 1. Simulation results.

1, 3, and 5. For comparison, we examine again the average per flow and total throughput, as well as Jain’s Fairness Index. The congestion control algorithm employed at the end hosts is TCP New Reno.

The results of these experiments are illustrated in Fig. 10. As seen in the figure, Imola attains  $\approx 12\%$  more throughput in this scenario and equalises flow throughputs (JFI=1), while dramatically reducing the frame loss rate. We note that further throughput improvements would be possible, by aggregating TCP acknowledgements within a single transmission at the last hop or piggybacking these on data link ACKs [29]; however, we leave such improvements for future work.

## 7. Related Work

**Scheduling in Multi-Hop Wireless Networks** received significant attention in the past [10–16, 33]. Eryilmaz *et al.* put forward a framework for developing distributed scheduling schemes for such networks. Stations’ attempt probabilities are adapted in [10] to approach an optimal throughput bound. The minimum number of slots required to schedule a set of links is studied in [11] and a distributed algorithm with approximate logarithmic complexity is given. Bui *et al.* specify a class of distributed algorithms proven to achieve a fraction of the capacity region [15]. An 801.11 overlay TDMA MAC is proposed in [13], which relies on network wide synchronisation. Recently, Sim *et al.* introduced a decentralised learning approach to schedule transmissions over pseudo-wired mm-wave links when, similarly to our approach, carrier sensing is bypassed, to tackle terminal deafness problems that arise with directional transceivers [33].

**Unfairness Mitigation in Multi-hop Networks** was studied in [17–19]. Shi *et al.* employ a control and multiple data channels to tackle flow starvation [18]. Scheduling in multi-channel multi-hop networks is formulated as convex optimisation problem and solved in [19]. Similar to our approach, XPRESS disables carrier sensing, but employs backpressure scheduling, relying on cross-layer interactions, to achieve close to perfect fairness in small-scale *centralised* WLANs [17]. Recently Liu *et. al* put forward an opportunistic scheduling approach to balance fairness and throughput performance in multi-hop wireless ad hoc networks [34]. The solution assumes an underlying carrier sensing based protocol, which is inherently prone to exposed and hidden terminal problems. Channel asymmetries in a multi-hop networks are tackled with a network coding approach in [35], though its applicability to data intensive scenarios is questionable, given the purpose-built low-rate nature of the proposed platform.

**Learning MAC Protocols** were proposed to enable deterministic channel access and improve the performance of *single-hop* WLANs [26, 36–41]. Barcelo *et al.* introduce CSMA/ECA, whereby 802.11 stations backoff for a fixed number of slots upon success, and revert to random access upon failure [39, 40]. Fang *et al.* refine this design, adding state memory to speed up convergence [26]. SRB implements a similar approach, but adapts the semi-random access procedure to the number of active stations [36]. Zame *et al.* give perfect coordination protocols that can achieve collision-free operation with less feedback. Chu *et al.* adopt a Q-Learning algorithm that switches an ALOHA network from random access to perfect scheduling [38]. In a similar fashion, a semi-distributed backoff (SDB) algorithm introduced in [41] performs opportunistic migration from random to deterministic backoff in WLANs. However, these approaches do not work in

multi-hop topologies as they require all stations to be within the same collision domain and follow the same slot boundaries.

In contrast to the aforementioned works, the solution we propose does not require global topology knowledge, precise synchronisation, or cross-layer interactions. Unlike previous learning approaches, Imola does not rely on channel state information and is provably effective in pathological multi-hop scenarios, as we recently showcased in [42] and detailed in this paper. Given the limited availability of TVWS channels, our proposal is particularly suitable for multi-hop White-Fi.

## 8. Conclusions & Future Work

In this paper we introduced Imola, a decentralised protocol for multi-hop White-Fi networks that circumvents the performance conundrums specific to carrier sensing based access mechanisms. With our approach, stations use reinforcement learning to achieve scheduled-like operation, without requiring synchronisation and periodic exchange of topology information. We detailed a proof-of-concept implementation on commodity hardware and reported experimental results that confirm Imola’s practical feasibility. Finally, simulation results demonstrated that our scheme achieves up to  $4\times$  more throughput than the standard 802.11af DCF and reduces frame error rate by up to 100%, in several multi-hop topologies.

To gain further insights into Imola’s practicality, future work will pursue additional evaluation in scenarios where nodes are switched off/turned on in response to varying traffic load and investigate the co-existence with legacy 802.11af stations, as well as other wireless technologies, including 802.15.4m WPAN and 802.22 WRAN.

## Acknowledgements

The research leading to these results has received funding from the European Commission grant no. H2020-645274 (EU WISHFUL project), was supported in part by a grant from Science Foundation Ireland (SFI), and co-funded under the European Regional Development Fund under grant no. 13/RC/2077. The University of Edinburgh is authorised to reproduce and distribute reprints and online copies for their purposes notwithstanding any copyright annotation hereon.

## References

- [1] TechRepublic, White space, the next internet disruption: 10 things to know, <https://tinyurl.com/NextInternetDisruption> (Mar. 2014).
- [2] The Hindu, Microsoft pitches for ‘White-Fi’ to provide last mile connectivity, <https://tinyurl.com/LastMileConnect> (Dec. 2014).
- [3] The Scotsman, Rural Scotland to lead with white space internet, <https://tinyurl.com/RuralScotland> (Oct. 2014).
- [4] B. Radunović, R. Chandra, D. Gunawardena, Weeble: enabling low-power nodes to coexist with high-power nodes in white space networks, in: Proc. ACM CoNEXT, Nice, France, 2012.
- [5] A. Giannoulis, P. Patras, E. Knightly, Mobile access of wide-spectrum networks: Design, deployment and experimental evaluation, in: Proc. IEEE INFOCOM, Turin, Italy, 2013.
- [6] IEEE Std 802.11af, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Amendment 5: Television White Spaces (TVWS) Operation, 2013.
- [7] P. Ren, Y. Wang, Q. Du, J. Xu, A survey on dynamic spectrum access protocols for distributed cognitive wireless networks, EURASIP Journal on Wireless Communications & Networking 60.
- [8] A. Tsertou, D. Laurenson, Revisiting the hidden terminal problem in a CSMA/CA wireless network, IEEE Trans. on Mobile Computing 7 (7) (2008) 817–831.

- [9] D. Vassis, G. Kormentzas, Performance analysis of IEEE 802.11 ad hoc networks in the presence of exposed terminals, *Ad Hoc Networks* 6 (3) (2008) 474–482.
- [10] C. Joo, N. B. Shroff, Performance of random access scheduling schemes in multi-hop wireless networks, *IEEE/ACM Transactions on Networking* 17 (5) (2009) 1481–1493.
- [11] M. M. Halldórsson, P. Mitra, Nearly optimal bounds for distributed wireless scheduling in the SINR model, *Distributed Computing* (2014) 1–12.
- [12] A. Behzad, I. Rubin, Optimum integrated link scheduling and power control for multihop wireless networks, *IEEE Trans. on Vehicular Technology* 56 (1) (2007) 194–205.
- [13] P. Djukic, P. Mohapatra, Soft-TDMAC: A software-based 802.11 overlay TDMA MAC with microsecond synchronization, *IEEE Trans. on Mobile Computing* 11 (3) (2012) 478–491.
- [14] P.-J. Wan, O. Frieder, X. Jia, F. Yao, X. Xu, S. Tang, Wireless link scheduling under physical interference model, in: *Proc. IEEE INFOCOM, Shanghai, China, 2011*.
- [15] L. Bui, S. Sanghavi, R. Srikant, Distributed link scheduling with constant overhead, *IEEE Trans. Netw.* 17 (5) (2009) 1467–1480.
- [16] A. Eryilmaz, O. Asuman, E. Modiano, Polynomial Complexity Algorithms for Full Utilization of Multi-Hop Wireless Networks, in: *Proc. IEEE INFOCOM, Anchorage, USA, 2007*, pp. 499–507.
- [17] R. Laufer, T. Salonidis, H. Lundgren, P. Le Guyadec, A cross-layer backpressure architecture for wireless multihop networks, *IEEE Trans. Netw.* 22 (2014) 363–376.
- [18] J. Shi, T. Salonidis, E. W. Knightly, Starvation mitigation through multi-channel coordination in CSMA multi-hop wireless networks, in: *Proc. ACM MobiHoc, Florence, Italy, 2006*.
- [19] S. Merlin, N. Vaidya, M. Zorzi, Resource allocation in multi-radio multi-channel multi-hop wireless networks, in: *Proc. IEEE INFOCOM, Phoenix, USA, 2008*.
- [20] P. Gupta, P. Kumar, The capacity of wireless networks, *IEEE Trans. on Information Theory* 46 (2) (2000) 388–404.
- [21] Y. Shi, Y. T. Hou, J. Liu, S. Kompella, How to correctly use the protocol interference model for multi-hop wireless networks, in: *Proc. ACM MobiHoc, New Orleans, USA, 2009*.
- [22] ITU-T, Recommendation g.1010 : End-user multimedia qos categories (Nov. 2001).
- [23] J. Lessmann, A. de la Oliva, C. Sengul, A. Garcia, M. Kretschmer, S. Murphy, P. Patras, On the Scalability of Carrier-grade Mesh Network Architectures, in: *Proceedings of Future Network & Mobile Summit, Warsaw, Poland, 2011*.
- [24] T. Moscibroda, R. Wattenhofer, Coloring unstructured radio networks, *Distributed Computing* 21 (4) (2008) 271–284.
- [25] J. Misra, D. Gries, A constructive proof of Vizing’s theorem, *Information Processing Letters* 41 (3) (1992) 131–133.
- [26] M. Fang, D. Malone, K. R. Duffy, D. J. Leith, Decentralised learning MACs for collision-free access in WLANs, *Wireless Networks* 19 (2013) 83–98.
- [27] A. Checco, D. Leith, Learning-based constraint satisfaction with sensing restrictions, *IEEE Journal of Selected Topics in Signal Processing* 7 (5) (2013) 811–820.
- [28] P. Salvador, L. Cominardi, F. Gringoli, P. Serrano, A first implementation and evaluation of the IEEE 802.11aa group addressed transmission service, *SIGCOMM Comput. Commun. Rev.* 44 (2014) 35–41.

- [29] P. Salvador, V. Mancuso, P. Serrano, F. Gringoli, 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* 13 (7) (2014) 1640–1652.
- [30] P. Patras, H. Feghhi, D. Malone, D. Leith, Policing 802.11 MAC Misbehaviours, *IEEE Transactions on Mobile Computing*.
- [31] D. S. Berger, F. Gringoli, N. Facchi, I. Martinovic, J. B. Schmitt, Friendly jamming on access points: Analysis and real-world measurements, *IEEE Transactions on Wireless Communications* 15 (9) (2016) 6189–6202.
- [32] L. Cerdà-Alabern, On the topology characterization of Guifi.net, in: *IEEE Intl Conf. on Wireless and Mob. Comp., Netw. and Comms (WiMob)*, 2012.
- [33] G. H. Sim, R. Li, P. Patras, C. Cano, D. Malone, J. Widmer, Learning from experience: Efficient decentralized scheduling for 60ghz mesh networks, in: *Proc. IEEE WoWMoM, Coimbra, Portugal*, 2016.
- [34] Q. Liu, K.-C. Leung, V. O. Li, Z. Zhao, G. Yang, L. Cui, Fairness and high-throughput scheduling for multihop wireless ad hoc networks, *Ad Hoc Networks* 52 (2016) 195–206.
- [35] A. Paramanathan, S. Thorsteinsson, D. E. Lucani, F. H. Fitzek, On bridging theory and practice of inter-session network coding for CSMA/CA based wireless multi-hop networks, *Ad Hoc Networks* 24, Part B (2015) 148–160.
- [36] Y. He, R. Yuan, J. Sun, W. Gong, Semi-random backoff: Towards resource reservation for channel access in wireless lans, in: *Proc. IEEE ICNP*, 2009.
- [37] W. Zame, J. Xu, M. van der Schaar, Winning the lottery: Learning perfect coordination with minimal feedback, *IEEE Journal of Selected Topics in Signal Processing* 7 (5) (2013) 846–857.
- [38] Y. Chu, S. Kosunalp, P. D. Mitchell, D. Grace, T. Clarke, Application of reinforcement learning to medium access control for wireless sensor networks, *Engineering Applications of Artificial Intelligence* 46 (2015) 23–32.
- [39] J. Barcelo, A. Toledo, C. Cano, M. Oliver, Fairness and Convergence of CSMA with Enhanced Collision Avoidance (ECA), in: *Proc. IEEE ICC*, 2010.
- [40] L. Sanabria-Russo, J. Barcelo, B. Bellalta, F. Gringoli, A high efficiency mac protocol for wlans: Providing fairness in dense scenarios, *IEEE Transactions on Networking*.
- [41] S. Misra, M. Khatua, Semi-distributed backoff: Collision-aware migration from random to deterministic backoff, *IEEE Transactions on Mobile Computing* 14 (5) (2015) 1071–1084.
- [42] N. Facchi, F. Gringoli, D. Malone, P. Patras, When Is the Right Time to Transmit in Multi-hop White-Fi?, in: *Proc. IEEE WoWMoM, Coimbra, Portugal*, 2016.