Link Quality Characterization in IEEE 802.11s Wireless Mesh Networks

by

Mohamed Riduan Abid

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
December 13, 2010

Keywords: Wireless mesh networks, IEEE 802.11s, hybrid wireless mesh protocol, Airtime, link quality, loss rate, physical interference model, ping-pong effect

Approved by

Saâd Biaz, Chair, Associate Professor, Computer Science and Software Engineering
Alvin Lim, Associate Professor, Computer Science and Software Engineering
David A. Umphress, Associate Professor, Computer Science and Software Engineering

Abstract

HWMP (Hybrid Wireless Mesh Protocol) has been set as the default routing protocol for the ongoing IEEE 802.11s WMN (Wireless Mesh Network) standard. Unlike most multi-hop routing protocols, which operate at the network layer, HWMP operates at the MAC layer and uses IEEE 802.11s Airtime as a routing metric.

In this dissertation, and in an attempt to delve into the subtleties of this new promising technology, we started first by deploying a real-world pre-IEEE 802.11s indoor WMN testbed that spans two separate buildings, and implements the main traits of this new IEEE standard, e.g., architecture, HWMP, and Airtime. We ascertained the practicality of the testbed by testing real-world traffic such as having Wi-fi enabled stations browsing the Internet, using the deployed WMN as a backhaul. We identified major practical issues and addressed them, e.g., clients association, Internetworking, and supporting multiple gateways. To encourage the use of real-world WMN testbeds, and to cope with the scarcity of such testbeds, we made the implementation open-source and available online.

IEEE 802.11s Airtime metric depends on the observed loss and transmission rates. However, IEEE 802.11s did not delineate a specific way to measure the loss rate; It is left as an open research issue. To estimate loss rate, most routing metrics, e.g., *ETX* and *ETT*, use the *broadcast* approach whereby broadcast probe frame losses are measured. Such an approach suffers fundamental shortcomings which were addressed by introducing the *passive* approach. This latter uses data traffic frames instead of broadcast frames. However, the *passive* approach still suffers from the major shortcoming of probing idle links. In this dissertation, we propose a novel loss rate estimation scheme that overcomes the shortcomings of both broadcast and passive approaches. The proposed scheme estimates losses by tracking their *causing* events, mainly contention and interference. However, interference measurement remains a challenging task merely because of the

impracticality of the current *interference models*. We propose a novel practical interference measurement framework that adapts the physical interference model and uses a probabilistic model for frame arrivals. A new interference-aware routing metric, ICE (*Interference and Contention Estimator*), is shaped and compared to *ETX* and IEEE 802.11s *Airtime* routing metrics using a real-world pre-IEEE 802.11s WMN testbed. ICE outperforms both ETX and IEEE 802.11s Airtime with an average of 16%. This outperformance is a significant improvement when taking into account that we were using a single-channel radio approach.

While experimenting with HWMP and Airtime, we noticed a "ping-pong" effect in the behavior of the network. The very few references to this effect in the literature condemn it as a perilous behavior but only superficially address the problem. In this dissertation, we also present a thorough study of the IEEE 802.11s Airtime ping-pong effect, and we highlight its correlation to the underlying rate control algorithms. Using different rate control algorithms (e.g., ARF, AARF, ONOE, AMRR and Constant rate), we prove that transmission rate adaptation is the principal cause behind the effect. We show that the effect is an inherent behavior and not necessarily a perilous one and that an accurate characterization of the effect will improve network performance. We present and discuss a novel research direction consisting of shaping *ping-pong-aware* mechanisms that, by detecting when a link undergoes such an effect, adapt the network resources for a better performance. In this context, we present a ping-pong-aware mechanism that is O(1), decentralized, and can be easily integrated into the IEEE 802.11s routing protocol. Using extensive ns-3 simulations, we show that the new mechanism has an average outperformance of 5%. This latter is a slight improvement; However, it remains as a proof of concept for future research in the direction of shaping better ping-pong-aware mechanisms.

Acknowledgments

I sincerely thank my supervisor Dr. Saâd Biaz for providing me with the opportunity to pursue my Ph.D. dream. What I learned from him is not only the knowledge, but the attitude towards research and academia in general. I am highly grateful for his advice during my research, and also for his encouragement during my frustration. Guidance from my committee members Dr. Lim and Dr. Umphress, and outsider reader Dr. Reeves were valuable and worth pursuing. I thank Dr. Bensaid, my teacher and dean, for encouraging me to make this dream happen. I have been blessed with the most wonderful and supportive family. There is little I can say to thank you all. This dissertation would not have been possible without the support, love, and devotion of my parents and my wife Hanaa. I thank God for granting me the miracle, my beloved daughter, who provides sunshine and joy to my life. To you, Nourane, I dedicate this dissertation, and thanks to Him.

Table of Contents

List of Figures

Chapter 1

Introduction

Due to their easy-to-deploy, self-healing, and reduced-cost features, WMNs [1] are emerging as a promising technology, and are attracting considerable attention in academia and industry as well. WMNs are easy-to-deploy as setting a WMN involves minimal wiring and configuration overhead: Placing the WMN nodes and powering them *On* is all that is required to operate a WMN. On the other hand, WMNs are self-healing thanks to the redundancy of wireless links: A failure in a wireless link incites the network to seek an alternative operational link, thus maintaining the network operational.

WMNs may serve a rich set of applications, e.g., wireless community networks, wireless enterprise networks, transportation systems, home networking and last-mile wireless Internet access. Providing last-mile wireless Internet access is one of the most promising applications as WMNs tremendously reduce the cost and the configuration overhead when compared with current solutions, e.g., Wi-Fi (IEEE 802.11) LANs [56]. In fact, the proliferation of Wi-Fi LANs played a tremendous role in the promising success of WMNs. However, and even though very successful, Wi-Fi LANs still suffer from the cost and the overhead of setting the wired backbone that connects the different APs (Access Points). In IEEE 802.11s WMNs, no wiring is required except for the AP which will serve as a gateway towards the Internet, even though the gateway AP can be wirelessly connected to the Internet. All other APs serve as routers and route data on behalf of each other towards/from the gateway AP. Therefore, covering a cell reduces to adding an AP without wiring it to the backbone network. This way, the WMN technology is considerably easing wireless coverage, especially when compared to the actual Wi-Fi WLANs solution. This is of tremendous importance to industry since easing wireless coverage equates affording more coverage and supporting more user connections, consequently generating greater returns to industry. In

this context, IETF (Internet Engineering Task Force) [14] is actively working to finalize the IEEE 802.11s standard [15] which will pave the way towards a successful worldwide industrialization of this promising technology.

In 2005, IETF set a mesh networking TG (Task Group) to standardize IEEE 802.11s. The work is still in progress, and the last IEEE 802.11s TG meeting was held in May, 2010 [15]. Even though the standard is not final yet, its main traits are set, e.g., architecture and MAC routing. The IEEE 802.11s TG set HWMP (Hybrid Wireless Mesh Protocol) [5] as the default routing protocol for all IEEE 802.11s compliant devices. HWMP is an adaptation of the well-known AODV [25] protocol that is composed of reactive on-demand and proactive tree-based routing. These latter can be used separately or simultaneously depending on the kind of application for which the WMN is deployed. HWMP uses MAC addresses and was amended *Airtime* [6] as a default routing metric. The amendment aims to maintain a minimum compatibility between the IEEE 802.11s devices to be manufactured by different companies.

Albeit a new technology, several WMNs solutions are already commercialized (e.g., by Strix Systems , Cisco, Nortel, Tropos, BelAir). Besides, valuable real-world research deployments are in place as well, e.g., the MIT Roofnet [61] and the Microsoft Mesh Networking project [50]. However, these deployments are not compliant to the new IEEE 802.11s standard and are proprietary.

The absence of non-proprietary and open-source testbeds is restricting the proliferation of the research in WMNs by forcing researchers to use simulation tools, e.g., ns-2 [63] and Qualnet [49]. Even though simulators are relatively good at approximating most of the real-world networking applications, they are still quite far from doing so in wireless indoor environments, which is the case with IEEE 802.11s WMN technology. In wireless indoor environments, an accurate RF propagation simulation is very crucial and it largely affects the wireless link quality characterization, mainly because of the unpredictable and mobile nature of obstacles, e.g., furniture, people, walls, etc. These latter have a direct and strong impact on most RF propagation characteristics, basically reflection, diffraction, path loss, and interference. In this dissertation, we present an open-source

implementation of an indoor IEEE 802.11s WMN testbed. The implementation is transparent, easy-to-deploy, and both the source code and deployment instructions are available online [4]. The implementation can serve as a blueprint for the WMN research community to deploy their own testbeds, negating the shortcomings of using simulation tools.

Wireless link quality characterization imposes the major challenge of which link quality parameter to account for, and this is basically because of the diversity and the complexity of the correlations (e.g., loss rate, interference, transmission/interference ranges, load, contention, delay, and transmission rate). Still, the loss and transmission rates remain the two key link quality parameters, as the former accounts for link reliability, and the latter for link bandwidth. Reliability and high bandwidth are the ultimate goals that are sought in any networking technology.

An accurate estimation of the loss rate is not an easy task. There are two basic approaches for loss rate measurement: the *broadcast* approach and the *passive* approach.

- In the broadcast approach, probe frames are periodically broadcast using a certain frequency, and the number of successfully received probes are used to induce the loss rate. Such an approach suffers fundamental shortcomings: *First*, broadcast frames generate an overhead in the network, which directly impacts the overall network performance. *Second*, in order to minimize the broadcast overhead, large averaging periods (e.g., 10 seconds with 1 probe per second) are used [16]. These large averaging periods do not respond quickly enough to the inherently time-varying nature of the channel quality. *Third* and worse, broadcast frames have a fixed size and are sent at the default *low* constant broadcast rate. This definitely does not represent the real traffic which has variable frames sizes and variable transmission rates. These facts yield inaccurate loss rate measurement under the *broadcast* approach.

- The *passive* approach uses real traffic frames instead of broadcast frames. Thus, it overcomes all the three formerly listed broadcast approach shortcomings: *First*, no overhead is induced in the network since the passive approach uses existing traffic frames. *Second*, existing traffic frames are not sent on a periodic basis as with broadcast frames, thus they can respond to the time-varying nature of channel quality as long as there is ongoing traffic. *Third*, traffic

frames have variable sizes and variable transmission rates. However, the passive approach suffers from the major shortcoming of not dealing with situations where there is no data traffic.

This dissertation proposes an alternative approach that overcomes both the shortcomings of the broadcast and the passive approaches. The approach *indirectly* estimates the loss rate by observing the *causes* of losses instead of the direct measurement of the *effect*, i.e., losses. The causes of wireless losses are basically contention and interferences:

- Contention contributes to frame losses by: 1) inducing excessive buffering of frames, while awaiting for free medium. This results in frames dropping due to buffers overflows, and 2) frame collisions, which further worsens in case of the absence of the RTS/CTS mechanism, which is the default setting on IEEE 802.11 network interface cards.

- Interferences contribute to frame losses by having a desired signal disrupted by other interfering signals, and thus resulting in a corrupted signal which is interpreted as a loss mainly because of the FCS (Frame Check Sequence) alteration.

While contention measurement is quite an easy task, an accurate interference measurement still imposes major challenges. There are two basic *interference models* [3]: 1) the *protocol* interference model, and 2) the *physical* interference model. Both models present major *practical* challenges. In the *protocol* interference model, which is based on transmission and interference ranges, the challenge is relevant to the complexity of measuring such ranges in real-world, especially in ad-hoc networks where there is no predefined topology, and particularly in in-door environments, of which WMN is a case, where obstacles are prevalent and unpredictable. In the *physical* interference model, which is based on $SINR$ (Signal-to-Interference-Noise-Ratio) values, the challenge is relevant to the model requiring access to the $SINR$ values of *all* simultaneously transmitting stations: a feature which is unavailable in commodity NICs which report the signal strength of *only one* successfully received frame at a time. To overcome this physical interference model challenge, we propose a probabilistic interference measurement framework that uses frame arrival models as a

means to infer probabilities of interference. These probabilities of interference are used as weights for signal strengths in the *physical* interference model.

In this context, and by adapting the physical interference model for practical use, we propose a new interference and contention-aware metric ICE (Interference and Contention Estimator) that overcomes the shortcomings of both the broadcast and the passive approach. ICE does not use any extra broadcast frames and overcomes the major shortcoming of probing idle links (which is inherent to the passive approach) as the absence of data traffic becomes a property to detect by itself, and it is interpreted as low contention and low interference. ICE accounts also for the asymmetric nature of wireless links by exchanging channel quality measurements, of both link directions, between the link endpoints. These measurements are piggybacked in the default IEEE 802.11s proactive broadcast routing frames without need for any extra broadcast frames. ICE performance is compared to *ETX* [16] and to IEEE 802.11s *Airtime* metric [6] using the deployed real-world tested. *ETT* [17] is omitted from our comparison as it is very similar in principle to the IEEE 802.11s default Airtime metric.

While experimenting with HWMP and with Airtime [6] as a routing metric, we observed a noticeable ping-pong effect in the network overall throughput. The effect consisted of having the network throughput frequently oscillating, during the life time of experiments, between high and low throughput values. The effect nature is vague and condemned to be a perilous one. In this dissertation, we highlight the nature of this effect, and by using different rate control algorithms (e.g., ARF, AARF, ONOE, AMRR and Constant rate), we prove the strong correlation of the effect to the underlying rate control algorithms. We show that the effect is an inherent behavior, and not necessarily a perilous one as referred to in literature. Furthermore, we allude to the fact that an accurate characterization of the effect can benefit the IEEE 802.11s overall network performance by shaping new mechanisms that account for it. We present a simple mechanism that, by capturing when the system is under the effect, reacts to it and adapts the routing protocol to improve the network performance. The mechanism deems a multi-hop wireless path as a chain of one-hop links whose strength is *equal* to the strength of its weakest link. The mechanism is O(1), decentralized,

and can be easily integrated into the IEEE 802.11s routing protocol.
The next sections, in this chapter, present the motivation behind the different aspects of this dissertation as well as the relevant literature.

1.1 Wireless Mesh Networks

As multi-hop wireless networks, WMNs [1] largely benefit from valuable research led so far on multi-hop networks in general and in particular in MANETs (Mobile Ad hoc Networks) [45–47]. MANETs failed to attract good civil applications, but have tremendously contributed to the proliferating success of WMNs: many WMNs protocols have been borrowed from MANETs. e.g., HWMP [5], Radio-Metric Ad hoc On-demand Distance Vector (RM-AODV) [18], Dynamic Source Routing (DSR) [19], Optimized Link State Routing (OLSR) [20]; The migration of such protocols profit from the exemption from the stringent constraints of mobility and power consumption. Such constraints were inherent to MANETs. However, such migrations should noticeably take into account major existing differences between MANETs and WMNs [22].

In general, multi-hop wireless networks have been extensively addressed in literature, especially in Routing [16, 17, 20, 23–25, 32]. Still, the capacity of such networks is deemed limited [3]. In an attempt to improve the capacity, different techniques have been proposed: load-balancing [28–30, 30], multi-channel [33–36], multi-radio [17, 28, 57], packet scheduling [41], directional antennas [42, 43], network coding [37–40], multi-hop routing [5, 19, 20, 23, 25]. However, most of these techniques depend at least on one common issue, which is link quality characterization:

1. *Load balancing*: We need to know the *least* loaded links, and *highest* loaded ones as well, in order to balance the load between the different possible routes.

2. *Multi-channel/multi-radio*: we need to know the least interfering channels/radios in order to enable maximum parallel transmissions and in order to perform optimal channel-to-interface as well as neighbor-to-channel bindings.

3. *Packet scheduling:* we need to know the optimal (i.e., minimum losses, less interferences, high transmission rates, etc) paths for packet scheduling.

4. *Directional Antennas*: we need to know the best vicinities (e.g., of minimum contention, load, and loss) in order to know the direction for steering the antennas. In case of non-steerable antennas, we need to know the best direction for frame forwarding and use the corresponding antenna.

5. *Multi-hop routing:* we need to know the *optimal* paths for frame forwarding.

Accordingly, a good link quality characterization of wireless links is crucial to multi-hop wireless networks performance optimization, and hence is critical to WMNs as well. Consequently, the fundamental question that arises is how to characterize a good link quality?

1.2 Wireless Link Quality Characterization

1.2.1 General Overview

When exposed to wireless link quality characterization, one is first confronted with the large set of parameters that impacts the overall wireless link quality, e.g., interferences levels, loss rate, delay (and its variation, i.e., jitter), bandwidth, transmission power, transmission ranges, interference ranges, contention, load. Consequently, one ought to raise the following legitimate question: Which parameters to consider as (*non*)essential for link quality characterization? As a first step towards the answer, we start by investigating how existing work, in literature, is addressing the issue of link quality characterization.

The common means, in literature, for characterizing wireless links are routing metrics. These latter differ in how they address the former question: While some metrics account solely for loss rates and/or bandwidth, others account solely for delay instead. Other metrics account for interferences, or even a combination of the formerly listed parameters. Given these different approaches, it becomes clearer that deciding on the critical or determinant link quality parameters is not a straightforward task. Another important point is that some routing metrics do *explicitly* account for some

parameters while *implicitly* accounting for others. For instance, RTT (Round Trip Time) [26] explicitly accounts only for delay. However, it implicitly accounts for contention and loss rate as well(*The way it does so will be discussed later in this Section*).

To further highlight the issue, we present next some of the most well-known metrics along with the parameters they account for. To emphasize how (*i.e., explicitly or implicitly*) a routing metric accounts for link quality parameters, we use the following notation:

$$metric\_name(E = \{p_1, p_2...\},\ I = \{p_n, p_{n+1}...\})$$

where p_i denotes the different link quality parameters, E is the set of *e*xplicitly accounted for parameters and I is the set of *i*mplicitly accounted for parameters.

1. ETX *(E=*$\{$*Loss rate*$\}$*)*:

   ETX (*Expected Transmission Count*) [16] measures the expected number of transmissions needed to successfully transmit a frame. To compute ETX, every node periodically broadcasts N probe frames over a certain time period T. Knowing N and T, every node counts the number of successfully received probe frames and computes the (reverse) delivery ratio in the reverse link as follows:

   $$d_r = \frac{R_r}{N} \tag{1.1}$$

   where R_r is the number of successfully received frames in the reverse direction.

   Besides, whenever a node broadcasts its N probe frames (always over the same time period T), it piggybacks on it the computed d_r values for all its neighbors. This way, every node can get its forward delivery ratios d_f, to other neighbors, by extracting them from the received probe frames. Once having both the forward and reverse delivery ratios to every neighbor, ETX is computed as follows:

   $$ETX = \frac{1}{d_f \times d_r} \tag{1.2}$$

ETX accounts, explicitly and solely, for loss rate as a link quality parameter.

2. ETT *(E=$\{$loss rate, bandwidth$\}$)*:

   ETT (*Expected Transmission Time*) [17] measures the time required for a frame to be successfully transmitted. ETT is computed as follows:

$$ETT = ETX \times t \tag{1.3}$$

   where t is the average time for a single frame to be transmitted regardless of the transmission being successful or not. $t = \frac{S}{B}$, where S is the size of the probe frame and B is the link bandwidth.

   ETT accounts, explicitly, for both loss rate and bandwidth as well.

3. IEEE 802.11s *Airtime (E=$\{$loss rate, bandwidth, channel characteristics$\}$)*:

   IEEE 802.11s Airtime is a radio-aware metric which is meant to measure the amount of consumed channel resources when transmitting a frame over a particular wireless link [6]. Airtime is computed as follows:

$$Airtime = \left(O_{ca} + O_p + \frac{B_t}{r}\right) \frac{1}{1 - e_{fr}} \tag{1.4}$$

   where O_{ca} and O_p are constants quantifying respectively the channel access overhead, and the protocol overhead. O_{ca} and O_p depend solely on the IEEE 802.11 modulation scheme, i.e., IEEE 802.11a/b/g, see Table 5.1. B_t is the number of bits in a probe frame and r is its transmission rate (in Mbps). e_{fr} is the frame error rate. IEEE 802.11s did not set a specific way to measure e_{fr}, it is left as a local implementation choice [21].

   Like ETT, Airtime accounts, explicitly, for loss rate and bandwidth. However, Airtime further accounts for the channel characteristics as well by means of O_{ca} and O_p.

4. RTT *(E=$\{$delay$\}$, I=$\{$contention, loss rate, queuing delay$\}$)*:

   The RTT (*Round Trip Time*) [26] of a hop is computed by sending a unicast probe frame

	802.11a	**802.11b/g**
O_{ca}	75 μs	335 μs
O_p	110 μs	364 μs

Table 1.1: Airtime metric constants

bearing a timestamp. Upon receiving the probe, the receiver responds with a unicast frame echoing the received probe which still bears the initial timestamp. This way the receiver can compute the elapsed time (RTT) by subtracting the initial timestamp from the current timestamp.

Even if RTT accounts solely for delay as a link quality parameter, it does *implicitly* account also for contention, queuing delay and loss rate:

- RTT *implicitly* accounts for contention since high contention levels would increase the waiting time (hence increasing RTT) needed to grasp the medium and vice-versa.
- RTT *implicitly* accounts for the queuing delay as this later is included in the RTT. RTT corresponds to the elapsed time since a frame leaves a station till it is received back.
- RTT *implicitly* accounts also for loss rate since a loss would induce re-transmitting the frame. Such re-transmissions increase the delay.

RTT accounts explicitly for delay, and implicitly for contention, queuing delay and loss rate.

5. PktPair *(E={delay}, I={contention, loss rate})*:

   PktPair (*Packet Pair*) [51] estimates the delay of a link by broadcasting two back-to-back frames: a short one (137 Bytes) followed by a relatively long one (1100 bytes). Only the first short frame is timestamped. The receiver computes the difference in time between receiving the two frames and echoes back the computed delay. In contrast to RTT, PktPair does not account for queuing delays since the two frames are sent back-to-back (back-to-back frames are to witness the same queuing delay). However, PktPair implicitly accounts for contention

and loss rate as in RTT.

PktPair accounts explicitly for delay, and implicitly for contention and loss rate.

6. WCETT *(E={loss rate, bandwidth}, I={interferences})*:

   The WCETT (*Weighted Cumulative Expected Transmission Time*) [17] for a given path p is computed as follows:

$$WCETT_p = (1-\alpha) \times \sum_{i=1}^{n} ETT_i + \alpha \times \max_{1 \leq j \leq k} X_j \tag{1.5}$$

   where X_j is the sum of the ETT values of all the single hops, in a path p, that are operating in a same specific channel j, and α is a weighting parameter. X_j is meant to account for channel diversity by accounting for the channel that may cause highest intra-flow interferences as a result of more hops operating in the same channel. WCETT was designed for multi-channel multi-radio networks.

   WCETT accounts for loss rate, bandwidth and interferences as well. However, WCETT does not explicitly measure interferences, it does only approximate interferences by accounting for the hops, in the same path, that are using the same channel, i.e., the number of links that can interfere with each other. In other words, WCETT is using a *heuristic* approach for interference measurement instead of an interference measurement *model.*

7. MIC *(E={loss rate, bandwidth}, I={interferences})*:

   The MIC (*Metric of Interference and Channel switching*) [52] for a given path p is computed as follows:

$$MIC_p = \frac{1}{N \times \min(ETT)} \sum_{linkl \in p} IRU_l + \sum_{nodei \in p} CSC_i \tag{1.6}$$

   where N is the total number of nodes in the network, IRU_l and CSC_i are computed as follows:

$$IRU_i = ETT_i \times N_l \tag{1.7}$$

$$CSC_i = w_1, \quad if \;\; CH(prev(i)) \neq CH(i); \quad CSC_i = w_2, \quad if \;\; CH(prev(i)) = CH(i) \tag{1.8}$$

$$0 \leq w_1 \ll w_2 \tag{1.9}$$

where N_l is the number of neighbors interfering with link l. $CH(i)$ is the channel assigned to node i and $prev(i)$ is its previous node along path p.

MIC is meant to account for both intra-flow and inter-flow interferences: CSC_i catches intra-flow interferences by assigning to it a high value, e.g., w_2, when the previous node is operating in the same channel, and a less value (w_1) otherwise. MIC accounts for inter-flow interferences by scaling ETT (in equation 1.7) by N_l (the number of interfering links).

MIC accounts for loss rate, bandwidth and interferences as well. However, like in WCETT, MIC does not explicitly measure interferences. It only estimates it by accounting for the number of interfering links and the number of path hops that are operating in the same channel. Similarly to WCETT, MIC uses a *heuristic* approach for interference measurement instead of an interference measurement *model*.

8. iAware *(E={loss rate, bandwidth, interferences})*:

   iAware (*Interference Aware*) [53] uses the same rationale and the same path metric formula as WCETT (see Equation 1.5), except that it replaces ETT_i with $iAware_i$:

$$iAware_p = (1 - \alpha) \times \sum_{i=1}^{n} iAware_i + \alpha \times \max_{1 \leq j \leq k} X_j \tag{1.10}$$

$$iAware_i = \frac{ETT_i}{IR_i} \tag{1.11}$$

   where IR_i is a SINR-dependent value which is computed under the physical interference model [3] as follows:

$$IR_i(u) = \frac{SINR_i(u)}{SNR_i(u)} \tag{1.12}$$

$$SINR_i(u) = \frac{p_u(v)}{N + \sum_{w \in I_w} \tau(w) P_u(v)} \tag{1.13}$$

where $p_u(v)$ is the signal strength of a packet from node v to node u, I_w is the set of nodes in the interference range of node u. $\tau(w)$ is the normalized rate, at which a node w generates traffic, averaged over a certain period of time: $\tau(w)$ represents the fraction of time node w occupies the channel.

iAware accounts for loss rate and bandwidth through the incorporation of ETT and also for interferences by using an SINR-based model. In contrast to WCETT and MIC, iAware is explicitly measuring the interferences by using a SINR-based model. However, iAware, simplifies the physical interference model by using normalized rates for weighting the signal strengths values (See Equation 1.13).

1.2.2 Which Link Quality Parameters to Account for?

By examining these different routing metrics, we can state that deciding on the set of parameters to account for, in characterizing the wireless link quality, is a multi-way issue as it can be approached differently. The challenge stems mainly from the diversity and the complexity of the correlations between these different link quality parameters. However, in the shade of the formerly presented routing metrics, we observe that all of them account primarily, either implicitly or explicitly, for loss and transmission rate. This fact exhibits a solid rationale since loss and transmission rates are highly impacting the overall network throughput: a minimum loss rate yields the highest reliability. When this latter is combined with a high link bandwidth, the system ideally reaches the maximum throughput.

In contrast to transmission rate, an accurate loss rate measurement is quite difficult to achieve, for the reasons highlighted in Section 1. As an alternative, we proposed observing the causing events instead of the effects, i.e., contention and interferences. The next section reviews interferences and their impact on wireless network performance.

1.3 Interferences

In physics, interference is the addition (superposition) of two or more waves that result in a new wave pattern. In computer science, this superposition of signals/waves causes bit alterations which in turn causes data and/or FCS (Frame Check Sequence) alteration. When the frame's data and/or its FCS are corrupted, the link layer drops the corrupted frame and hence generates a loss.

1.3.1 Accurate Interference Measurement: The Challenging Aspects

There is a general consensus in literature about the impact of interferences in wireless networks [3, 66]. Interferences degrade wireless networks performance by their direct/strong contribution to the generation of wireless losses. Besides, interferences also increase the contention level when an interfering signal is above the carrier-sense threshold, causing frame emission to be delayed till the channel is clear. The interfering signals can also originate from undesired communications, e.g., microwaves, bluetooth devices, traffic in a different network. An accurate interference measurement is a challenging task for three main reasons:

1. *High time complexity*: There are $O(n^2)$ links in a network with n nodes, and if only pairwise interferences are considered, then $O(n^4)$ links must be checked. In [67], J. Padhye et al. reduce this pairwise complexity from $O(n^4)$ to $O(n^2)$. However, this still considers only pairwise interferences, which does not reflect the real world. A signal can simultaneously interfere with many signals, not only with one.

2. *Inadequacy of Interference models for use in practical systems:* The *protocol* model [3] which characterizes interferences by considering both transmission and interference ranges, exhibits a strong inadequacy for use in practical systems. This is merely because the transmission and interference ranges cannot be computed at all in ad-hoc topologies. Even, in a controlled-topology network, the computation is $O(n^2)$, and exhibits a high level of imprecision because of the complex RF propagation aspects. Furthermore, transmission/interference

ranges vary in time because of mobile obstacles and variable transmission powers. On the other hand, the *physical* model [3], even if it is simpler than the protocol model, it requires the access to the $SINR$ values of all interfering stations: a feature which is unavailable in commodity NICs as these latter report the signal strength of *only* one signal at a time. We refer back to this model with further details in Section 3.

3. *Inadequacy of simulation tools:* Simulation tools are indeed inadequate in dealing with interferences as they can not capture the complex aspects of RF propagation such as multi-path fading, interferences, path loss, reflection and diffraction. In indoor environments, the situation is further aggravated because of the unpredictable and mobile nature of obstacles, e.g., furniture, people, walls, etc. This unpredictable and mobile nature of obstacles has a direct and strong impact on the behavior of most RF propagation characteristics, basically reflection, diffraction, and path loss. These facts render very complex an accurate modeling of the interferences phenomenon. Thus, most simulation tools remain simplistic in the way they are modeling interferences. For instance, ns-2 [63], which is the most widely used tool in academia, uses a simplified version of the physical interference model (the capture threshold model) that accounts for only one interfere at a time [8]. This is definitely not the case in real-world interferences where there can be several simultaneous interferes.

1.3.2 Common Heuristics for Interference Measurement in Routing Metrics

To cope with the formerly presented interferece measurement challenges, most routing metrics use simple heuristics. In this section, we highlight the most known interference-aware routing metrics and the way they measure interference:

1. $WCETT$ (Weighted Cumulative Expected Transmission Time) [17] uses the maximum sum of the ETT values of all single hops, on a given path, that are operating in the same channel (See Equation 1.5). By doing so, $WCETT$ is favoring paths with more channel diversity, i.e., with less interfering channels. This is a good heuristic but still no interferences are

measured/computed: $WCETT$ computes only the maximum number of hops operating in the same channel, and uses it as a weight in Equation 1.5.

2. MIC (Metric of Interferences and Channel switching) [52] tries to improve over $WCETT$ by additionally accounting for inter-flow interferences as $WCETT$ accounts solely for intra-flow interferences, i.e., interferences occurring within the same path. However, to do so, MIC measures the number of neighbors that can interfere with a given link l and uses that number to weight the ETT of the link (Equation 1.7). Like $WCETT$, MIC uses a good heuristic as well since it favors paths with less inter and intra-flow interferences. However, it does not explicitly measure any interferences: it only computes the number of interfering nodes within a path and uses that number to compute the corresponding IRU (value (Equation 1.7).

3. Unlike $WCETT$ and MIC, $iAware$ [53] is explicitly measuring interferences by computing the signal strengths of neighboring links. However, $iAware$ is *too simplistic* when it used the normalized rates as weights for the signal strengths under the physical interference model (Equation 1.13). Normalized rates, even if they are in the range [0, 1], they are quite far from representing a probability of interference simply because they totally don't account for the counterpart node with whom the interference may take place. Regardless of the sending rate of the counterpart node, the locally-computed normalized rate will be always the same, and this is a big flaw: When the counterpart node will be sending at higher rate, and thus injecting more frames in the air, the probability to have an interference should increase and vice-versa.

1.4 IEEE 802.11s Airtime Ping-pong Effect

While experimenting with the IEEE 802.11s HWMP protocol, and using Airtime as a routing metric, in the deployed IEEE 802.11s WMN testbed, we noticed a ping-pong effect in the

behavior of the network. This latter consisted of a continuous oscillation in the network throughput between high and low values. At first, we thought the effect was an anomaly that stems from an undetected flaw, e.g., in coding or in experimental settings. However, after re-examining the code and the experimental settings, it turned out that the effect is still persisting. As a next step, we took over the same experiments but with ETX as a routing metric instead. This resulted in a noticeable mitigation in the effect strength. As a primary observation, we postulated that the effect ought to be relevant only to the IEEE 802.11s Airtime metric.

Afterwards, and as a first step ahead, we conducted a literature review whereby we, surprisingly, found only a unique reference to the effect [9]. This latter reference superficially addresses the effect and condemns it to be a perilous behavior and without any solid clarification. Furthermore, the authors even changed the IEEE 802.11s computational formula of Airtime [6] to render it independent of the transmission rate variable, and this in order to eliminate the "so-named" perilous behavior.

Given that the ping-pong effect ought to impact the overall performance of this new IEEE 802.11s technology as it pertains to an integral part to the IEEE 802.11s HWMP routing protocol which is the Airtime routing metric, and given the very scarce literature on the effect, we think that this effect should be allotted further research, and that a good characterization of it can even help improve the overall network performance by merely accounting for the effect presence, and this is what we are trying to also present in this dissertation.

1.5 Contributions

The main contributions of this dissertation are:

- A presentation of the subtleties of wireless link quality characterization.
- A presentation of a novel approach for accurate loss measurement that observes the causing events instead of the effects.

- A presentation of a novel framework for interference measurement under the physical interference model.
- A presentation of a new interference and contention aware routing metric.
- Highlight and characterization of the Airtime ping-pong effect.
- A proof of the strong correlation of the effect to adaptive rate control algorithms.
- A proof that the effect is an inherent behavior and not necessarily a perilous one.
- A presentation of a decentralized ping-pong-aware mechanism that improves the overall network performance.
- A presentation of an easy-to-deploy and open-source IEEE 802.11s WMN testbed implementation.
- A solution to some of the main IEEE 802.11s implementation problems, e.g.,:
  - Clients association problem
  - Internetworking WMNs
  - Supporting multiple gateways
- A highlight of the main traits of the new IEEE 802.11s standard.
- A highlight and implementation of the IEEE 802.11s HWMP routing protocol and the IEEE 802.11s Airtime routing metric.

The rest of this dissertation is organized as follows:

In Chapter 2, we present the main traits of the ongoing IEEE 802.11s, with an emphasis on the subtleties of the IEEE 802.11s HWMP protocol and the IEEE 802.11s Airtime routing metric. In Chapter 3 we present the issue of wireless link quality characterization, we highlight the shortcomings of the current schemes, and propose a new interference-aware routing metric that outperforms

ETX and IEEE 802.11s Airtime. The Airtime ping-pong effect is thoroughly covered in Chapter 4, and its correlation to the underlying rate adaptation algorithms is analyzed and experimentally proved. In Chapter 5, we present the details of a real-world open-source implementation of an indoor IEEE 802.11s WMN testbed. We highlight major implementation problems and suggest suitable solutions. We show how the implementation is indeed easy-to-deploy, and thus encouraging the research community to use it as a blueprint to deploy their own testbeds. Finally, in Chapter 6, we conclude and allude for future research steps.

Chapter 2

The IEEE 802.11s Standard

In this chapter we highlight the main traits of the ongoing IEEE 802.11s standard [2]. We outline its time-line progress, present its architecture, and delve into the details of it routing protocol.

2.1 Progress and Standard Main Areas

The IEEE 802.11s standard started initially as a study group in 2003, then became a Task Group in July 2004. The first draft was accepted in March, 2006, and the last TG meeting was held in March 2010 [15]. The standard is concerned with five main areas:

1. Architecture
2. Routing in MAC layer
3. MAC enhancements
4. Internetworking
5. Security

In this dissertation, we refer to areas 1, 2 and 4, as they are relevant to this work. Areas 1 and 2 are introduced in next paragraphs, while area 4 is deferred to Section 5.3.

2.2 Architecture

IEEE 802.11s defines three types of stations:

1. MPs (Mesh Points): MPs are wireless stations that perform routing *only*.

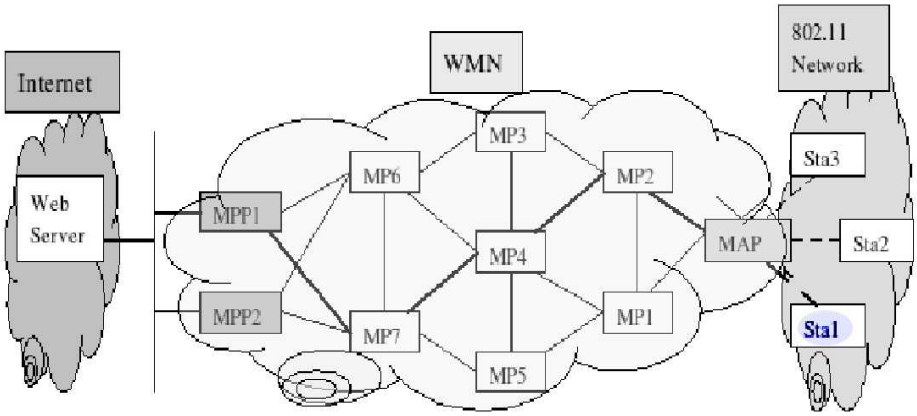

Figure 2.1: WMN Architecture

2. MAPs (Mesh Access Points): MAPs are MPs with additional access point capabilities. Besides performing routing, MAPs aggregate traffic from/towards legacy 802.11 stations. A MAP can be thought of as a legacy access point which performs routing *also*.

3. MPPs (Mesh Portal Points): MPPs are MPs that serve as gateways to other non-mesh networks, e.g., Internet. MPPs aggregate traffic from/towards the non-mesh networks.

To illustrate the functionality of every mesh node type, Figure 2.1 depicts a scenario where an end user, e.g., a Wi-Fi enabled station, is browsing the Internet. The end-user, at station *sta1*, is connected to a legacy IEEE 802.11 network through a mesh access point (*MAP*). The MAP is routing frames, and has mesh point (MP2) as a next-hop. Mesh point (MP2) has mesh point (MP4) as its next-hop, and (MP7) afterwards. This latter forwards data towards the Mesh Portal Point (MPP1) which serves as a gateway towards the Internet web server. The role of the HWMP routing protocol [5] is to determine the best sequence of hops for a data frame to get to its final destination. This is carried out by selecting the path that has the best (least) Airtime [6] value.

2.3 MAC Routing

Multi-hop routing is a key issue in IEEE 802.11s. IEEE 802.11s performs routing using MAC addresses and uses either four *or* six MAC addresses depending on the nature of the traffic.

When both the source and the destination are mesh points in the WMN, only four addresses are used. When the source or the destination is outside the WMN, i.e., a non-mesh point (e.g., an IEEE 802.11 or an IEEE 802.3 station), six addresses are used. The IEEE 802.11s frame header bear an *AE* (Address Extension) mode bit that discriminates between the two modes. In the cases where WMNs are used for last-mile Internet access, the *AE* bit is set to 1 as both the destination and the source are non-mesh points. The scenario, formerly illustrated in Figure 2.1, is a relevant case as the source is a wireless station residing in an IEEE 802.11 network, and the destination is a web server residing in the Internet.

Adding two extra MAC addresses, to the ordinary four IEEE 802.11 addresses, is meant to track the MAC addresses of the non-mesh source and non-mesh destination as these would be lost once the frame enters/quits the mesh network. The IEEE 802.11s six MAC addresses are:

1. Destination MAC Address: MAC address of the next-hop in a given route.
2. Source MAC Address: MAC address of the sender.
3. Destination Proxy Address: MAC address of the destination MPP/MAP from where the frame will leave the mesh network.
4. Source Proxy Address: MAC address of the MPP/MAP from where the frame entered into the mesh network.
5. Final Destination Address: MAC address of the final non-mesh destination.
6. Originator Address: MAC address of the non-mesh station that originated the frame.

HWMP [5] is the default routing protocol in IEEE 802.11s. HWMP is a hybrid protocol being both reactive and proactive. The two behaviors can be used separately or simultaneously depending on the application.

2.3.1 Reactive Routing in HWMP

RM-AODV (Radio-Metric Ad hoc On Demand Distance Vector) [18] is the reactive protocol in HWMP. RM-AODV is an adaptation of the AODV [25] protocol that uses Airtime [6] as a link quality metric. Reactive routing protocols initiate route discovery requests only when needed, such as in case of a route failure or a route time-expiration.

In RM-AODV, when a node needs a path to a certain destination, it broadcasts a PREQ (Path Request) message that contains the MAC address of the destination. Every PREQ message bears a unique sequence number that distinguishes it from other PREQ messages. When an intermediate MP receives a PREQ message, it first checks the *freshness* of the received message by comparing its sequence number with the last locally stored sequence number. A PREQ message is processed only when it has a greater sequence number or when it has the same sequence number but exhibiting a better route.

After receiving a fresher PREQ message, intermediate nodes update their *reverse* path towards the source, update the routing metric by including the weight of the last hop, and then re-broadcast the updated PREQ message. Subsequent MPs proceed the same way until the PREQ reaches the destination.

When the destination MP receives the PREQ message, it checks its freshness in the same way as other intermediate MPs, then it updates its *reverse* path towards the source. The destination then formulates a RREP (Route Reply) message which is afterwards uni-casted towards the source. Intermediate MPs receiving the RREP message update their *forward* path towards the destination, update the routing metric, and then forward the updated RREP towards the source.

In case of node failure, the intermediate MPs detecting the failure send a RERR (Route Error) message towards the source to inform it about the breakage of the link. The source can then re-initiate a new route discovery using a new PREQ message.

2.3.2 Proactive Routing in HWMP

The HWMP proactive mode [5] is a tree based routing [10, 33].

In the proactive mode, every root mesh point (i.e., MPP) periodically broadcasts PREQ messages bearing unique sequence numbers. The protocol is very similar to reactive HWMP, presented in Section 2.3.1, in how intermediate nodes react and process PREQ messages. The only differences are:

- In reactive HWMP, PREQ messages are sent in a reactive way (i.e., when needed), whereas in proactive HWMP, PREQ messages are sent proactively (i.e., in advance) and on a periodic basis (e.g., every 1 second).
- In reactive HWMP, PREQ can be sent by any mesh node type (MP, MAP and MPP), whereas in proactive HWMP, PREQ messages are sent *only* by MPPs. These latter represent the trees roots. MPs and MAPs do only forward PREQ messages.

By having every MPP periodically broadcasting PREQ messages, tree-like topologies are built with MPPs as roots and MAPs as leaves. By comparing the routing metrics of the different PREQ messages, MAPs select the best MPP and thus adhere to the tree whose root is the selected MPP.

The proactive protocol is ideal for the scenario where WMNs are used for last-mile Internet access since most of the traffic is directed towards/from the gateway MPPs that connect the WMN to the Internet.

2.4 IEEE 802.11s Radio-Aware Airtime Metric

IEEE 802.11s Airtime [6] is a radio-aware metric which is meant to measure the amount of consumed channel resources when transmitting a frame over a particular wireless link. Airtime is computed as follows:

$$Airtime = \left(O_{ca} + O_p + \frac{B_t}{r}\right) \frac{1}{1 - e_{fr}} \tag{2.1}$$

where O_{ca}, O_p and B_t are constants quantifying, respectively, the Channel Access Overhead, the Protocol Overhead and the number of Bits in a probe frame. r is the transmission rate (in Mbps) for a frame of size B_t, and e_{fr} is the frame error rate. IEEE 802.11s did not delineate a specific way to measure e_{fr} , it is left as an implementation issue [21].

Unlike ETX (Expected Transmission Count) [16] which accounts solely for frame error rate, Airtime accounts for both frame error rate and link bandwidth as well, which is also the case with ETT (Expected Transmission Time) [17]. However, Airtime further accounts for channel access and protocol overheads.

IEEE 802.11s characterizes the link quality by means of observing the loss and transmission rates. However, the standard did not delineate a specific way for such an observation, mainly how to measure the loss rate, and this is, in fact, the central issue in link quality characterization. Next chapter presents the subtleties of link quality characterization and suggests a novel scheme for loss rate estimation.

Chapter 3

Wireless Link Quality Characterization

In this chapter, we present the different aspects of wireless link quality characterization and show how transmission and loss rates are the most important link quality metrics to account for. We highlight and present the shortcomings of both the broadcast and passive approaches of loss measurement, and suggest a novel scheme that relies on interference measurement.

We introduce the current models for interference measurement, highlight their shortcomings, and suggest a novel framework for interference measurement that is based on the physical interference measurement. A new interference-aware metric is presented and compared against ETX and IEEE 802.11s Airtime metrics using a real-world testbed. Experiments are detailed and results are presented and discussed.

3.1 How to Accurately Characterize Wireless Link Quality?

Given the large set of parameters that affect the overall quality of a wireless link, e.g., loss rate, transmission rate, interferences, contention, delay, load, contention, transmission/interference ranges, we must first decide on the set of parameters to consider. In fact, there is a trade-off between keeping the *parameter set* as small as possible (in order to minimize measurements overhead) and having a representative *parameter set* capable of fairly characterizing wireless link quality.

In the formerly reviewed link quality characterization schemes (see Section 1.2.1), different *parameter sets* are used: while ETX [16] accounts solely for loss rate, ETT [17], WCETT [17], MIC [52], and iAware [53] do all account for loss rates as well as transmission rate. Furthermore, WCETT, MIC and iAware account additionally for interferences. On the other hand, RTT [26] and PktPair [51] account only for delay. Other metrics account for the load as well [27–29]. Hence, we see that deciding on the *parameter set* is a *fuzzy* process merely because of the "*imprecision*",

diversity, and the complexity of the correlations between these different link quality parameters: For instance, we see that incorporating both interference and loss rate into link quality characterization involves some redundancy since interferences are highly correlated to losses. Measuring both interferences and losses means measuring the losses (*that are caused by interferences*) twice!

In an attempt to de-fuzzify the set, we do last incorporate a deterministic ingredient drawn from a network end-user perspective, which always cares about two main qualities:

1. Receiving the *whole* requested data; this is characterized by loss rate.

2. Receiving the requested data as *quickly* as possible (high bandwidth) and without distortion (good QoS, e.g., jitter in case of multimedia applications); This is characterized by the link bandwidth, i.e., the transmission rate.

Thus, we state that loss and transmission rates are the parameters to be included in every optimal link quality characterization approach. Still, other parameters can be included as well; However, we think that the previously mentioned trade-off between the parameter set granularity and the measurement overhead is highly worth consideration. However, an accurate measurement of these parameters remains the key issue.

3.2 Need for Accurate Loss Rate Measurement

The common way for loss rate measurement in most routing metrics (e.g., ETX [16], ETT [17], WCETT [17], MIC [52], and iAware [53]) is the *broadcast* approach whereby periodical broadcast probe frames are used to "*directly*" measure the losses. As formerly presented in Section 1, the broadcast approach suffers three fundamental shortcomings that makes loss rate measurement inaccurate:

1. Due to their broadcast nature, the probe frames add significant load to the network, and thus they are using more network resources and are increasing the contention levels. A fact which impacts the overall network performance especially in wireless networks where the air medium resource is critical and shared by all communications.

2. In attempt to overcome these latter shortcomings, large averaging periods are used (e.g., 10 seconds. with 1 probe frame per second [16]). These large averaging periods do definitely not respond to the time-varying nature of the channel link: In a wireless link, the channel quality can change in very short time periods (hundreds of milliseconds).

3. The characteristics of the broadcast probe frames, basically the size and the transmission rate, are constants. Broadcast frames are of fixed size and are sent at the *"low"* constant rate, hence they cannot represent the dynamics of real traffic frames. These latter are sent using different frame sizes as well as different transmission rates (e.g., when using adaptive rate control schemes).

Taking into account these fundamental shortcomings, the broadcast approach would definitely yield inaccurate loss rate measurements.

In attempt to overcome the broadcast approach shortcomings, the *passive* approach [11], on the other hand, proposed the use of real traffic frames instead of broadcast frames, a fact that would definitely overcome all the shortcomings of the broadcast approach as no broadcast frames are used. However, the passive approach suffers, in its turn, from the major shortcoming of probing idle links: The passive approach should continuously track if a link is idle or not, and when a link turns out to be idle, the passive approach should switch back to the broadcast approach, and then back again to the passive approach when the link becomes busy again. Such switchings back and forth, as well as the continuous tracking of the idle-status of links, induces additional overhead.

To overcome the shortcomings of both approaches, we propose the *indirect* approach of estimating the loss rate by measuring the *events causing* losses instead of directly measuring the losses by themselves, as with the broadcast and the passive approaches. The events causing wireless losses are basically contention and interferences.

Unlike the passive approach, our approach does not suffer from the fundamental shortcoming of probing idle links: The approach interprets the absence of traffic as a valuable information to detect by itself since this would indicate minimum interferences and minimum contention, and hence favoring such a link to be selected by the routing protocol.

Adopting such an alternative would induce more accuracy in link quality characterization and with minimum overhead, as no broadcast probes are used at all. Only real traffic frames are used instead for loss measurement. However, an accurate interference measurement is also required, and as it was highlighted in the literature (Section 1.3), interferences measurements imposes inherent challenges as well. The next section suggests a suitable solution.

3.3 Interferences Measurement

3.3.1 Interferences Measurement under the Physical Interference Model

3.3.1.1 The Challenges

Before we state the problem of interference measurement, we should first select an underlying interference model. In fact, the *protocol* and the *physical* interference models are the most studied ones in literature [3, 8, 66, 67]. Other interference models are basically just variations of these two former models, e.g., the interference capture and the interference range models [8].

The protocol model is complex, as it depends on transmission and interference ranges (which are hard to compute on practical systems, specially in ad-hoc networks where the topology is unpredictable). The physical model, on the other, hand is simpler as it relies solely on the interfering signals strengths. Furthermore, the physical model is the closest one to reality as it does *"physically"* model the interferences by accounting for the "physical" quality of "wave superposition" where interfering signals add up (i.e., superpose) to produce the resulting signal. In [8], A. Iyer *et al.* concluded, through a solid comparative study about the impact of different interference models on wireless network performance, that an SINR-based model is the minimum level of detail that should be employed to model wireless interferences. A. Iyer *et al.* strongly recommend the use of the additive physical interference model because of its realistic approach.

The physical interference model [3] states that a communication from a node v to a node w is

successful only and only if its SINR is above a certain threshold β:

$$SINR_{|v,w)} = \frac{SS_{|v,w)}}{N_w + \sum_{x \in I_w} SS_{|x,y)}} > \beta \tag{3.1}$$

where $SS_{|v,w)}$ and $SS_{|x,y)}$ denote the signal strengths of the uni-directional links $|v,w)$ and $|x,y)$. I_w is the set of nodes in the interference range of w, and N_w is the thermal noise level at node w.

Even though the physical model is the closest one to reality and exhibits a good level of simplicity, it still imposes a fundamental experimental problem. In practice, applying the physical model inequality (3.1) is quite unfeasible for the following two main reasons:

1. Commodity wireless card drivers can have access to the signal strength of ***only one*** frame at a time, whereas the physical interference model inequality (3.1) requires the simultaneous access to the signals strengths of ***all*** interfering frames.

2. Commodity wireless card drivers can track the signal strengths of only successfully received frames, hence missing interference events.

In practice, we can estimate the average signal strengths of all interfering nodes but we cannot know which nodes will be simultaneously transmitting, i.e., which nodes will be interfering. In such a non-deterministic situation, using probabilities is the natural and mathematical response. Accordingly, we propose weighting the signal strengths, in the physical interference model inequality (3.1), by probabilities of interference. The next section highlights the proposed solution.

3.3.1.2 Our Approach

When using probabilities of interference as weights for signal strengths, the physical model inequality (3.1) becomes:

$$SINR_{|v,w)} = \frac{SS_{|v,w)}}{N_w + \sum_{x \in I_w} P_w^v|x,y) SS_{|x,y)}} > \beta \tag{3.2}$$

where $P_w^v|x,y)$ is the probability that a signal originating from node x towards node y will interfere, at node w, with the signal originating from node v towards node w.

This way, we suggest a probabilistic approach, for interference measurement under the physical model, where the signal strengths are weighted by the probabilities of interference. These probabilities should logically depend on the arrival rates of the concerned links: A pair of links witnessing high load should have a higher probability of interference that another pair of links whose load is relatively lower. Next section highlights the issue and presents the solution.

3.3.2 IEEE 802.11 Traffic Modeling and Interferences Probabilities Computation

To compute the interferences probabilities ($P_w^v|x,y)$ in Equation 3.2), we need to model the IEEE 802.11 DCF traffic. In other words, we need to model the frame arrival process. Assuming we have such a model, then the probability for two frames to interfere can be easily computed. However, as far as we know, no frame arrival model has been proposed yet for IEEE 802.11 DCF traffic: In IEEE 802.3 LANs, the traffic model was studied and it appears to be self-similar [12]. The question heretofore is: Which model should be used for IEEE 802.11 DCF traffic?

Since no traffic model for IEEE 802.11 DCF has been proposed (so far), then there is no deterministic answer to the former question. Consequently, and in an attempt to have this work as the first one to propose a framework for interferences measurement under the physical model using traffic modeling, we simplify the formerly stated problem by assuming that IEEE 802.11 DCF traffic follows a stochastic/random process, and we model the frames arrivals using a Poisson process [13]. This way, we also raise a fundamental case for future research, in such a direction, that can provide a well-founded solution to accurate interference measurement under the well-known physical interference model. Last and not least, we further support our assumption by the following single argument:

- IEEE 802.11 DCF traffic still exhibits at least a non-negligible level of stochasticity/randomness through the use of the IEEE 802.11 backoff mechanism.

Under the former assumption, if the expected number of frames arrivals during a given time period T is λ, then the probability that there will be exactly k arrivals (k being a non-negative integer, $k = 0, 1, 2, ..$) during T is:

$$Pr(N = k, \lambda T) = \frac{e^{-\lambda T} \times (\lambda T)^k}{k!} \tag{3.3}$$

Let $F_{|v,w)}$ and $F_{|n,m)}$ denote the frames sent in the uni-directional links $|v, w)$ and $|n, m)$, and $\lambda_{|v,w)}$, $\lambda_{|n,m)}$ their respective arrival rates.

Accordingly, $F_{|n,m)}$ would interfere, at w, with $F_{|v,w)}$ only and only if $F_{|n,m)}$ arrives at node w during the time interval when $F_{|v,w)}$ is undergoing capture at node w receiver. Hence time T, in Equation 3.3, would correspond to the time interval during which frame $F_{|v,w)}$ occupies the channel. We denote it $\tau_{|v,w)}$:

$$\tau_{|v,w)} = \frac{S_{|v,w)}}{r_{|v,w)}} \tag{3.4}$$

where $S_{|v,w)}$ and $r_{|v,w)}$ are respectively the length and the transmission rate of frame $F_{|v,w)}$.

This way, the probability of having interference between two frames $F_{|v,w)}$ and $F_{|x,y)}$ corresponds to the probability of having one frame $F_{|x,y)}$ arriving during frame $F_{|v,w)}$ channel occupancy time $\tau_{|v,w)}$:

$$P_w^v|x, y) = Pr(N = 1, \lambda_{|x,y)}\tau_{|v,w)}) = e^{-\lambda_{|x,y)}\tau_{|v,w)}} \times \lambda_{|x,y)}\tau_{|v,w)} \tag{3.5}$$

However, since we are computing these interferences over quite long averaging periods (e.g., a couple of seconds) when compared to the transmission time of a single frame (which is of the order of milliseconds), and in order to avoid the overhead of computing the probabilities of interferences for every single frame, we see to compute it over the averaging period T instead of the frame transmission time τ. Consequently, the probability of having interferences between frames corresponding to links $|v, w)$ and $|x, y)$, with the former link as being the desired one, is the probability of having at least one frame $F_{|x,y)}$ sent during the channel occupancy time of node v:

$$P_w^v|x, y) = Pr(N \geq 1, \lambda_{|x,y)}\tau_{|v,w)})$$

$$= 1 - Pr(N = 0, \lambda_{|x,y)}\tau_{|v,w)})$$

$$= 1 - e^{-\lambda_{|x,y)}\tau_{|v,w)}}$$

and finally the physical model $SINR$ inequality (11) converges as follows:

$$SINR_{|v,w)} = \frac{SS_{|vw)}}{N_w + \sum_{x \in I_w}(1 - e^{-\lambda_{|x,y)}\tau_{|v,w)}}) \times SS_{|x,y)}} \tag{3.6}$$

The $\tau_{|v,w)}$ values are easily computed using Equation 3.4. The arrival rates values ($\lambda_{|x,y)}$) can be easily obtained, as well, by tracking how many frames, pertaining to link $|x, y)$, arrive at the receiver w.

3.4 Contention Measurement

In IEEE 802.11, contention refers to the situation where at least two stations are competing with each other for the free medium. The medium is sensed free when either there is no ongoing signal at all or when the ongoing signal(s) are below a certain carrier sense threshold.

We measure contention at a given node v by accounting for *all* frames that are heard at node v regardless of whether node v is the desired destination or not. We also account for the locally generated frames, as these latter will generate contention in the vicinity of node v. Thus, we define the *Contention Indicator* (CI_v) at a node v, where ($0 \leq CI_v \leq 1$), as follows:

$$CI_v = \frac{\sum_{i \in F_v} \tau_i}{\tau} \tag{3.7}$$

where F_v is the set of all frames heard at node v, over a certain time period τ (including frames sent or received by node v), and τ_i is the transmission time of frame i:

$$\tau_i = \frac{S}{r} \tag{3.8}$$

where S is the frame size and r is its transmission rate.

3.5 Designing a New Routing Metric

Before discussing the issue of designing a new routing metric, we would like first to address two important relevant issues: 1) the asymmetric nature of wireless link, and 2) how to disseminate the computed SINR and contention values without using broadcast frames?

3.5.1 The Asymmetric Nature of Wireless links

The quality of a wireless link definitely depends on both the reverse and forward components of the link. The fact that most multi-hop routing protocols, e.g. AODV [25], DSR [19], OLSR [20], use the forward path in order to select the reverse path and vice versa, induces a radical contradiction when only the quality of the forward path is used to select the reverse path. To cope with this, it is essential for the routing metrics to incorporate both the qualities of the forward and the reverse components of a link into a single metric. For instance, when considering interferences level as a link quality metric, the interferences levels at both destination and source should be incorporated into a single link quality metric since the link endpoints are likely to have different interferences levels as a result of different power levels, different transmission rates, and different vicinities as well. In fact, this does not apply only to interferences but to any other link quality metric as well. The following general example does further illustrate the point:

let us formally denote $|n, m)$ as the forward link from node n to node m, and $Q|n, m)$ as its quality, e.g., $SINR$, and let us assume the following arbitrary link quality values:

- $Q|S, D_1) = 3;\ Q|S, D_2) = 2$
- $Q|D_1, S) = 1;\ Q|D_2, S) = 5$

where S is a source node trying to select either node D_1 or node D_2 as its next hop.

If only the forward link quality is taken into account then S will pick D_1 since it corresponds to a better quality, e.g., larger $SINR$: $Q|S, D_1) > Q|S, D_2)$, and this is regardless of the reverse path $|D_2, S)$ which exhibits a much better quality than the reverse path $|D_1, S)$.

Ideally in such a case, S should select D_2 as a next hop instead of D_1 since the reverse path

$|D_2, S)$ is much better than the reverse path $|D_1, S)$, and thus it would cope with the difference in quality between the forward link components. When incorporating both link directions into a single metric, e.g., by using the product of the two link directions qualities, the selection of D_2 as a next hop proves logical:

$$Q|S, D_2) \times Q|D_2, S) = 10 > 3 = Q|S, D_1) \times Q|D_1, S)$$

Hence it becomes very crucial to account for the quality metrics of both the reverse and forward components of a wireless link. The formerly defined $SINR$ (3.6) and CI (3.7) parameters were accounting for only one link direction. To take into account the asymmetric nature of the wireless links, we define the $SINR$ of a link (v, w) as follows:

$$SINR_{(v,w)} = SINR_{|v,w)} \times SINR_{|w,v)} \tag{3.9}$$

where $SINR_{|v,w)}$ and $SINR_{|w,v)}$ are respectively the Signal-to-Interference-Noise ratios of the forward and the reverse unidirectional links $|v, w)$ and $|w, v)$. Similarly, we define the *Contention Indicator* of a link (v, w), where $(0 \leq CI_{(v,w)} \leq 1)$, as follows:

$$CI_{(v,w)} = CI_v \times CI_w = \frac{(\sum_{i \in F_v} \tau_v) \times (\sum_{i \in F_w} \tau_w)}{\tau^2} \tag{3.10}$$

3.5.2 How to Disseminate the Computed SINR and Contention Values?

In order for every node to compute the asymmetric $SINR$ and CI values for a link, it has to be aware of the forward $SINR$ and CI values at all its neighbors. Consequently, whenever a node w computes the *reverse* $SINR$ and CI values for all its neighbors, these values need to be disseminated to neighbors in order for them to be able to compute the *asymmetric* $SINR$ and CI values according to Equations (3.9,3.10). The question heretofore is: How to disseminate these values with minimum overhead?

Normally, broadcast probe frames are the common means to disseminate such information. However, since we are trying to minimize the overhead, we propose that every node piggybacks the

computed $SINR$ values for all *reverse* links to all its neighbors as well as its own CI value, in the default IEEE 802.11 routing broadcast frames. Upon receiving a routing broadcast frame, every node looks up its MAC address, retrieves its *forward* $SINR$ value then combines it with the locally computed *reverse* $SINR$ to finally compute the total *asymmetric* $SINR$ value of the link. The CI values are computed in a similar way.

3.5.3 A New Routing Metric

When designing routing metrics, one has to address that fundamental issue of whether a routing metric is suitable or not for a given specific routing protocol?

A systematic analysis of the relationships between routing metrics and routing protocols has been conducted in [48], where Y. Tang *et. al* empirically proved that *Isotonicity* is a must for all flooding-based protocols, e.g., HWMP [5] which is a variation of AODV [25]:

Given a quadruplet $(S, \oplus, w, \preceq)$, where S is the set of all paths, w a function that maps a path to a weight (i.e., a routing metric), $\preceq$ is an order relation, and $\oplus$ is the path concatenation operation. *Isotonicity* is defined as follows:

- *The quadruplet* $(S, \oplus, w, \preceq)$ *is isotonic if* $w(a) \preceq w(b)$ *implies both* $w(a + c) \preceq w(b + c)$ *and* $w(c' + a) \preceq w(c' + b)$*, for all* $a, b, c, c' \in S$.

To account for isotonicity, *ICE* for a given link (v, w) is computed as follows:

$$ICE_{(v,w)} = \frac{CI_{(v,w)}}{SINR_{(v,w)}} \times t \tag{3.11}$$

where $CI_{(v,w)}$ and $SINR_{(v,w)}$ are given respectively by Equations (3.9, 3.10) and t is the transmission time. *ICE* of a given path p is the sum of the *ICE* values of its single-hop links:

$$ICE(p) = \sum_{l \in L_p} ICE_l \tag{3.12}$$

where L_p is the set of links in path p. This way, *ICE* penalizes routes with more hops.

Thus, we can easily see that *ICE* is isotonic since:

- *ICE* of a route is the sum of the *ICE* values of its hops.
- *ICE* of a hop is always a non-negative value.
- *ICE* is inversely proportional to the link quality, i.e., the best link quality corresponds to least ICE value.

3.6 Experiments

The coming experiments are run in a real-world indoor IEEE 802.11s WMN testbed. The testbed is open-source and linux-based. The implementation details are covered in section 5.

3.6.1 Topologies

We used two different topologies to test the new routing metric. In the first topology (see Figure 3.1) we deployed a wireless mesh network composed of 11 mesh nodes, one 802.11 client, and one 802.3 station. The 11-node WMN is connected to the Internet. Using Iperf [65], we created UDP and TCP connections between the 802.11 client (connected to the WMN through a Mesh Access Point) and the 802.3 server (connected to the Internet).

In the second topology, we connected two WMNs (see Figure 3.2) through Internet. The WMN to the left is composed of 7 nodes, and the WMN to the right is composed of 8 nodes. Similarly to the first topology, UDP and TCP connections were created using Iperf. The major difference in this topology is that both the client and the server are 802.11 stations located into two different WMNs.

These WMNs were deployed at the second floor of the Shelby Center at Auburn University.

3.6.2 Settings

The mesh points network interface cards run the Madwifi driver [64]. HWMP PREQ (Path REQuest) messages were periodically sent every 4 seconds and ETX probe frames were 1024 Bytes long and periodically sent every 1 second [16]. Rates were extracted through the ”/Proc”

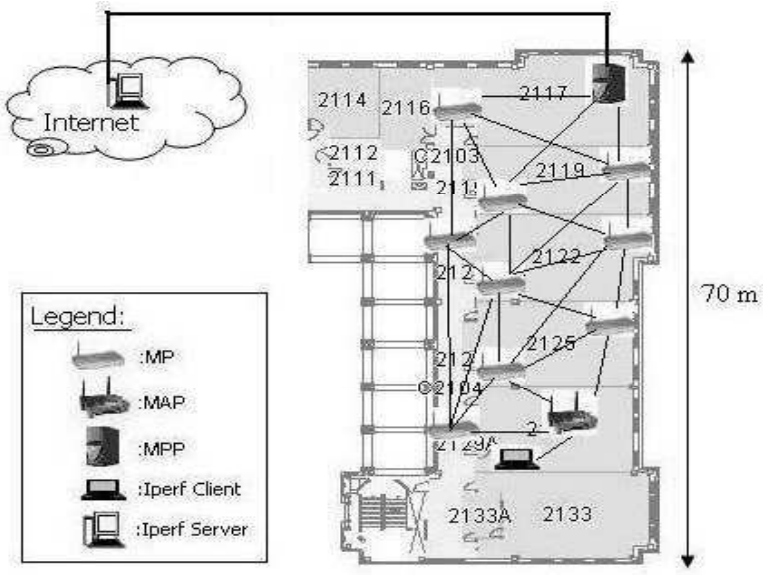

Figure 3.1: IEEE 802.11s WMN Tesbed, Topology 1

system files and the rate adaptation algorithm was kept to the default one, which is SampleRate [60] in Madwifi drivers. Madwidi allows the creation of virtual interfaces; two virtual interfaces were created for ICE computation in every WMN node: One virtual interface is set to the *Monitor* mode and the other is set to the *ad-hoc* mode. The *Monitor* interface is to overhear all ongoing signals.

In both topologies, the WMN network interface cards are operating in the 802.11g channel 1 [62], while the clients are operating in the 802.11g channel 11 in order to minimize the interferences between the client and the WMN nodes.

Both TCP and UDP traffic were supported in order to fully ascertain the functionality of the WMN testbed. Real web surfing sessions, using the WMN testbed as a backhaul, were tested and successful.

During experiments, the TCP and UDP sessions were repeatedly run, using Iperf, over periods of 20 seconds and averaged to plot the network throughput in function of client bandwidth.

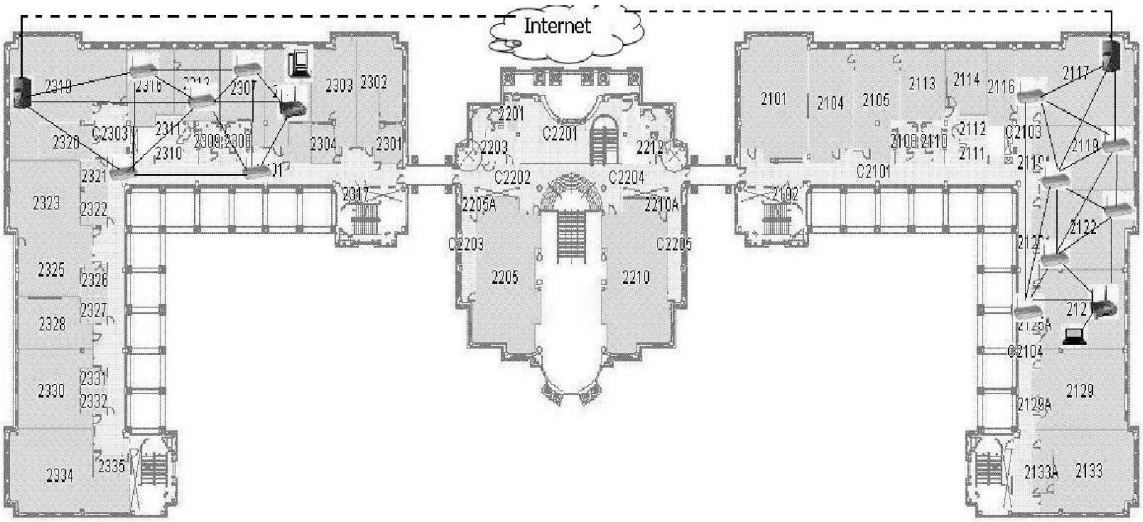

Figure 3.2: IEEE 802.11s WMN Tesbed, Topology 2

Routing Metrics	**Topology 1**		**Topology 2**	
	UDP	TCP	UDP	TCP
ETX	1.18	0.95	1.19	0.94
Airtime	1.10	0.96	1.28	0.92
ICE	1.36	1.18	1.41	1.01

Table 3.1: UDP and TCP Throughput Averages (*in Mbps*) Comparison

3.6.3 Results and discussion

In both topologies, we see that ICE generally outperforms both ETX and Airtime (see Figures 3.3, 3.4, 3.5, 3.6).

In topology 1, ICE outperforms ETX with an average of 15% for UDP and 24% for TCP traffic. ICE outperforms also Airtime with an average of 23% for both UDP traffic and TCP traffic (See Table 3.1). In topology 2, ICE outperforms ETX with an average of 18% for UDP and 7% for TCP traffic. ICE outperforms also Airtime with an average of 10% for UDP traffic and an average of 9% for TCP traffic.

On average, ICE throughput exhibits an improvement of 16% in average over ETX and of 16% over *Airtime* in average as well. This relatively good improvement is a proof of the ICE concept which relies on accurate link quality measurement. Yet, we have the strong belief that ICE outperformance could be greater if deployed in large-scale WMNs with more paths redundancy: In

these relatively small-sized topologies we used, ETX, Airtime and ICE ought to select the same paths when there are not enough alternatives. However, the fact that ICE is a passive approach, guarantees a minimum outperformance thanks to not using broadcast probes as well as to its responsiveness to the time-varying aspect of wireless link quality. Unlike other passive approaches, ICE does not suffer from the shortcoming of probing idle links as the absence of data is a quality to detect by itself since the absence of traffic in ICE is interpreted to represent minimum interferences and minimum contention levels.

Last and not least, we believe that the ICE concept can benefit other techniques, especially the multi-channels/Radios one. Migrating and adapting ICE for use with multiple-channels/radios is our next future research step.

As mentioned, formerly in Section 1.4, these last experiments exhibited a ping-pong effect in

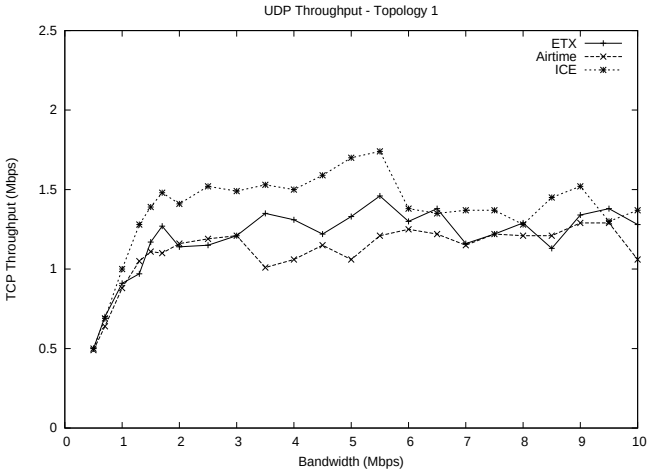

Figure 3.3: UDP Throughput - Topology 1

the network behavior with IEEE 802.11s Airtime. The effect seemed first to be an anomaly, and after a thorough re-investigation of the experimental settings as well as the source code, the effect turned out to be a persistent one. Next chapter highlights the subtleties of the effect and proves its origin .

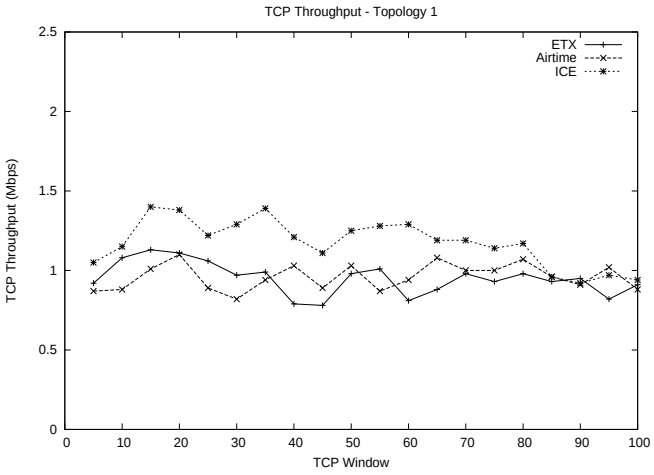

Figure 3.4: TCP Throughput - Topology 1

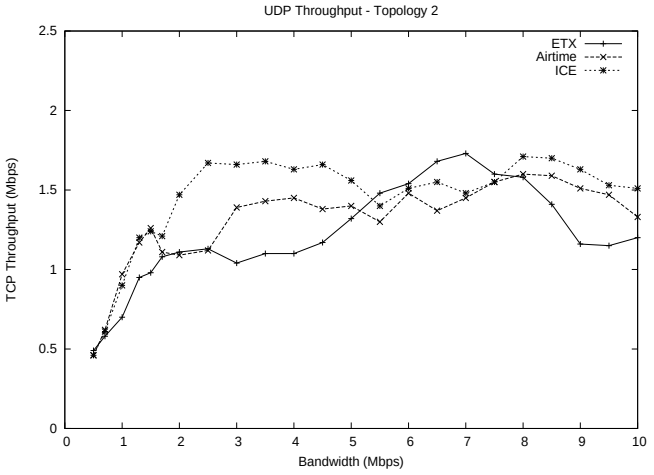

Figure 3.5: UDP Throughput - Topology 2

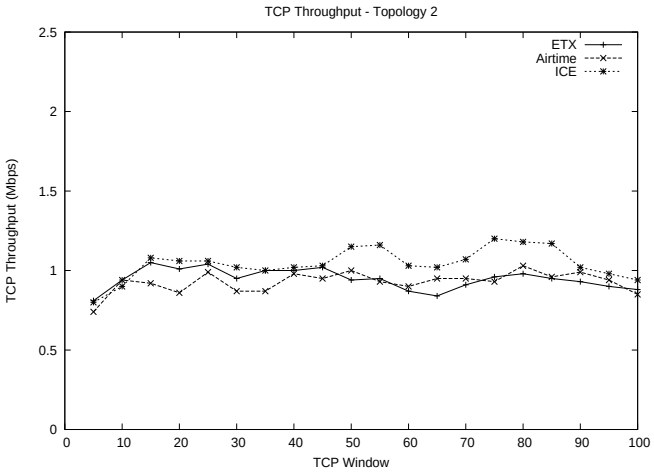

Figure 3.6: TCP Throughput - Topology 2

Chapter 4

The IEEE 802.11s Airtime ping-pong effect

In this chapter we introduce the IEEE 802.11s Airtime ping-pong effect, we highlight its nature, and unveil the cause behind it. Using extensive ns-3 simulations, we show that the main cause behind the effect is the transmission rate variance by means of adaptive rate control algorithms. By proving that the effect is an inherent behavior, we introduce a novel research direction based on shaping *ping-pong-aware* mechanisms that adapt the network to the effect. In this context, we propose a ping-pong-aware mechanism that operates at the level of the IEEE 802.11s HWMP routing protocol and that detects when a link undergoes such an effect. By doing so, the mechanism tries to adapt the routing protocol by adjusting the link Airtime metric value. The mechanism can either favor or disfavor the concerned link based on the current stage of the ping-pong effect. We show that effect has two stages.

4.1 Overview

While experimenting with the default IEEE 802.11s HWMP (Hybrid Wireless Mesh Protocol) routing protocol [5] using IEEE 802.11s Airtime [6] as a routing metric, we observed a noticeable ping-pong effect in the network overall throughput. The effect consisted of having the network throughput frequently oscillating, during the lifetime of the experiments, between high and low throughput values. At first, we thought the effect merely stems from an implementation anomaly, e.g., a bug. However, after a through examination of the implementation source code, and after taking over the same experiments using ETX (Expected Transmission Count) [16] as a routing metric instead of Airtime, we noticed a considerable mitigation in the effect. Thus, and as a preliminary explanation, we postulated that this behavior is *only* relevant to the IEEE 802.11s Airtime routing metric. In this context, and in an attempt to shed further light into the effect, we

conducted a literature review whereby we, surprisingly, found only a unique reference to the effect [9]. This latter reference is superficially addressing the effect, and condemning it as a perilous behavior without solid clarification. Furthermore, the authors even changed the IEEE 802.11s formulation of Airtime [6] to render it independent of the transmission rate variable, and this in order to eliminate the "so-named" perilous behavior. Given that the ping-pong effect ought to impact the overall performance of this new IEEE 802.11s technology as it pertains to an integral part to the IEEE 802.11s HWMP routing protocol which is the Airtime routing metric, and given the very scarce literature on the effect, we think that this effect should be allotted further research, and that a good characterization of it can help improve the overall network performance by merely accounting for the effect presence.

In contrast to ETX [16], which accounts solely for frame error rate, Airtime accounts for both frame error rate and bandwidth. When an active link becomes loaded, the routing protocol "disfavors" it by increasing its Airtime metric value: This latter increase would result from a "detected" increase in the loss rate as a consequence of the link quality being degrading because of it becoming loaded, and thus the routing protocol will advice and select another less-loaded link. However, in its turn, the new selected link will become loaded, and this will incite the routing protocol to start over the same procedure again. This latter may merely result in re-selecting the first discarded link, especially when there is not enough link redundancy. In this manner, the procedure continues on, and thus inducing a "ping-pong" effect which consists of the cyclic procedure of switching back and forth between alternative links.

In fact, the ping-pong effect should be witnessed with *any* routing metric, regardless of its formulation, since the main cause behind it is having active links becoming alternatively loaded and unloaded, a fact that can happen under whatever routing metric. However, the exception is that the effect is intense with Airtime, and this is further seconded when compared to the ETX ping-pong effect. This leads to the postulate that a candidate reason, for this intense Airtime ping-pong effect, is the dependence of Airtime on the transmission rate since ETX is not so dependent.

Still, Airtime exhibits a relatively weak ping-pong effect when used with a non-adaptive rate control algorithm such as Constant rate. Indeed, during experimentation, we observed that the effect is noticeably more intense when used with adaptive rate control algorithms (e.g., ARF [55] and AARF [58]) than when used with Constant rate. In such a case, and when a link quality degrades because of it becoming loaded, these algorithms tend to decrease the transmission rate in order to avoid further losses since the loss rate is becoming high. Since Airtime is inversely proportional to the transmission rate, this will increase the Airtime value. On the other hand, when the link becomes unloaded, the adaptive rate control algorithms will increase the transmission rate in order to profit from the good link quality, and thus decreasing the Airtime value. This way, we see that the Airtime metric is *simultaneously* impacted by the variances in both the loss rate and the transmission rate, which makes it very prone to frequent changes, a fact that ought to be the principal reason behind such an effect. The loss and transmission rates are correlated to each other by means of adaptive rate control algorithms.

In this dissertation, we present a thorough study of the effect. Using extensive ns-3 simulations, we depict and analyze the strong correlation of the effect to the underlying rate control algorithms. We show that the effect is an inherent behavior and not necessarily a perilous one, by showing that IEEE 802.11s can perform better under intense instances of the ping-pong effect.

An accurate characterization of the effect can benefit the IEEE 802.11s overall network performance by shaping new mechanisms that account for it. In this context, we present a simple mechanism that, by detecting when the system is under the effect, reacts to it and adapts the routing protocol to improve the network performance. The mechanism deems a muli-hop wireless path as a chain of one-hop links whose strength is *equal* to the strength of its weakest link. The mechanism is O(1), decentralized, and thus easy-to-deploy. The proposed mechanism can reach up 40% improvement in the overall network performance. This result is very encouraging towards further research on the effect and towards shaping ping-pong aware mechanisms, specifically at the level of routing and rate control algorithms, that by accounting for this inherent behavior can improve

the overall network performance.

4.2 Characterizing The Airtime Ping-Pong Effect

As introduced formerly, while experimenting with Airtime using an IEEE 802.11s WMN indoor testbed [4], we noticed a strong ping-pong effect in the behavior of the network. The effect consisted of a continuous swing in the network throughput (As reported by Iperf [65]). At first, we thought this was merely an exception in the network behavior due to a certain undetected anomaly, e.g., a code bug. However, and after a thorough inspection of the source code [4], we run the same experiments again and using different settings: The effect turned out to be persistent. To gain more insight on the effect, we took over the same experiments but using ETX [16] as a routing metric instead of Airtime, and it turned out that ETX exhibits a weak ping-pong effect when compared to the one of Airtime. Furthermore, and surprisingly, Airtime performed slightly better than ETX despite the effect persistence. At that point, we postulated that the effect has to do only with Airtime since ETX was exhibiting a weak effect.

In an attempt to decipher this unique behavior, we decided to study it separately using simulations this time as this would allow for more extensive experiments, contrary to real-world testbeds where experiments are expensive in terms of time, settings, and management. Furthermore and most important is that in such a kind of problem, different topology settings need to be deployed in order to allow for various traffic patterns. A fact which is not that affordable in real-world testbeds where the geographical limitations of the deployment space (e.g., buildings with offices) does not allow for such a diversity in multi-topology deployments.

Using ns-3 [54], we run extensive experiments with different settings, e.g., bandwidth, rate control algorithms, and topologies. The simulation results were seconding the observation of the real-world experimentation concerning the persistence of the ping-pong effect in IEEE 802.11s WMNs under Airtime as a routing metric. Next section highlights the experiments.

4.2.1 Experimenting the Ping-Pong effect

This experiment is but one among the numerous set of experiments we carried in order to ascertain the persistence of the Airtime ping-pong effect (*Seeking dissertation space compactness, only one experiment is outlined in this section*).

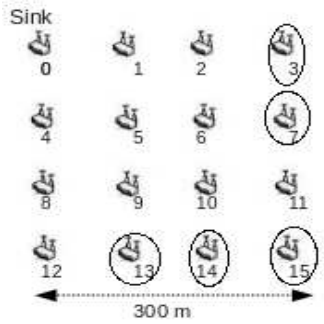

Figure 4.1: Ping-pong effect: Topology

4.2.2 Experimental Settings

The WMN in this sample experiment, is a 4x4 grid topology with a distance of 300 meters (see Figure 4.1), that simulates the case where WMNs are used for last-mile Internet access. Node(0) is set as a sink representing the WMN gateway, and nodes (3, 7, 15, 14, and 13) are set as sources representing WMN access points. The sources are transmitting UDP traffic, at a constant rate (10 Mbps), towards the sink. All source nodes start transmitting at the same time and continue transmission for periods of 60 seconds.

The WMN is configured to run the *proactive* HWMP routing protocol since the traffic is always directed towards the sink node, and thus shaping a tree-like topology with the sink as the tree root (Proactive HWMP is a tree-based routing protocol). The inter-time between the HWMP proactive PREQ (Path Request) broadcast messages is 2 seconds, and we used the following rate control algorithms were: ARF [55], AARF [58], AMRR [58], ONOE [59], and Constant Rate.

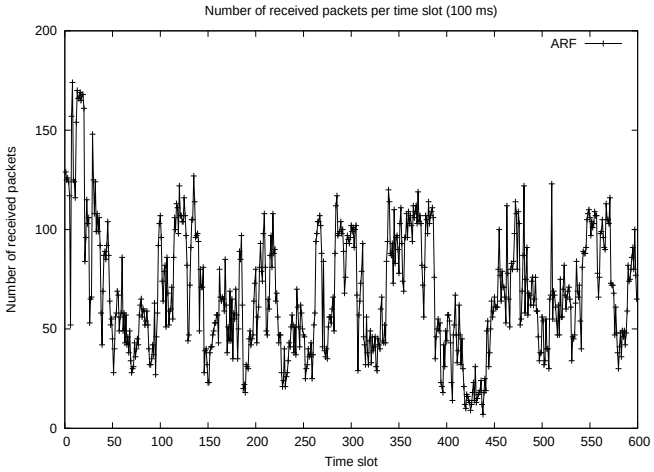

Figure 4.2: ARF: Aggregated number of received packets

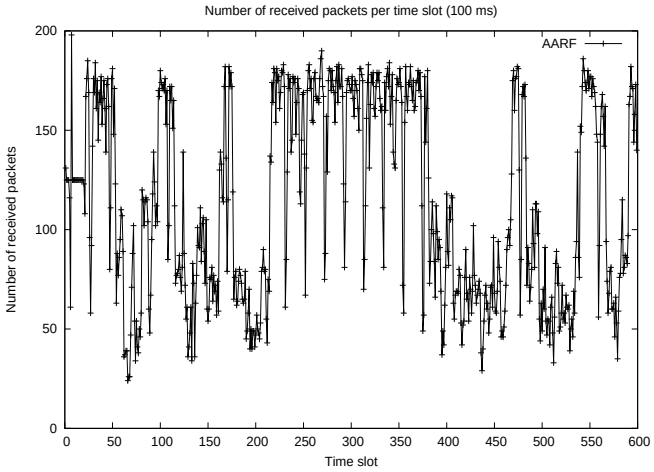

Figure 4.3: AARF: Aggregated number of received packets

4.2.3 Results

We tracked the variance of the *aggregated* received packets at the sink (i.e., the number of packets received from *all* five source nodes) by means of a packets counter which is initially set to zero and incremented every time the sink receives a packet. After every time slot of 100 ms, the counter is saved and re-initialized back to zero. This process continues for the life time of the experiment (60 seconds), and thus tracking the number of received packets in a total of 600 time slots. Figures 4.2, 4.3, 4.4, 4.5, and 4.6 plot the number of aggregated received packets, at the sink, for every 100 ms time slot and for each of the underlying rate control algorithms.

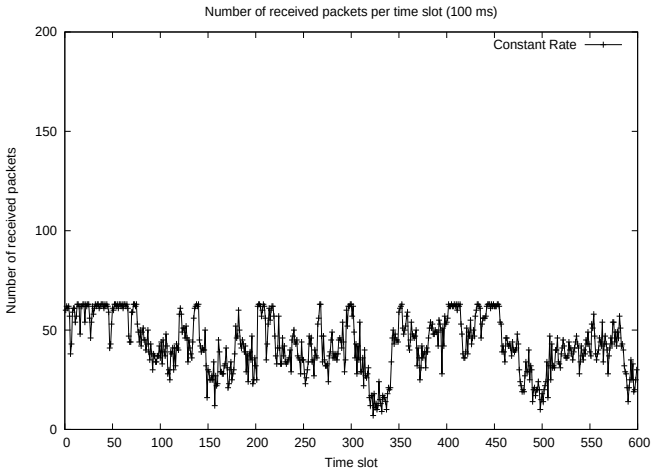

Figure 4.4: Constant Rate: Aggregated number of received packets

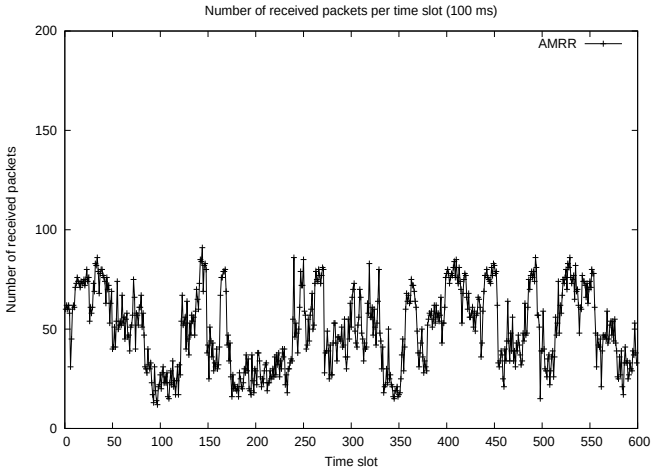

Figure 4.5: AMRR: Aggregated number of received packets

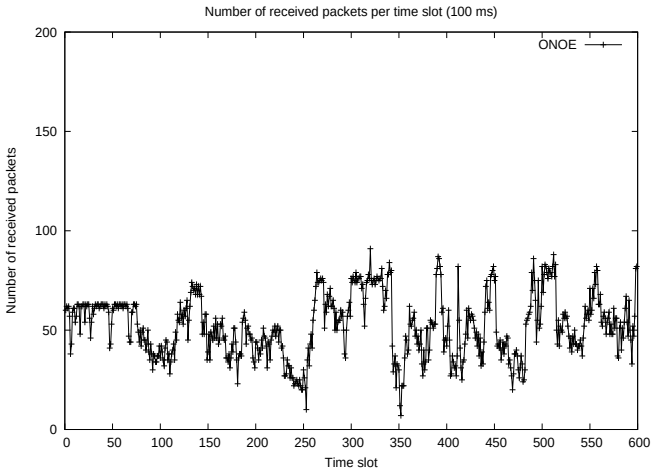

Figure 4.6: ONOE: Aggregated number of received packets

In the latter figures, we clearly notice that the different rate control algorithms all exhibit ping-pong effects but with different magnitudes. To quantify these magnitudes, we computed the standard deviations, of the number of received packets per time slot, for the different rate control algorithms, see Figure 4.7. In the last figure, we see that the different rate adaptation algorithms have, indeed, different ping-pong effect magnitudes. Airtime exhibits the strongest effect under AARF and ARF, and the weakest effect under Constant Rate.

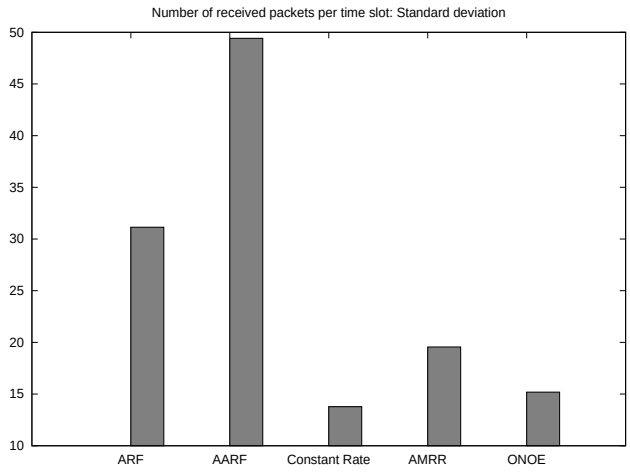

Figure 4.7: Ping-pong effect magnitudes

This last result was *general* to *all* experiments we ran except that ARF and AARF were alternating in terms of who exhibited the strongest effect. Definitely, Airtime always exhibits the weakest ping-pong effect when using Constant Rate. As a primary observation, we postulated the following:

1. There is a strong correlation between the ping-pong effect and the underlying rate control algorithm.

2. The ping-pong effect is not that perilous.

These latter observations are discussed and analyzed in the following section.

4.3 Analysis

According to Equation (2.1), Airtime depends on both the loss rate (e_{fr}) and the transmission rate (r). With adaptive rate control algorithms [68], the transmission rate is continuously adjusted, and thus changing, to accommodate the varying link quality. This latter is mainly represented by the observed loss rate. Under a good link quality, adaptive rate control algorithms increase the transmission rate in order to profit from the actual good link quality. However, in such a case, and after a certain period of time, the link quality will "*forcibly*" degrade because of the *expected* increase in the link load:

- Since adaptive rate control algorithms increase the transmission rates as a reaction to a good link quality, stations will send at higher rates and thus will generate more traffic (i.e., more load).

- Airtime will decrease, in order to reflect the actual good link quality, since the transmission rate did increase (*Airtime is inversely proportional to transmission rate*). This will urge the routing protocols of other stations to favor the link and route their traffic through it, and thus generating more traffic (i.e., more load).

Thus, every unloaded link is "condemned" to become loaded after a while, assuming there is enough traffic. When this happens, the loaded link will witness more frame losses, mainly because of congestion (due to high load), and the link quality will start deteriorating. In such a case, the adaptive rate control algorithms, will decrease the transmission rates in order to cut down the load, and this will be captured by Airtime through *both* the increase in the frame loss and the decrease in the transmission rate. Thus the parallel change in both loss rate and transmission rate will *"accelerate"* the procedure of increasing Airtime, and thus advising the routing protocol of the local station, as well as of other stations, to disfavor the link and redirect their traffic towards other less-loaded links.

In this way, we see that adaptive rate algorithms largely impact the Airtime metric values, and "*accelerate*" both processes of loading and unloading links, a fact which intensifies the process of

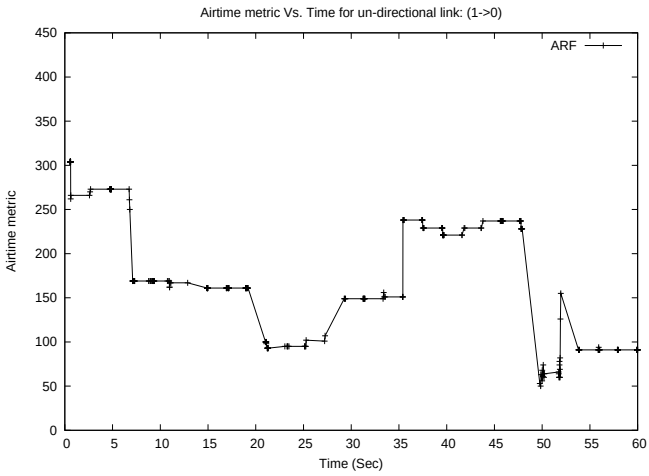

Figure 4.8: ARF: Airtime metric variance

switching back and forth between loaded and unloaded links, and thus intensifying the ping-pong effect. On the other hand, when no adaptive rate algorithm is used (i.e., Constant rate), the process of loading and unloading links is "*slower*" as Airtime is impacted by the *sole* variation in the observed loss rate, and not by the variation in the transmission rate as this latter remains constant.

Finally, we assert that the Airtime ping-pong effect is highly correlated to adaptive rate control algorithms, and that such a correlation is due to the induced changes in the transmission rates as a reaction to the observed link quality.

4.3.1 Links Behaviors

To further ascertain the last assertion, we tracked the behavior of individual links in terms of the variances in transmission rates, loss rates, and Airtime values. We used three different rate control algorithms: ARF, AARF and Constant rate. Using the same former topology (see Figure 4.1), we analyzed 10 individual links. These latter *all* exhibited the same behavior in terms of the correlations between transmission rates, loss rates, and Airtime. Figures [4.8 - 4.16] depict such a behavior for the sample uni-directional link from node(1) to node(0).

Figures (4.8, 4.9, 4.10) depict the Airtime variance, Figures (4.11, 4.12, 4.13) depict the transmission rate variance, and Figures (4.14, 4.15, 4.16) depict the loss rate variance. Note that

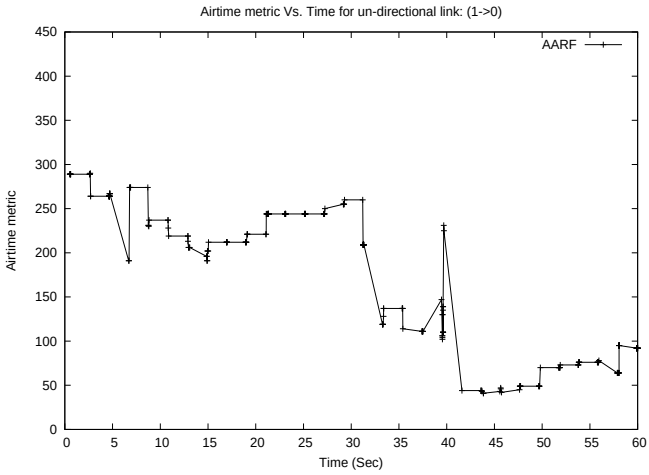

Figure 4.9: AARF: Airtime metric variance

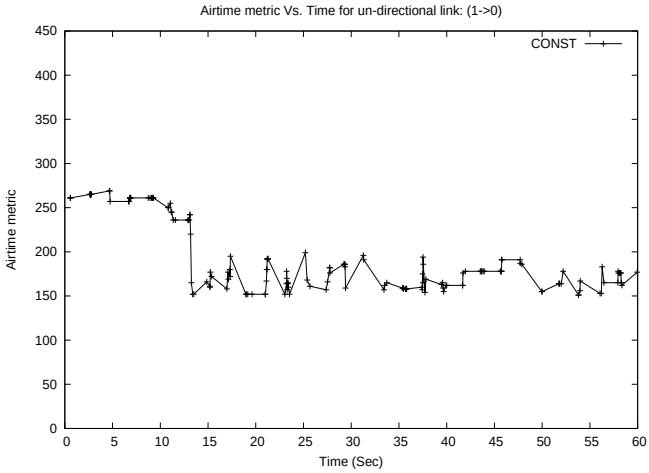

Figure 4.10: Constant Rate: Airtime metric variance

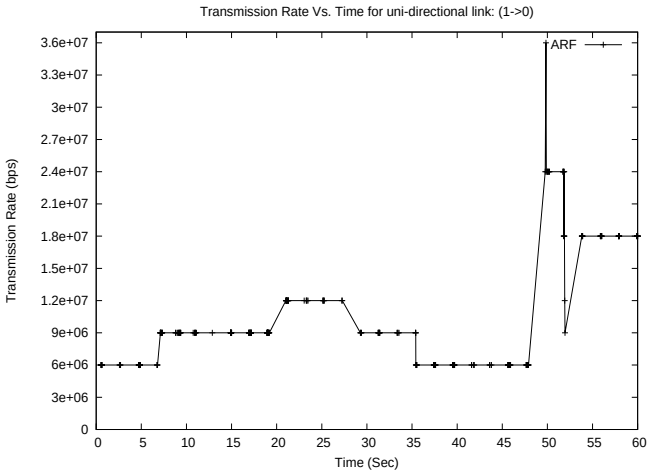

Figure 4.11: ARF: Transmission rate variance

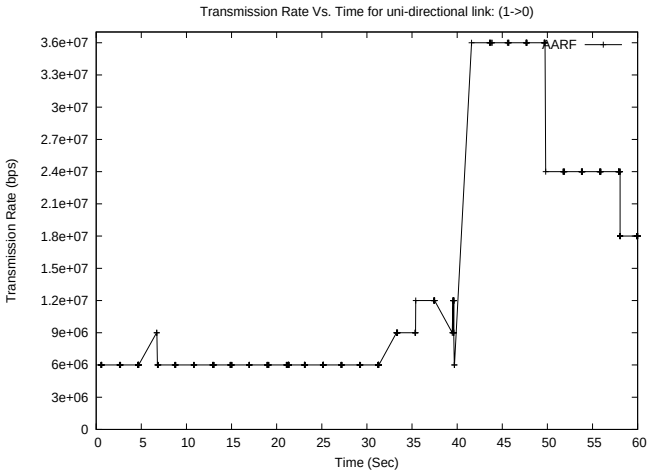

Figure 4.12: AARF: Transmission rate variance

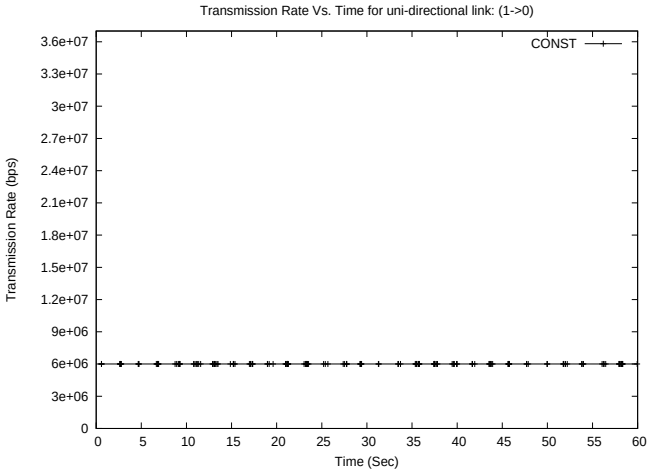

Figure 4.13: Constant Rate: Transmission rate variance

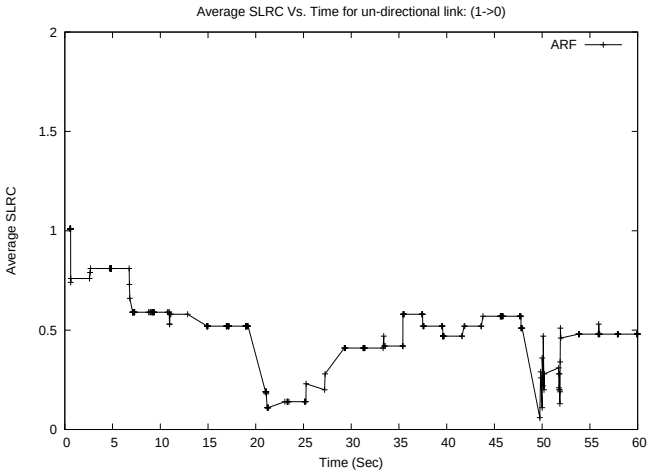

Figure 4.14: ARF: Loss rate variance

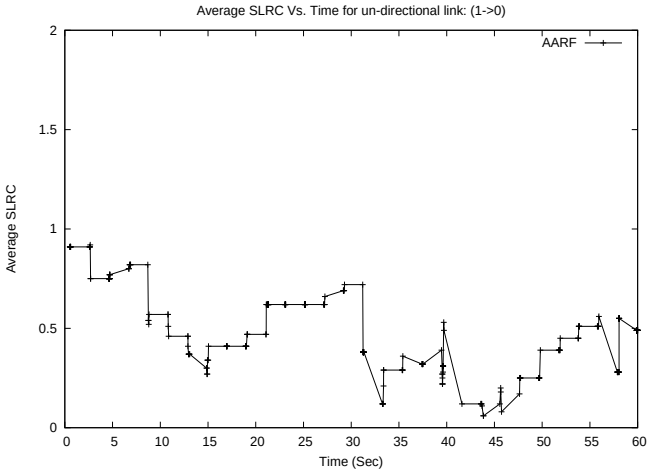

Figure 4.15: AARF: Loss rate variance

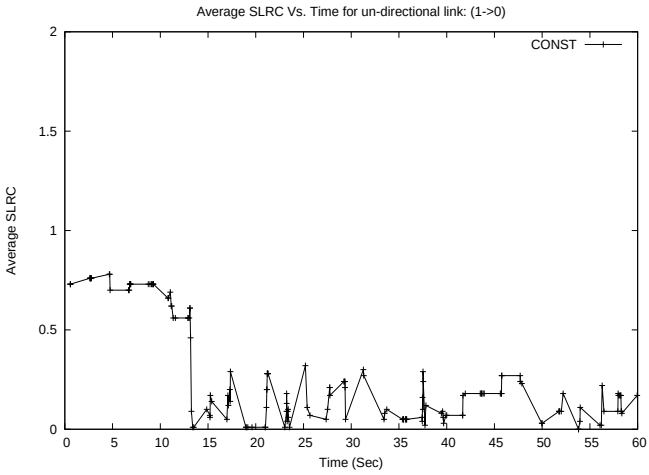

Figure 4.16: Constant Rate: Loss rate variance

for loss rate, we plotted the average SLRC (Station Long Retry Count) as this latter is the metric used by ns-3 to represent loss rate.

AARF switches transmitting at the high rate of 36 Mbps, around the 40^{th} second (see Figure 4.12), and keeps transmitting so for approximately 10 seconds. In parallel, we notice that the AARF loss rate starts dropping down at the 40^{th} second as well, and then starts slightly increasing at the 44^{th} second (see Figure 4.15). This illustrates the very rationale behind adaptive rate control algorithms which consists of maximizing profiting from the good channel condition (low loss rate values) by transmitting the maximum possible packets (i.e., by increasing the transmission rate). ARF exhibits a quite similar behavior starting around the 47^{th} second (see Figure 4.11) by switching to the high transmission rate of 36 Mbps. However, ARF quickly switches back to the rate of 24 Mbps afterwards. In parallel, we notice that the loss rate started decreasing (good channel) at the 47^{th} second as well, see Figure 4.14, and then increasing back around the 50^{th} second. This explains the quick dropping back in the ARF transmission rate from 36 Mbps to 24 Mbps.

Under the Constant Rate control algorithm, we notice that Airtime is quite stable around the same value (160), see Figure 4.10, regardless of the frequent small fluctuations that are due to the parallel fluctuations in the loss rate (see Figure 4.16) and to the maintained Constant rate (see Figure 4.13). On the other hand, ARF and AARF Airtime values, see Figures 4.11 and 4.12, are more varying, a fact which makes the link more prone to the process of being favored and disfavored by routing protocols, and thus prone to witness a more intense ping-pong effect. Indeed, with the Constant Rate control algorithm, the link Airtime value is not that varying, a fact which renders the link stable in the eyes of the routing protocol. This latter will not go through the process of switching back and forth in selecting the link, and thus minimizing the intensity of the ping-pong effect.

These last facts do further sustain the former assertion about the strong correlation between adaptive rate control algorithms and the ping-pong effect. Still, the other fold we have to assert is whether the ping-pong effect is a perilous behavior or not?

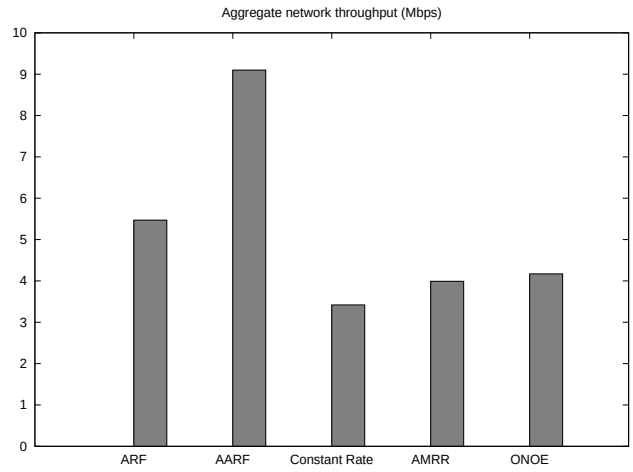

Figure 4.17: Network throughput and Rate control algorithms

4.3.2 Is the Airtime Ping-pong Effect a Perilous Behavior?

Being an exceptional behavior, the ping-pong effect tend to qualify itself of being perilous merely because of the ”unusual” frequent oscillations in the network throughput. However, what we will try to sustain, in this section, is that the effect is not that perilous as it seems to be [9], and that it is an inherent behavior instead.

By looking at the former numbers of received packets per time slot, see Figures (4.2 4.3, 4.4, 4.5, and 4.6), we notice that the rate control algorithms that exhibit strong ping-pong effects (e.g., AARF and ARF) are oscillating between ”*very*” high and low values of received packets (The low values are not qualified of ”*very*” because they are quite in the same range ([0, 30]) for all rate control algorithms). To quantify these last facts, we computed and plotted the corresponding network throughput values, see Figure 4.17. By comparing these latter values to the corresponding ping-pong effect standard deviations, see Figure 4.7, we do clearly draw that the rate control algorithms that exhibit the strongest ping-pong effects are the ones that have the highest network throughput.

These facts are sustaining that the ping-pong effect is not that perilous, if not to say a healthier behavior, since strong ping-pong effects, in the last experiments, exhibited parallel strong throughput values. We do further sustain the last assertion by drawing the following analogy:

In the former section (Section 4.3), we outlined that the main cause of the ping-pong effect is

the *accelerated* process of switching back and forth between loaded and unloaded paths. This *acceleration* is caused by the frequent changes on the link transmission rate due to the adaptive nature of the underlying rate control algorithm. In fact, when a link is loaded, it becomes very natural, and even a *must*, for it to be alleviated by reducing its load. On the other hand, when a link is unloaded, it becomes very beneficial for it to serve more traffic. Thus, we do analogically compare the ping-pong effect to a "*breathing*" mechanism whereby systems, under harsh circumstances (analogy to a congested network), are repeatedly, and strongly, switching back and forth (ping-pong effect) between the inhalation (more load) and exhalation (less load) processes. More thoroughly and precisely, a steady and "harmonious" switching behavior, even if strong, is a symptom of a "healthier" system, and such was the case with AARF in the former experiment (see Figure 4.3), where the switching back and forth is indeed strong (e.g., high received packets) and frequent (i.e., high standard deviation, see Figure 4.7), and the corresponding network throughput is the highest as well. To further generalize these assertions, we conducted further experiments that are highlighted in the next section.

Scenario Seq#	(Node#, Node transmission start time (Sec))				
1	(3,0)	(4,9)	(11, 8)	(12, 38)	(10, 21)
2	(8,0)	(3,26)	(4, 9)	(9, 39)	(1, 7)
3	(4,0)	(13,24)	(9, 5)	(6, 31)	(11, 7)
4	(8,0)	(3,18)	(7, 12)	(5, 15)	(15, 30)
5	(3,0)	(14,23)	(1, 20)	(9, 15)	(4, 9)
6	(5,0)	(13,8)	(6, 6)	(8, 12)	(14, 25)
7	(13,0)	(7,21)	(2, 39)	(4, 32)	(10, 5)
8	(12,0)	(14,19)	(11, 27)	(7, 23)	(5, 13)
9	(5,0)	(6,1)	(13, 22)	(8, 5)	(3, 6)
10	(9,0)	(6,2)	(11, 19)	(4, 9)	(2, 31)

Table 4.1: Scenarios Settings

4.3.3 Generalization

In these experiments, five sources nodes, whose locations and transmission starting times are randomly picked, are deployed in the same 4x4 grid topology as in Figure 4.1, and with Node(0) always representing the sink. The sources traffic generation rates are made variable in order to experience with different network loads and congestion levels.

Using 10 different scenarios, 10 different traffic generation rates, and 3 different rate control algorithms, we run a total of 300 experiments. Table 4.1 depicts the settings of the randomly generated scenarios.

The experiments were run for a duration of 60 seconds. Note that the transmission times were randomly generated in the [0, 40] seconds interval in order to always keep the last 20 seconds, of the lifetime of the experiments, witnessing simultaneous transmissions of all existing nodes. Besides, the first randomly picked node is always transmitting at second zero.

Figure 4.18 depicts the *average* number of received packets, at the sink, for each of the 10 scenarios, using the three different rate control algorithms (ARF, AARF, and Constant rate). The average is obtained by using 10 different source traffic generation rates ranging from 1Mbps to 10Mbps with a 1 Mbps step. We notice that Airtime is considerably performing better with ARF and AARF than with Constant Rate, and this holds for all 10 scenarios. This generalizes the result in Section 4.3.2, see Figure 4.17, about having the worst Airtime performance when using Constant rate. Furthermore, we notice that there is, indeed, a strong correlation between the network performance and the strength of the ping-pong effect. To highlight such a correlation, we plotted the average standard deviations of the number of received packets (which we named the "ping-pong effect index"), see Figure 4.19. By matching each of the three graphs in Figure 4.18 to its corresponding graph in Figure 4.19, we clearly notice such a correlation. This result generalizes the assertion made in Section 4.3.2, see Figures 4.17 and 4.7, about the correlation between high network throughput and strong ping-pong effects. This proves that the Airtime ping-pong effect is indeed not a perilous behavior, if not to say a healthier one.

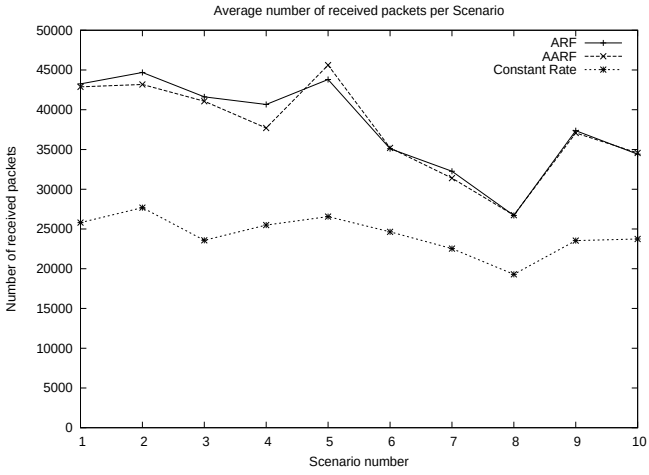

Figure 4.18: Throughput Comparison

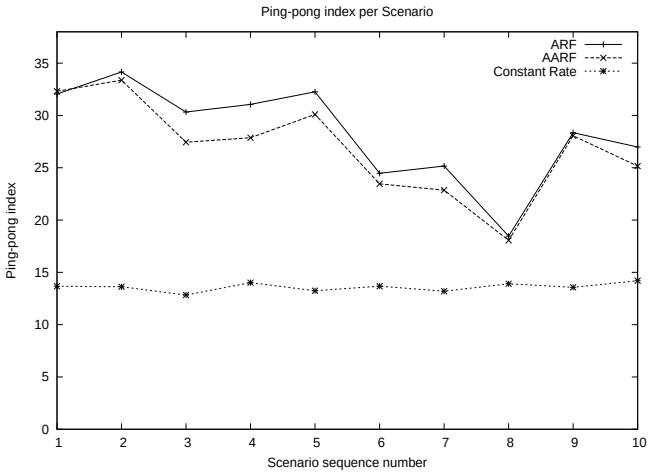

Figure 4.19: Ping-pong effects comparison

4.4 Ping-pong-aware Mechanisms

Since the Airtime ping-pong effect is an inherent behavior in IEEE 802.11s wireless mesh networks, it becomes necessary to (*re*)shape mechanisms to be aware of such an effect. To our knowledge, there no such mechanism, in literature so far.

Every ping-pong-aware mechanism, and regardless of its functionality, should first be able to detect when a link is witnessing, or about to witness, the ping-pong effect. Hence, a crucial pre-requisite step towards designing ping-pong-aware mechanisms, is the "accurate" detection of the effect occurrence.

4.4.1 Ping-pong Effect Detection

From the experimental observations, as well as the analysis, we conducted so far, we see that the ping-pong effect is made of two phases (see Figure 4.20):

- Ping phase: In this phase, which lasts for a duration Δ_{Ping}, the link is witnessing a high load and the throughput is sharply decreasing because of the high loss rate, the decrease in transmission rate, and the routing protocol re-directing frames through other links.
- Pong phase: In this phase, which lasts for a duration Δ_{Pong}, the link is in the process of recovering from the previous phase and witnessing a monotonic increase in the throughput. This is due to the low loss rate, the increased transmission rate, and the routing protocol "favoring" frames to be forwarded through the link.

Accordingly, we suggest a ping-pong-detection mechanism that tries to capture if the system is under one of the following phases:

- Ping phase
- Pong phase
- Ordinary phase (no ping-pong effect)

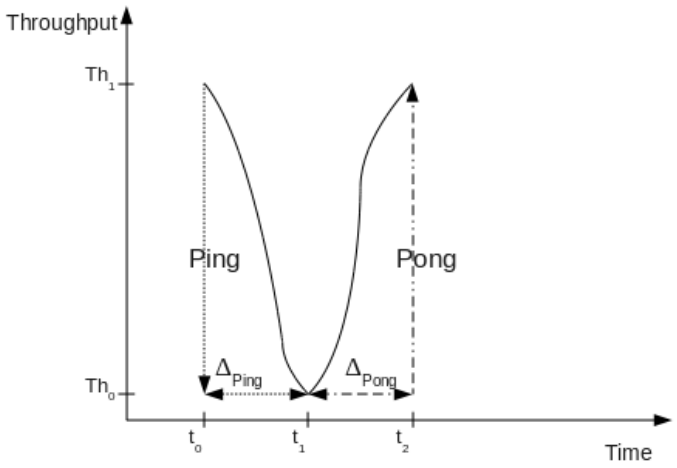

Figure 4.20: The Ping-pong Effect: Illustration

Discriminating between the ping and pong phases is very crucial as a ping-pong-aware mechanism should react differently to each of the phases since these latter are completely different from each other in terms of link quality aspects, e.g., loss rate, transmission rate, and link throughput. We suggest the following *conservative* ping-pong-detection algorithm:

Alg. 1 Ping-pong effect detection Algorithm.

---

if loss rate $increases$ ***and*** transmission rate $decreases$ **then**
    mark link to be in $Ping$ phase
else if loss rate $decreases$ ***and*** transmission rate $increases$ **then**
    mark link to be in $Pong$ phase
else
    mark link to be in $Ordinary$ phase

The algorithm is *conservative* in that it makes *sure* that a system is indeed in one of the two ping-pong phases by requiring that *both* transmission rate and loss rate are *simultaneously* changing. This should cope with the cases where only the loss rate is changing, which is a very common fact (see Figures 4.14, 4.15, 4.16).

4.4.2 A Ping-pong-aware Mechanism

Using the former ping-pong-detection algorithm, we present a ping-pong-aware mechanism that operates at the level of routing. The mechanism is O(1) and decentralized. The rationale behind the mechanism is two-fold:

- Help the network decreasing the high loss rate witnessed during the Ping phase because of the link becoming loaded.
- Help the network further increasing the throughput by profiting from the good link quality witnessed in the Pong phase.

The mechanism accomplishes its goals by the following:

If a link is detected to be in the Ping phase, we do further increase its Airtime metric value in order to prevent, as much as possible, the routing protocol from forwarding further frames through the link, and thus decreasing the loss rate. The idea behind is to make the link look worse by giving it more weight, and thus labeling it as a candidate *weakest chain link*: The strength of a chain is *equal* to the strength of its weakest link. By analogy, a multi-hop wireless path is a chain of one-hop links. However, the way routing algorithms are measuring the *strength* (i.e., quality) of a multi-hop path is by summing up the "strengths" of every link [16, 17]. These strengths correspond to the numerical values of the underlying routing metrics. This does not go along with *weakest chain link* fact. To concertize this last fact, let us formally denote a three-hops path P which is made of the following one-hop links: (a, b), (b, c), and (c, d), where a, b, c, and d denote four wireless stations, and let Q_P, $Q_{(a,b)}$, $Q_{(b,c)}$, and $Q_{(c,d)}$ denote the corresponding link Quality metric values (e.g., Airtime). By assuming the following two cases:

- Case 1: $Q_{(a,b)}$ = 40, $Q_{(b,c)}$ = 50, $Q_{(c,d)}$ = 60;
  Q_P=$Q_{(a,b)}$+$Q_{(b,c)}$+$Q_{(c,d)}$=150;
- Case 2: $Q_{(a,b)}$ = 100, $Q_{(b,c)}$ = 10, $Q_{(c,d)}$ = 10;
  Q_P=$Q_{(a,b)}$+$Q_{(b,c)}$+$Q_{(c,d)}$=120;

The routing protocol will definitely favor the path in Case 2 as it has a much better Airtime value (smaller value) than in Case 1. However, the path in Case 2 ought to be the worst as it has the worst ”weakest link”.

Indeed, in most routing protocols, individual links are given the same weight regardless of a link being it the weakest or not. In this proposed ping-pong-aware mechanism, we try to give more weight to the weakest links, by presuming that a link which is in the Ping phase is by de facto a ”weakest link” as it is witnessing a sharp decrease in the throughput.

On the other hand, when a link is in the Pong phase, we do further decrease its metric in order to ”attract” further frames, and thus profiting from the exceptional very good link behavior in order to increase the throughput.

We set the step by which we should increase/decrease the Airtime metric to be equal to the step by which the metric did last decrease/increase. Alg. 2 depicts the procedure.

Alg. 2 Ping-pong aware mechanism Algorithm.

compute $Airtime$
$Ping \Leftarrow 0$
$Pong \Leftarrow 0$
if $loss_{current}$ ¿ $loss_{previous}$ ***and*** $rate_{current}$ ¡ $rate_{previous}$ **then**
    $Ping \Leftarrow 1$
elseif $err_{current}$ ¡ $err_{previous}$ ***and*** $rate_{current}$ ¿ $rate_{previous}$ **then**
    $Pong \Leftarrow 1$
if $Ping = 1$ ***or*** $pong = 1$ **then**
    get $Airtime_{previous}$
    $Airtime+ = Airtime - Airtime_{previous}$
    if $Airtime < 0$ **then**
        $Airtime \Leftarrow 0$
return $Airtime$

4.4.3 Experiments

Using ns-3, we implemented the ping-pong-aware algorithm as a module in the HWMP routing protocol. We run the same experiments, as in Section 4.3.3, and we compared the network

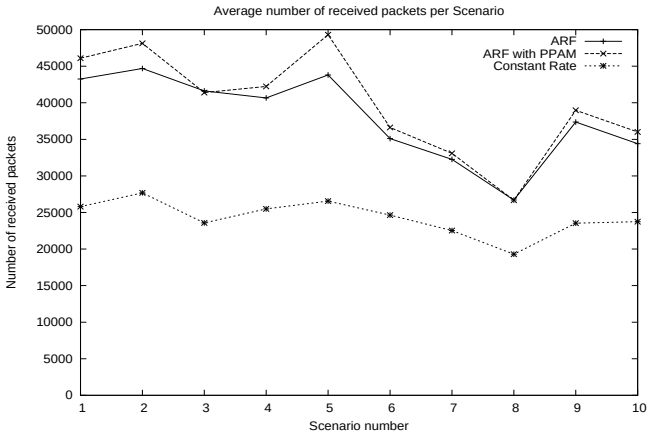

Figure 4.21: Throughput comparison: ARF vs. ARF with PPAM

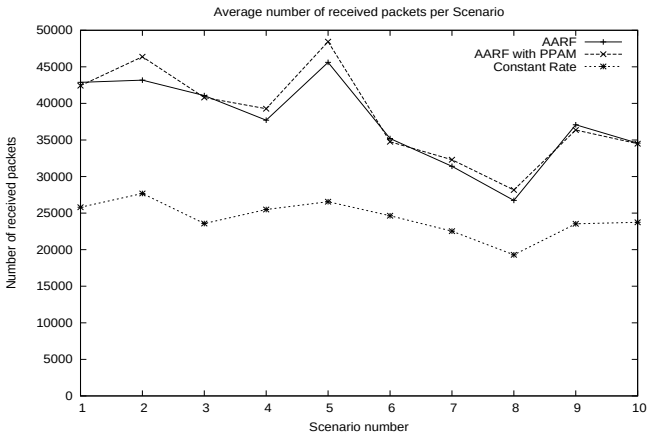

Figure 4.22: Throughput comparison: AARF vs. AARF with PPAM

performance and behavior, under ARF and AARF as rate adaptation algorithms, and with and without using the ping-pong-aware mechanism (PPAM). Figures 4.21 and 4.22 depict the average numbers of received packets at the sink.

From the last figures, we notice that the network performs better under the ping-pong-aware mechanism, for both ARF and AARF. However, the performance is on average a slight one: 5% for ARF and 4% for AARF. However, the algorithm performance reaches up to 40% improvement in some scenarios with specific traffic generation, e.g., for scenario 5 (see Table 4.1) with a 7 Mbps traffic rate.

However, this is still a proof of concept for this new research direction in suggesting and shaping ping-pong-aware mechanisms. By analyzing the network behavior, of individual scenarios, when

using the ping-pong-aware-mechanism, we noticed that this latter indeed adjusts the ping-pong effect for the benefit of improving the network performance. Figures 4.23 and 4.24 depict such an instance.

In the last figures, we plotted the number of received packets per time slot, and we clearly see that the PPAM is adjusting the ping-pong effect by mainly increasing its low boundaries, and increasing its high boundaries. These definitely result in increasing the performance.

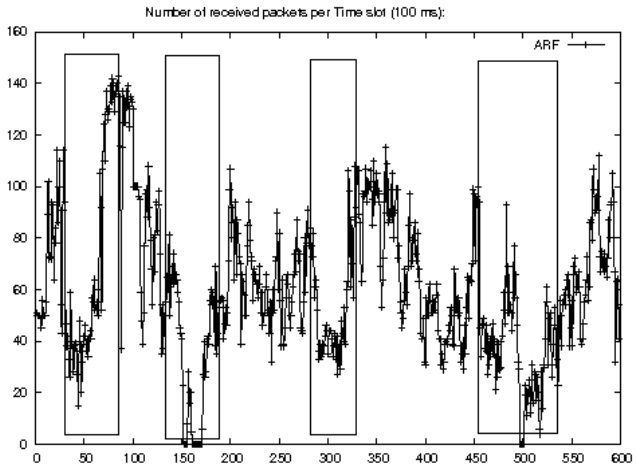

Figure 4.23: Network Ping-pong effect: Scenario 1

4.5 Conclusion

In this chapter we shed further light on the ping-pong effect and we proved the way it is correlated to the underlying rate control schemes. We showed that it is an inherent behavior and not a perilous one as it is referred to in literature. We suggested a new research direction that deems the effect as inherent and thus suggesting the shaping of ping-pong-aware mechanisms that can cope with the effect. We presented a ping-pong-aware mechanism that operates at the level of routing. Regardless of its slight outperformance, it is still a proof of concept for research in such a direction.

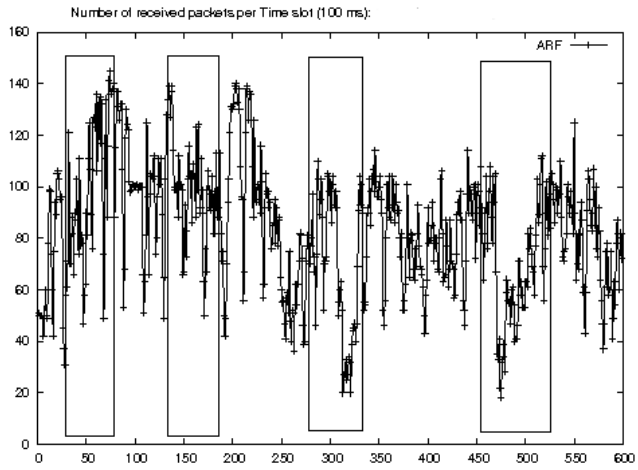

Figure 4.24: Network Ping-pong effect with PPAM : Scenario 1

Chapter 5

An Open-source IEEE 802.11s Indoor Wireless Mesh Network Testbed

In this chapter we highlight the details of the IEEE 802.11s wireless mesh network testbed implementation. We delineate the system design and its components, and we highlight major implementation problem, e.g., clients association, Internetworking, and supporting multiple gateways.

5.1 Motivation

The absence of non-proprietary and open-source testbeds is restricting the proliferation of the research in WMNs by forcing researchers to use simulation tools, e.g., ns-2 [63] and Qualnet [49]. Eventhough simulators are relatively good at approximating most of the real-world networking applications, they are still quite far from doing so in wireless indoor environments, which is the case with IEEE 802.11s WMN technology. In wireless indoor environments, an accurate RF propagation simulation is very crucial, and it largely affects the overall observed network performance. For instance, RF propagation is of paramount importance in determining interferences levels. These latter has proved to be of great impact to the overall performance of multi-hop wireless networks [3, 66]. However, an accurate RF propagation simulation of indoor environments remains quite impossible.

Indeed, and as referred to in Section 1.3, simulators can not capture the complex aspects of RF propagation such as multi-path fading, interferences, path loss, reflection, and diffraction. Furthermore, in indoor environments, of which WMNs are a case, the situation is further worsening because of the unpredictable and mobile nature of obstacles, e.g., furniture, people, walls, etc. This unpredictable and mobile nature of obstacles has a direct and strong impact on the behavior of most RF propagation characteristics, basically reflection, diffraction, and path loss. These facts render

very complex an accurate modeling of the interferences phenomenon. Thus, most simulation tools remain simplistic in the way they are modeling interferences. For instance, ns-2, which is the most widely used tool in academia, uses a simplified version of the physical interference model (the capture threshold model) that accounts for only one interfere at a time [8]. This is definitely not the case in real-world interferences where they can be caused by different simultaneous interferes. Nevertheless, simulators are still very common in research since they remain the sole experimentation alternative when the deployment costs of a real-world testbed are not affordable.

In this context, we present an open-source real-world IEEE 802.11s WMN testbed implementation that would encourage the academia WMN research community to migrate to the use of real-world testbeds, instead of simulators, by deploying the proposed testbed or by using it as a blueprint for deploying their own testbed. The presented testbed is easy-to-deploy as it uses off-the-shelf commodity hardware and software. The linux-based testbed code is open-source, written in C language, and available online [4]. Besides serving the purpose of allowing the WMN researches to easily build their own WMN testbed, the testbed can serve as a building block for a widely-used open-source WMN.

By delving into the testbed implementation subtleties, this dissertation sheds further light into the new IEEE 802.11s standard and addresses major encountered implementation problems such as clients association, Internetworking and supporting multiple gateways. Clients association problem stems from the fact that the original IEEE 802.11 MAC addresses are lost once the corresponding frames leave the WMN towards the Internet. To address this problem, we propose a time-efficient mechanism that maps back to the original IEEE 802.11 MAC addresses and their associating access points, and handles mobile stations hand-offs. With Internetworking, we propose a NAT (Network Address Translation) mechanism that solves the problem of the irrelevance of the IP addresses of the Wi-Fi legacy stations outside WMNs. This irrelevance stems mainly from IEEE 802.11s mesh nodes using MAC addresses for routing. By using MAC addresses for routing, the WMN nodes are not contributing to the non-mesh IP routing protocols, e.g., RIP and

OSPF. And last, when multiple gateways are present, there is always a better route to every operational gateway, and deciding on the best gateway to select requires a way for inspecting the status of every gateway. We propose a mechanism, that uses timestamps for synchronizing between the different gateways.

To ascertain the functionality of the testbed, both TCP and UDP were tested using real-world traffic. Real-world web browsing sessions from Wi-Fi enabled stations, using the testbed as a wireless backhaul, are operational. The current testbed uses 15 nodes and spans two separate locations that are connected through the Internet, see Figure 3.2.

5.2 Implementation

5.2.1 System Design

As mentioned earlier, IEEE 802.11s performs routing at the MAC layer. We opted an easy-to-deploy implementation that can be deployed using off-the-shelf IEEE 802.11 commodity hardware, thus requiring no changes to the IEEE 802.11 driver. In fact, an optimal implementation would require a thorough change of the MAC driver as IEEE 802.11s uses six MAC addresses instead of the four that are used in IEEE 802.11. Still, for our implementation to approximate as much as possible an optimal MAC protocol implementation, we deployed two kernel modules that drop all traffic going to/from the TCP-IP stack. These kernel modules are placed at the PRE-ROUTING and LOCAL-OUT hooks of Netfilter [69] (see Figure 5.1). Besides, by using *Datalink Raw Sockets*, the user-space generated 802.11s frames bypasse the TCP-IP stack. This way, the TCP-IP stack becomes transparent to our user-space implementation and thus ideally approximating a kernel implementation.

The system is made of three modules (see Figure 5.2) that communicate between each other using IPC (Inter Process Communication) shared memories:

1. Routing

2. Data forwarding

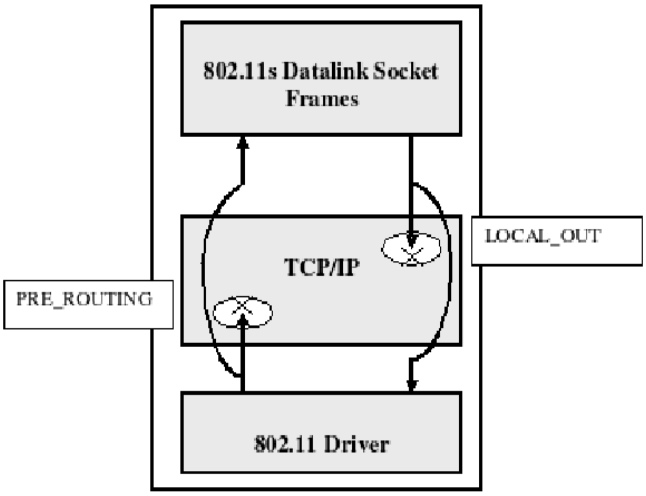

Figure 5.1: IEEE 802.11s testbed; The deployed Netfilter hooks

3. Link quality measurement

Only the link quality measurement module is identical in all WMN nodes, i.e., MAPs, MPs, and MPPs. For the two other modules, their implementation depends on the type of the node. Next sections highlight the implementation details for each module and shows how the implementation differ according to the type of the mesh node.

5.2.2 Data Forwarding Module

The Data forwarding module is the module that relays the received frames to the next hop in the route towards the final destination. To do so, the module extracts the next hop MAC address from the *Proxying table*. This latter is populated by the Routing module, and is implemented as a shared memory structure which is accessed (for writing) by the Routing module and (for reading) by the Data forwarding module. The *Proxying table* contents depends on the type of the mesh node where it is deployed on, see Figure 5.2. We used different names in order to highlight the difference, e.g., *MPP_Proxying_Table* for MPPs, *MP_Forwarding_Table* for MPs, and *MAP_Proxying_Table* for MAPs:

- *MPP_Proxying_Table*: This table is a structure that is made of the set of "*MAP_Proxy*" entries, and a "*visible_MAPs*" counter that counts how many MAPs are visible to every MPP.

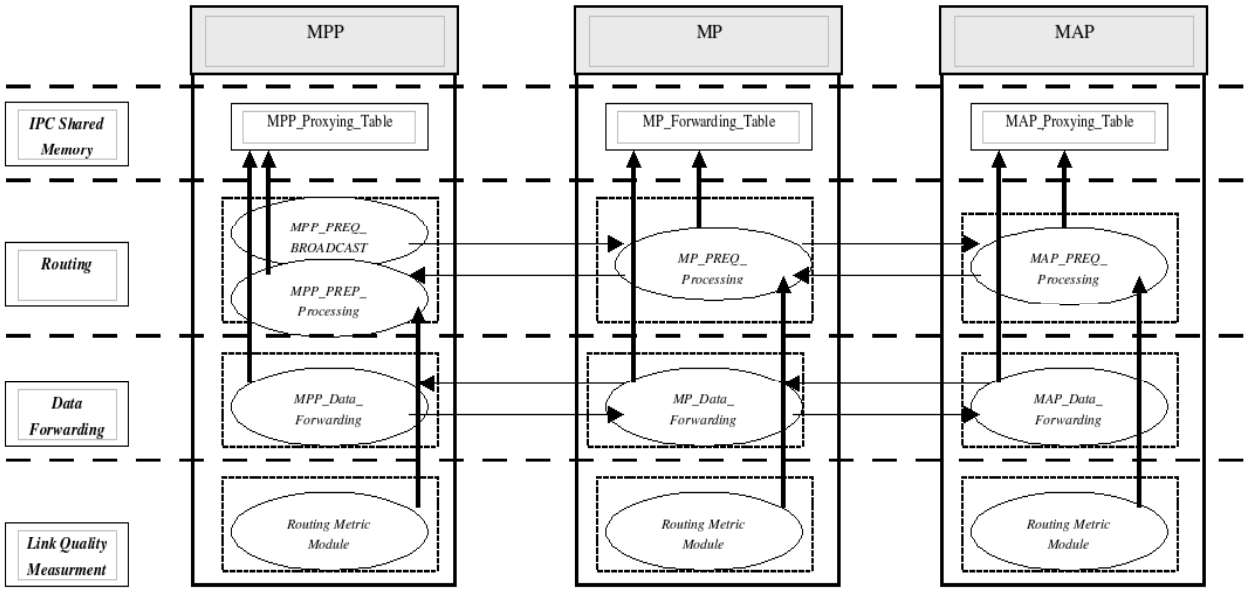

Figure 5.2: IEEE 802.11s testbed; System Design

Since MPPs serve as gateways connecting the WMN to non-mesh networks, the traffic will be either directed from the non-mesh network towards the MAPs, or from the MAPs towards the non-mesh network. Thus, every MPP has to track every visible MAP as well as its corresponding next hop, this latter is communicated by the Routing module. The "*MAP_Proxy*" entry has the following items:

- The MAC address of the MAP.
- The MAC address of the next hop in the route towards the MAP.
- The Sequence number of the last RREP (Route Reply) that was received from the MAP.
- The number of hops in the route towards the MAP.
- The routing metric value for the route towards the MAP.

- *MAP_Proxying_Table*: This table is a structure that is made of the set of "*MPP_Proxy*" entries, a "*visible_MPPs*" counter that counts how many MPPs are visible to every MPP, and a set of boolean "*active_MPP*" numbers that tracks if a MPP is still active (i.e., alive) or not. Every MAP is thus keeping a record for every single MPP, and continuously tracking the

MPP status (i.e., active or not) by comparing the actual time to the last time when the MAP received a PREQ (Path Request) message from the MPP. The "*MPP_Proxy*" entry has the following items:

- The MAC address of the MPP.
- The MAC address of the next hop in the route towards the MPP.
- The Sequence number of the last PREQ (Path Request) that was received from the MPP.
- The timestamp when the last PREQ message was received.
- The number of hops in the route towards the MPP.
- The routing metric value for the route towards the MPP.

The timestamp is used to track if an MPP is alive and as well as to decide on the best MPP in case the WMN is supporting multiple gateways. This latter issue is further covered in Section 5.3.3.

- *MP_Forwarding_Table*: This table is a structure that is made of the set of "*MAP_Proxy*" entries, the set of "*MPP_Proxy*" entries", a "*visible_MAPs*" counter, and a "*visible_MPPs*" counter. Since MPs are forwarding frames towards MAPs and MPPs as well, every MP has to keep track of all visible MAPs and MPPs. The "*MAP_Proxy*" entries and "*MPP_Proxy*" entries have been outlined above. When the MP receives a 802.11s frame originating from a MPP, it will map into the set of "*MAP_proxy*" entries in order to retrieve the MAC address of the next hop towards the destination MAP. On the other hand, the MP will map into the set of "*MPP_proxy*" entries, when it receives a frame originating from a MAP, in order to get the corresponding MPP entry.

The way the Data forwarding module is building the appropriate 802.11s depends on the type of the mesh node as well. Next paragraphs highlights the fact.

5.2.2.1 Data Forwarding in MPPs

MPPs have two network interfaces, an IEEE 802.3 interface and an IEEE 802.11 *ad-hoc* interface, that process two different types of frames:

- Upon reception of an IEEE 802.3 frame, the *MPP_Data_Forwarding* module extracts the destination IP address and MAC address, then it retrieves the MAC address of the MAP where the destination (an IEEE 802.11 legacy station) resides. The MAP MAC address is retrieved from the MPP *Association Table*. This latter basically stores the MAC addresses of the MAPs with which every IEEE 802.11 station is associated (Association tables will be further highlighted in in Section 5.3). Once the destination MAP MAC address is retrieved, the corresponding next hop MAC address is retrieved from the *MPP_Proxying_Table*, and the appropriate IEEE 802.11s header frame is built. This latter contains basically the six MAC addresses that will be used to get the frame, through the IEEE 802.11s WMN network, towards its final destination. The six MAC addresses are built as follows:

    1. Destination MAC address: The next hop MAC address which was retrieved from the proxying table.
    2. Source MAC address: The MAC address of the sender, i.e., the MAC address of the MPP (in this case).
    3. Destination Proxy MAC address: The MAC address of the destination MAP.
    4. Source Proxy MAC address: The MAC address of the MPP. The MPP, in this case, is acting as a source and as a proxy as well.
    5. Final Destination MAC address: The destination MAC address which is retrieved from the local association table.
    6. Originator MAC address: The source MAC address retrieved from the original IEEE 802.3 frame.

The MPP then strips off the IEEE 802.3 frame header and replaces it with the IEEE 802.11s frame header while maintaining intact the payload of the IEEE 802.3 frame. The resulting IEEE 802.11s frame is then forwarded using the MPP wireless ad-hoc interface.

- Upon reception of an IEEE 802.11s frame in its ad-hoc interface, the MPP reads the IEEE 802.11s frame header, extracts the originator MAC address and its IP address, and creates an entry for it in the MPP association table. The entry contains the originator IP address, the originator MAC address, and the associated MAP MAC address. This latter corresponds to the *source Proxy MAC address* (Address 4 in the IEEE 802.11s frame header, see Section 2.3). Afterwards, the MPP strips off the IEEE 802.11s frame header and replaces it with an IEEE 802.3 frame header containing the local MPP MAC address as a source address.

5.2.2.2 Data Forwarding in MPs

In contrast to MPPs, *Data Forwarding* in MPs is quite straightforward as it deals with only one type of frames: IEEE 802.11s frames. However, the MP still has to deal with two types of frames that differ in terms of their final destination which can be either in an IEEE 802.3 network (e.g., frame destined to Internet) or in a IEEE 802.11 network (frame destined to a Wi-Fi station). In other words, the two frames can differ in terms of whether their final destination, within the WMN, is either a MPP or a MAP.

Upon reception of an IEEE 802.11s frame, the MP extracts the destination proxy address, i.e., the MPP or MAP MAC address, consults its *MP_Forwarding_Table* and retrieves the corresponding entry which contains the next hop MAC address. The IEEE 802.11s frame is then updated by changing only two of the six MAC addresses in the IEEE 802.11s frame header. The remaining four are kept intact. The two updated MAC addresses are the destination MAC address and the source MAC address which become, respectively, the retrieved next hop MAC address and the MP MAC address.

5.2.2.3 Data Forwarding in MAPs

MAPs have two network interfaces, a *managed mode* (i.e., Access point) IEEE 802.11 interface and an *ad-hoc mode* IEEE 802.11 interface, that process two different types of frames:

- Upon reception on an IEEE 802.11 frame, from its access point interface, the MAP first reads the source MAC address and then formulates the six MAC addresses in the IEEE 802.11s header frame as follows:

  1. Destination MAC address: The MAC address of the next hop in the route towards the destination MPP. This is retrieved from the MAP_Proxying_Table.
  2. Source MAC address: The MAC address of the local MAP.
  3. Destination Proxy Address: The MAC address of the MPP for which the frame is destined. It is retrieved from the MAP_Proxying_Table, and corresponds to the best MPP in case of supporting multiple gateways.
  4. Source Proxy Address: The MAC address of local MAP.
  5. Final Destination: Unknown at this stage. A NULL address is put in.
  6. Originator: The source MAC address, retrieved from the original IEEE 802.11 frame.

  The MAP then strips off the IEEE 802.11 header, replaces it with the shaped IEEE 802.11s header, and forwards the frame through its ad-hoc IEEE 802.11 interface.

- On the other side, upon reception of a IEEE 802.11s frame in its ad-hoc interface, the IEEE 802.11s header frame is stripped off, while keeping the frame payload intact, and replaced by a broadcast IEEE 802.11 header frame.

5.2.3 Routing Module

The Routing module is the module that implements the HWMP routing protocol. In this implementation, and since we are experimenting with IEEE 802.11s WMNs for last-mile Internet

access, we implemented the proactive part of the HWMP routing protocol, see Section 2.3, since all of the data traffic will be directed towards/from the WMN MPPs (gateways) that connect the WMN to the non-mesh networks, e.g., the Internet. In such a scenario, the tree-based routing is the optimal solution, and it is the case with IEEE 802.11s HWMP which hybrids a reactive and a proactive routing protocols to be used separately or simultaneously depending on the kind of the application.

The Routing module main task is to decide on the best sequence of hops for frame forwarding. To do so, the module gets the routing metrics of the different hops from the Link Quality Measurement module, see Figure 5.2, via IPC shared memories, and disseminate them in the network using the proactive HWMP protocol. As with any multi-hop routing protocol, deciding on the best route is done on a hop-by-hop basis, and at any time when the module decides on a best next hop towards a destination (e.g., MPP or MAP), this latter is communicated to the Data forwarding module by updating the corresponding entry in the *Proxying_Table*.

The Routing module implementation depends also on the type of the mesh node where it is deployed. The next paragraphs highlight these differences.

5.2.3.1 Routing in MPPs

In MPPs, the routing module is made of two sub-modules:

1. *MPP_PREQ_Broadcast* module: It is the main module in the proactive HWMP protocol as it is the one who initiates route discovery by means of periodic broadcasting of PREQ (Path Request) messages. In our implementation, the PREQ message consists of the following items:

    - The MPP MAC address: Only MPPs broadcast PREQ messages.
    - The number of hops, initially set to zero.
    - The PREQ message sequence number: Uniquely identifies a PREQ message.
    - The route metric: Initially set to zero.

- Element ID: This is used to differentiate between PREQ and RREP messages.

2. The *MPP_RREP_Processing* module: This module processes RREP (Route Reply) messages forwarded by intermediate MPs and originating from different MAPs. RREP messages are always originating from MAPs as acknowledgments for the receipt of PREQ messages. RREP messages consists of the following items:

   - The MAC address of the MPP who originated the PREQ message which is acknowledged by this RREP message. This used by intermediate MPs in order to know the final MPP destination.
   - The MAC address of the MAP who is originating the RREP message.
   - The number of hops.
   - The RREP sequence number.
   - The route metric.
   - Element ID.

   Upon reception of a RREP message, the MPP extracts the MAC address of the source MAP, looks up its entry it the *MPP_Proxying_Table*, updates the corresponding routing metric, and sets the next hop in the reverse path towards the MAP who initiated the RREP message. Thus, besides acknowledging PREQ messages, RREP messages set reverse paths towards the source MAPs as well.

5.2.3.2 Routing in MPs

As *intermediate* routers that forward frames towards MPPS and MAPs, the *MP_PREQ_Processing* module processes two types of messages: PREQ and RREP messages.

- Upon reception of a PREQ message, the module checks first the freshness of the message in order to process it as only fresher messages must be considered. First, the module extracts the MAC address of the sender MPP, then uses this address to look up its corresponding entry in

the *MP_Forwarding_Table*. Once the last sequence number is retrieved, the PREQ message is discarded in case it corresponds to a non-fresh message (i.e., of a smaller sequence number). If the PREQ message is a new one then the routing module updates its *reverse* path entry towards the corresponding MPP. Afterwards, the PREQ message is updated by adjusting the routing metric value. The module retrieves the routing metric value, e.g. ETX or Airtime, that corresponds to the link from where the MP received the PREQ message and adds it to the current PREQ routing metric, then rebroadcasts the message. If the PREQ message is of equal freshness as the current sequence number, the PREQ message is then processed only and only if it corresponds to a better route. If that is the case, the PREQ is processed as if it was a fresh PREQ message.

- Upon reception of a RREP message, the MP extracts the MAC address of the sender MAP, looks up its entry in the forwarding table, and then updates the corresponding *reverse* path to the MAP. To forward the received RREP message, the MP looks up the destination MPP MAC address and retrieves its entry from the forwarding table. The retrieved entry contains the MAC address of the next hop in the route towards the destination MPP. Afterwards, the routing metric in the RREP message is updated, and then the message is forwarded towards the next hop.

5.2.3.3 Routing in MAPs

Upon reception of a PREQ message, the *MAP_PREQ_Processing* module retrieves the MAC address of the MPP which issued the PREQ message, then retrieves the entry (from the *MAP_Proxying_Table*) that corresponds to the sender MPP and checks the freshness of the current PREQ message. If the PREQ message is a fresh one, then the corresponding entry in the *MAP_Proxying_Table* is updated and the routing metric value is updated the same way as with MPs.

Still, since the PREQ sequence numbers pertain to *only one* single MPP, the selected route (having the best routing metric and a fresh sequence number) would correspond to the best route leading to one specific MPP, and since there can be other MPPs (i.e., multiple gateways), other better

routes to other MPPs may exist as well. To address this problem, we kept a routing entry, in the *MAP_Proxying_Table*, for every MPP, and we timestamp those entries with the time they were last updated in order to assure the selection of the MPP that has the best route as well as a fresh route (Further details, on the multiple gateways problem, are presented in Section 5.3).

Once the proxying table is updated, a RREP message, that acknowledges the receipt of the MPP PREQ message, is formed and sent back to the selected MPP.

5.2.4 Link Quality Measurement Module

Link Quality Measurement is the module that computes the link quality for all the one-hop links from a node to all its neighbors. Regardless of the type of the mesh node, the implementation is identical. However, the implementation only depends on the kind of the routing metrics which is used by the network, i.e., ETX, Airtime, or ICE. Next paragraphs highlight the implementations details for each of these latter routing metrics.

5.2.4.1 ETX Module

The ETX module consists of two sub-modules: The *ETX_Broadcast* module and the *ETX_Collect* module.

- The *ETX_Broadcast* module periodically broadcasts probe frames (e.g., 1 frame per second) embedding the number of probe frames received in the *reverse* path from every neighboring MP during a certain time period T. *ETX_Broadcast* does not compute the *reverse* delivery ratios, it gets them from the *ETX_Collect* module via IPC (Inter Process Communication) shared memories.

- *ETX_Collect* is the module that computes the ETX values for all links towards neighboring MPs. First, in order to compute the *reverse* delivery ratios, the module counts the number of broadcast probe frames received from every single neighboring MP during the period of time T. Once computed, these ratios are communicated to the *ETX_Broadcast* module in order to

piggyback them in the broadcast probe frames. On the other side of neighboring MPs, when a MP receives a probe frame, its *ETX_Collect* module extracts the corresponding *reverse* delivery ratio that were embedded by the *ETX_Broadcast* module of the sender MP and counts, at the same time, the received probe frames as well in order to infer the corresponding *forward* delivery ratio. In this manner, the *ETX_Collect* module of every node gets both the *forward* and the *reverse* delivery ratios for every link towards a neighboring MP. These ratios are then used to compute the corresponding ETX values (see Equation 1.2.1), which are then communicated to the routing module for route selection. Every mesh node keeps an "*ETX_entry*" for each of its neighbors. The "*ETX_entry*" is made of the following items:

- Neighbor MAC address
- Forward delivery ratio
- Reverse delivery ratio
- ETX value

After a period of time T (e.g., 10 seconds), the "*ETX_entries*" are written to a shared memory which is accessed by the corresponding Routing module. In this manner, every node can get the ETX value of any link towards any of its neighbors.

5.2.4.2 Airtime Measurement

The Airtime measurement was easily implemented by integrating the ETX module. As presented in Section 2.4, Airtime is computed as follows:

$$Airtime = \left(O_{ca} + O_p + \frac{B_t}{r}\right) \frac{1}{1 - e_{fr}} \tag{5.1}$$

According to the equation (5.1), Airtime depends on two basic variables: The frame error rate e_{fr} and the bit rate r. The other constants depend solely on the underlying modulation technique, see Table 5.1.

	802.11a	802.11b/g
O_{ca}	75 μs	335 μs
O_p	110 μs	364 μs

Table 5.1: IEEE 802.11s Airtime metric constants

To compute the frame error rate, we first compute the reverse and the forward *delivery* ratios. Once the reverse and forward *delivery* ratios are computed, they are used to infer the reverse and forward *failure* ratios as follows:

$$f_r = 1 - d_r, \quad f_f = 1 - d_f$$

These latter are used to approximate the frame error rate as follows:

$$e_{fr} = f_r \times f_d$$

The forward and reverse delivery ratios are communicated by the ETX module

The bit rates are extracted using the ”/Proc” system files, and the rate adaptation algorithm was kept to the default, which is SampleRate [60] in Madwifi drivers [64].

5.2.4.3 ICE Measurement

To compute ICE, we used the capability of the Madwifi driver in allowing for the creation of multiple virtual interfaces in the same network interface card. We created two virtual interfaces in every mesh node: One is set to the ”ad-hoc” mode and the other is set to the ”monitor” mode.

In the monitor mode, the network interface card can overhear all ongoing signals, and hence can have access to their characteristics, e.g., RSSI (Received Signal Strength Indicator), transmission rates, frame length. These different values are used to compute the SINR (Signal to Interference Noise Ratios) as well as the Contention Indicator.

The compute the SINR and Contention indicator values for every link, the ICE module implements a hash table that keeps an entry for every WMN node. The hash table has as a hashing key the MAC address of the mesh node, and every hash entry consists of the following items:

- Neighbor MAC address.
- Last RSSI value for the last received frame.

- Average RSSI value: Averaged over a certain period T.
- Average transmission rate.
- Average frame length.
- RSSI sum: sum of the RSSI values for all the frames received during the time period T.
- Frame arrivals: Number of received frames during the averaging period T.

The RSSI sum is used to compute the average RSSI for every node. The average frame length and the average transmission rate are used to compute the average frame transmission time which is used as a time interval variable in the Poisson distribution for computing the probabilities of interference, see Equation 3.2 in Section 3.3.2. Once computed the SINR values are then piggybacked in the HWMP PREQ messages so that other nodes can compute the corresponding asymmetric SINR values, see Section 3.5.1. In fact, the locally computed SINR values correspond only to the reverse SINR values. The forward SINR values are computed at the counter part nodes.

On the other hand, by overhearing all frames, regardless of whether they are destined to the local station or not, every node can compute the contention level by summing the transmission times of all overheard frames over a certain averaging period of time T.

5.3 Implementation Problems

During implementation, we faced three basic problems: 1. clients association, 2. Internetworking, and 3. addressing multiple gateways. The next sections highlight these problems and suggest suitable solutions.

5.3.1 The Clients Association Problem

The clients association problem arises when an IEEE 802.11s frame leaves the WMN via a MPP gateway. In such a scenario, the MAC address of the source IEEE 802.11 legacy station as well as that of the MAP who originated the frame are lost forever. This is due to the IEEE 802.11s

frame headers being stripped off at the level of the gateway, and replaced by the corresponding MAC header, e.g., an IEEE 802.3 header. Thus, when the MPP receives back a frame as a response to one already sent, the MPP cannot forward it to the right destination as it does not know the MAC address of the MAP where the destination resides.

To solve this problem, we propose two alternatives:

1. Forcing every MAP to periodically broadcast the IP and the MAC addresses of its associated legacy IEEE 802.11 stations.

2. Having the MPP extract the needed information from forwarded frames, and store them in a table.

Both alternatives would meet the goal of having the MPP remember the MAP from which it received the concerned frame. However, we opted for the second alternative as the first one would induce more overhead in the network because of the broadcast nature of the approach. Besides, the first approach becomes delicate in the case where the legacy IEEE 802.11 stations are mobile. In this latter case, a MAP may broadcast that a station is associated with it while the station already moved to another MAP coverage area. In other words, the first approach does not handle hand-offs.

In the second approach, the MPP extracts the following data from IEEE 802.11s frames:

- Source IP address
- Source MAC address
- MAP MAC address

The extracted data is then stored in a table. To cope with the look up time of the table, which is of $0(n)$, we implemented the association table as a hash table with a chained list collision resolution strategy. We simplified the hash function to allow for further alleviation in terms of computational speed:

$$Hash(IP_{addr}) = IP_{addr}[3] \,\%\, 256$$

This way, when the MPP receives an incoming packet from the non-mesh network, it uses the destination IP address in order to map to the appropriate entry in the hash table in order to retrieve the corresponding MAC addresses of both the IEEE 802.11 legacy station and the associating MAP. These latter are then used to formulate the appropriate IEEE 802.11s MAC header. In this manner, MPPs can forward the received frames, from the non-mesh network, to the intended MAPs. The MAPs, in turn, will deliver the frames to the right destination using the MAC addresses of the IEEE 802.11 legacy stations which were also retrieved from the association table.

5.3.2 Interconnecting WMNs

With Interconnecting WMNs, another problem arises due to the irrelevance of the IP addresses of the legacy IEEE 802.11 stations outside WMNs, i.e., in external networks. This irrelevance stems from the fact that IEEE 802.11 legacy stations would have IP addresses that are unrecognizable in external networks since these legacy stations are not directly connected to those networks. The legacy stations are connected to external networks via WMN nodes. In IEEE 802.11s, every mesh node is a router. However, since these mesh nodes use MAC addresses for routing, they remain invisible to external IP networks protocols (e.g., RIP and OSPF), thus rendering the IP addresses of the legacy stations unrecognizable as well since these latter are directly connected to the mesh nodes. In fact, external networks can recognize *only* the IP address of the gateway MPP through its non-mesh network interface, e.g., an IEEE 802.3 interface. All other mesh nodes, as well as the client nodes in the IEEE 802.11 networks that use the WMN as an interface, are invisible to external networks. To solve this problem we implemented a NAT (Network Address Translation) mechanism at the level of the WMN gateways (i.e., MPPs). These latter have two interfaces, a WMN interface and an IEEE 802.3 interface.

Accordingly, when a frame is about to leave the WMN towards an external network, e.g., the Internet, the MPP gateway reads the IP address of the originator, i.e., the IEEE 802.11 legacy station, and replaces it by its own IP address. The packet is then sent. However, since there will be

other frames originating from different WMN clients, replacing only the IP address is not sufficient as there is no way for mapping back to the original IP address. To solve this problem, we implemented another hash table, that has as a hash key the combination of the source port, the destination port, and the destination IP address. In this manner, when a connection is created, an entry in the hash table is added, and it is comprised of the following quadruplet: destination IP, original IP, destination port and source port. When a response packet is received back, the gateway maps to the right entry in the hash table using the destination IP, the source port and the destination port. This way the original IP is retrieved. The original IP is then used to map into the association table in order to get its corresponding MAC address as well as the MAC address of the MAP where the original legacy station resides.

Even if the presented NAT mechanism seems logical, this approach did not work at first. We realized that the gateway (MPP), when receiving a response packet, especially in TCP, it replies to the received packet by establishing a TCP connection for itself as well. This results in sending duplicate packets (one from the MPP and one from the legacy station) as a response to a single TCP connection establishment, and thus ending up with two TCP connections. This problem was easily solved when we deployed the Netfilter [69] modules, presented in Section 5.2.1. The Netfilter modules drop all IP packets destined to the gateway and keep intact the packets destined to IEEE 802.11 legacy as the framework uses *Raw Datalink* sockets for frame forwarding. These Raw Datalink sockets read frames at the MAC layer before the Netfilter module drops the packets.

5.3.3 Addressing Multiple Gateways (MPPs)

When processing HWMP PREQ (*Path REQuest*) messages, a MAP has to process only fresher PREQ messages, i.e., the ones that bear a larger or equal sequence number than the last sequence number. However, when a better PREQ message is processed and the corresponding routing entry is updated, a problem arises when the wireless mesh network has multiple MPPs since there will be different sets of PREQ sequence numbers. Each set pertains to *only* one MPP since a PREQ message is uniquely broadcasted by one MPP.

Accordingly, when selecting a better sequence number, this latter would correspond only to a better route toward the specific MPP which originated the concerned PREQ message. The situation becomes quite critical because of the likeliness of having alternate routes to other MPPs with better routing metrics. Besides, since PREQ messages bear sequence numbers that cannot be synchronized as they are sent from different MPPs, comparing sequence numbers issued from different MPPs is nonsense. Furthermore, if we just select the MPP that has the best route, this may not work since the selected route may not refer to a *fresh* route and hence invalid.

To cope with this problem, we adopted a simple approach that consists of keeping a routing entry for every MPP and time-stamping it with the time it was created. The MPPs routing entries consist of the following fields: MPP MAC address, next hop MAC address, routing metric, sequence number, and timestamp. This way, when a PREQ message is received, the MAC address of the sender MPP is retrieved and used to map to the right MPP entry. The sequence number as well as the routing metric of the received PREQ message are then compared to the ones in the appropriate MPP entry. If it corresponds to a better route, the MPP entry is updated with the just received PREQ message and instantly timestamped. However, the updated route still corresponds only to the best route towards a specific MPP. This latter may not correspond to the best gateway to choose for frames forwarding towards external networks.

To select the best MPP we proceeded as follows: Whenever an MPP entry is updated and timestamped as a result of a new processed PREQ message, the timestamps of both the active route and the new candidate route should first conform to the following inequality:

$$t_c - \delta < t_a < t_c \tag{5.2}$$

where t_a and t_c denote, respectively, the timestamps of the active and the candidate routes. δ is a time threshold that assures that the active route is a "still-valid "one. We set the threshold to depend on the MPP broadcast period. The experimental value of the threshold is set to twice the MPP broadcast period in order to account for the case when a PREQ broadcast message has been lost.

If t_a and t_c conform to inequality (5.2), and the candidate route exhibits a better routing metric that the active route, then this latter is updated. By doing this, the MPP with the best route, as well as a "still-valid "route, is guaranteed selection.

5.4 Testbed Deployment Instructions

All the files referred to in this section are available online [4].

For every type of mesh node (i.e., MPP, MAP and MP), we present how to deploy the three different system modules: Data forwarding, routing, and link quality measurement. Data communication between these three modules is implemented via IPC (inter Process Communication) shared memories.

5.4.1 Deploying Mesh Access Points (MAPs)

- *Deploying the Data Forwarding module:* Compile and run the *"MAP_ Data_Forwarding.c"* file.

- *Deploying the Routing module:* The routing module depends on the underlying "Link quality measurement" module as the two modules exchange link quality values through the IPC mechanism. Hence, depending on whether the WMN is using ETX or Airtime as a routing metric, we need to compile and run one of the following files:*"MAP_PREQ _Processing_ETX.c"* or *"MAP_PREQ_ Processing_Airtime.c"*.

- *Deploying the Link Quality Measurement module:* The deployment of this module is the same for all three types of mesh nodes. Hence, it will be separately covered in section 5.4.4

5.4.2 Deploying Mesh Portal Points (MPPs)

- *Deploying the Data Forwarding module:* Compile and run the *"MPP_ Data_Forwarding.c"* file.

- *Deploying the Routing module:* Compile and run the following two files: *"MPP_PREQ_Broadcast.c"* and *"MPP_RREP _Processing.c"*.
- *Deploying the Link Quality Measurement module:* See Section 5.4.4

5.4.3 Deploying Mesh Points (MPs)

- *Deploying the Data Forwarding module:* Compile and run the *"MP_ Data_Forwarding.c"* file.
- *Deploying the Routing module:* As with MAPs, we need to compile one of the following two files depending on whether the WMN is using ETX or Airtime as a routing metric: *"MP_ PREQ_Processing_ETX.c"* or *"MP_PREQ_Processing_Airtime.c"*.
- *Deploying the Link Quality Measurement module:* See Section 5.4.4

5.4.4 Deploying the Link Quality Measurement module

This module is the same for all WMN nodes and it must be deployed in all WMN nodes. The deployment of this module depends on whether the WMN is using ETX or Airtime as a routing metric:

- *Deploying ETX:* Compile and run the following two files: *"ETX_ Broadcast.c"* and *"ETX_Collect.c"*.
- *Deploying Airtime:* Compile and run the "Airtime.c" file. However, we need to compile and run also the *"ETX_Broadcast.c"* and *"ETX_Collect.c"* files as the Airtime module uses the forward and reverse delivery ratios for error frame computation. These latter ratios are computed and communicated by the ETX module.

5.4.5 Deploying the Netfilter Modules

Two Netfilter kernel modules must be placed at the PRE-ROUTING and LOCAL-OUT hooks of Netfilter as explained in Section 5.2.1. To hook the two modules, do *"make"* and *"insmod"* the following two kernel files: *"preModule.c"* and *"localOutModule.c"*.

5.5 Conclusion

In this chapter we presented the implementation details for the deployed IEEE 802.11s wireless mesh network. The implementation was made open-source, transparent, and easy-to-deploy for the research community to reuse the code. Furthermore, the implementation can be used as a blueprint to build one's own testbed, and thus overcoming the shortcomings of using simulations tools.

The implementation source code is maintained online [4]. Comments and suggestions are welcome.

Chapter 6

Conclusion and Future Work

We proposed a novel approach for loss rate estimation that overcomes the shortcomings of both the *broadcast* and the *passive* approach. We proposed also a novel scheme for interference measurement under the *physical interference model*. To assess these new approaches, we shaped a new interference and contention aware metric whose performance was compared against ETX and *Airtime* using an indoor pre-IEEE 802.11s Wireless Mesh Network testbed.

Experiments showed an ICE throughput improvement of 16% on average over ETX and of 16% over *Airtime* in average as well. This relatively good improvement is a proof of the ICE concept which relies on accurate link quality measurement. Yet, we have the strong belief that ICE outperformance could be greater if deployed in large-scale WMNs with more paths redundancy: In these relatively small-sized topologies we used, ETX, Airtime and ICE ought to select the same paths when there are not enough alternatives. However, the fact that ICE is a passive approach, guarantees a minimum outperformance thanks to not using broadcast probes, and thanks to its responsiveness to the time-varying aspect of the wireless link quality. Unlike other passive approaches, ICE does not suffer from the shortcoming of probing idle links as the absence of data is a quality to detect by itself since the absence of traffic, in ICE, is interpreted to represent minimum interferences and minimum contention levels.

We also presented a thorough characterization of the Airtime ping-pong effect, and we provided an analytical study for its causes as well as its behavior. We highlighted its high correlation to the underlying rate control algorithms by using different algorithms, e.g., ARF and AARF, and we showed that it is an inherent behavior stemming from links alternating between the status of being loaded and unloaded. Contrary to the available literature, we proved that the Airtime ping-pong

effect is not a perilous one and that a good characterization of it can help in shaping ping-pong-aware mechanisms that can improve the network performance. The shaping of such mechanisms relies on the crucial prerequisite of detecting when a link undergoes such an effect.

We proposed a ping-pong-aware mechanism that is O(1), decentralized, and that can be easily integrated into the IEEE 802.11s routing protocol. The mechanism proved to have a slight increase in the average overall network throughput. However, the mechanism showed how the ping-pong effect can be adjusted in order to improve network performance. By doing so, we are suggesting a novel research direction which is based on deeming the ping-pong effect as an inherent behavior, and thus inciting the shaping of ping-pong-aware mechanisms that can cope with the effect for the benefit of network performance improvement.

Last and not least, we presented a real-world IEEE 802.11s WMN testbed implementation which is open-source and easy-to-deploy. The implementation is an easy-to-deploy one as it does not involve any IEEE 802.11 MAC driver alteration and thus it can be deployed using off-the-shelf IEEE 802.11 wireless cards. The implementation was made open-source and transparent by making the full source code available online. We identified and highlighted most important implementation problems (e.g., clients association, Internetworking and addressing multiple gateways) and presented workable solutions. Two WMNs, that are located in separate buildings and composed of 11 nodes and 15 nodes respectively, were successfully connected through the Internet. Real-world web browsing sessions, from IEEE 802.11 legacy stations that are connected to the Internet via the deployed WMNs, are tested and operational.

We have the strong belief that the presented implementation will be of great input to the WMN research community, especially in academia, as it provides a way for deploying a testbed and therefore enabling more concise and real-world experimental research.

As a future work, we are intending to migrate and adapt ICE for use with multiple-channels/radios, and continue further research on the ping-pong aware mechanisms by improving the ping-pong effect detection mechanism, and seeking schemes to integrate the ping-pong-aware mechanisms into other levels, e.g., adaptive rate control algorithms.

Bibliography

[1] F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," *Elsevier Science*, 2005.

[2] X. Wang and A. O. Lim, "IEEE 802.11s Wireless Mesh Networks: Framework and Challenges," *Elsevier Science*, 2008.

[3] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, 2000.

[4] A Pre-IEEE 802.11s Wirless Mesh Network Testbed, "http://www.eng.auburn.edu/users/abidmoh/,".

[5] M. Bahr, "Update on the Hybrid Wireless Mesh Protocol of IEEE 802.11s," *IEEE Conference on Mobile Adhoc and Sensor Systems*, 2007.

[6] M. Bahr, "Proposed Routing for IEEE 802.11s WLAN Mesh Networks," *WICON*, 2006.

[7] IEEE802.11, "http://standards.ieee.org/getieee802/download/ 802.11-1999.pdf," 1999.

[8] A. Iyer, C. Rosenbrg and A. Karnik, "What is the Right model for wireless channel interference?," *IEEE Transactions on wireless comm.*, 2009.

[9] R.G. Garroppo, S. Giordano, D. Iacono and L. Tavanti, "Notes on implementing a IEEE 802.11s mesh point," *Elsevier Computer Communications, No. 33*, 2010.

[10] R. Orgier, F. Templin and M. Lewis, "Topology dissemination based on reverse-path forwarding (TBRPF)," *RFC 3684. IETF*, 2004.

[11] S. Lee, B. Bhattacharjee, and S. Banerjee, "Efficient geographic routing in multi-hop wireless networks," *MOBIHOC*, 2005.

[12] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, "On the self-similar nature of ethernet traffic," *IEEE/ACM Transactions on Networking*, 1994.

[13] J. F. Kingman, "Poisson Processes (Oxford Studies in Probability)," *Oxford University Press*, 1993.

[14] The Internet Engineering Task Force (IETF), "http://www.ietf.org/".

[15] Status of Project IEEE 802.11s, "http://www.ieee802.org/11/Reports/tgs_update.htm," *May*, 2010.

[16] D. De Couto, D. Aguayo, J. Bicket and R. Morris, "High-throughput path metric for multi-hop wireless routing," *MOBICOM*, 2003.

[17] R.Draves, J. Padhye and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," *MOBICOM*, 2004.

[18] Aoki, H. et al, "IEEE 802.11s Wireless Mesh Network Technology," *IEEE NTT DoCoMo Technical Journal*, 2006.

[19] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad-hoc wireless networks," *Mobile Computing Kluwer Academic Publishers*, 1996.

[20] Clausen, T. H. and Jacquet P, "Optimized Link State Routing Protocol (OLSR)," *IETF Experimental RFC 3626*, 2003.

[21] "DOC IEEE 802.11-07/0631r0," May 2007.

[22] J. Jun and M. L. Sichitiu, "The Nominal Capacity of Wireless Mesh Networks," *IEEE Wireless Commun.*, 2003.

[23] C. E. Perkins and P. Bhaghwat, "Destination Sequenced Distance Vector Routing," *SIGCOMM*, 1994.

[24] V. D. Park and M. S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," *INFOCOM*, 1997.

[25] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," *WMCSA*, 1999.

[26] A. Adya, P. Bahl, J. Padhye, A. Wolman and Z. Lidong, "A multi-radio unification protocol for IEEE 802.11 wireless networks," *BroadNets. First International Conference on Broadband Networks*, 2004.

[27] D. M. Shila and T. Anjali, "Load aware Traffic Engineering for Mesh Networks," *ICCCN07, Proceedings of 16th International Conference on Computer Communications and Networks*, 2007.

[28] P. Hsiao, A. Hwang, H. Kung and D. Vlah, "Load-Balancing Routing for Wireless Access Networks," *INFOCOM*, 2001.

[29] J. Gao and L. Zhang, "Load Balanced Short Path Routing in Wireless Networks," *INFOCOM*, 2004.

[30] V. Mhatre, F. Baccelli, H. Lundgren, and C. Diot, "Joint MAC-aware routing and load balancing in mesh networks," *ACM CoNEXT07*, 2007.

[31] A. Le, D. Kum, Y. Cho, and C. Toh, "Routing with Load-Balancing in Multi-Radio Wireless Mesh Networks," *IEICE TRANS. COMMUN.*, No. 3, 2009.

[32] M. Gerla, X. Hong, and G. Pei, "Fisheye State Routing Protocol for Ad Hoc Networks," *in Interent Draft, draft-ietf-manet-fsr-03.txt*, No. 3, 2002.

[33] A. Raniwala and T. Chiueh, "Architecture and Algorithms for an 802.11-Based Mulit-Channel Wireless Mesh Network," *INFOCOM*, 2005.

[34] J. So and N. Vaidya, "Multi-Channel MAC for Ad hoc Networks : Handling multi-channel hidden terminals using a single transiever," *MOBIHOC*, 2004.

[35] Y. Liu and E. Knightly, "Opportunistic Fair Scheduling over Multiple Wireless Channels," *INFOCOM*, 2003.

[36] A. H. M. Rad and V. W. Wong, "Joint channel allocation, interface assignment and mac design for multi-channel wireless mesh networks," *INFOCOM*, 2007.

[37] S, Katti, S. Gollakota, and D. Katabi, "Embracing Wireless Interference: Analog Network Coding," *SIGCOMM*, 2007.

[38] C. Fragouli, D. Katabi, A. Markopoulou, M. Medard, and H. Rahul, "Wireless Network Coding: Opportunities and Challenges," *MILCOM*, 2007.

[39] G. Woo, P. Kheradpour, D. Shen, and D. Katabi, "Beyond the Bits: Cooperative Packet Recovery Using Physical Layer Information," *MOBICOM*, 2007.

[40] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft, "XORs in the air: practical wireless network coding," *IEEE/ACM Transactions on Networking*, 2008.

[41] H. Luo, S. Lu and V. Bharghavan, "A new model for packet scheduling in multihop wireless networks," *MOBICOM*, 2000.

[42] R. Ramanathan, J. Redi, C. Santivanez, D. Wiggins, and S. Polit, "Adhoc networking with directional antennas: a complete system solution," *IEEE Journal on Selected Areas in Communications*, 2005.

[43] R. Choudhury, X. Yang, R. Ramanathan, and N. H. Vaidya, "On designing MAC protocols for wireless networks using directional antennas," *IEEE Transactions on Mobile Computing*, 2006.

[44] MIT Roofnet., "http://pdos.csail.mit.edu/roofnet."

[45] IETF, Mobile Ad-hoc Networks (MANET) Working Group., "http://www.ietf.org/html.charters/manet-charter.html."

[46] M. Conti, S. Giordano, "Multihop ad hoc networking: the theory," *IEEE Communications Magazine 45 (4), pp. 78-86*, 2007.

[47] M. Conti, S. Giordano, "Multihop ad hoc networking: the reality," *IEEE Communications Magazine 45 (4), pp. 88-95*, 2007.

[48] Y. Yang and J. Wang, "Design Guidelines for routing metrics in multi-hop wireless networks," *INFOCOM*, 2008.

[49] Scalable Networks., "http://www.scalable-networks.com/products/qualnet/."

[50] Self Organizing Wireless Mesh Networks., "http://research.microsoft.com/en-us/projects/mesh/."

[51] S. Keshav, "A Control-theoretic approach to flow control," *SIGCOMM*, 1991.

[52] Y. Yang, J. Wang and R. Kravets, "Designing Routing Metrics for Mesh Networks," *WiMesh*, 2005.

[53] A. Subramanian, M. Buddhikot and S. Miller, "Interference Aware Routing in Multi-Radio Wireless Mesh Networks," *WiMesh*, 2006.

[54] The ns-3 network simulator. [Online]. Available: "http://www.isi.edu/nsnam/ns/"

[55] A. Kamerman and L. Monteban, "WaveLAN II: A high-performance wireless LAN for the unlicensed band," *Bell Labs Technical Journal*, pp. 118–133, 1997.

[56] IEEE802.11, "http://standards.ieee.org/getieee802/download/ 802.11-1999.pdf," 1999.

[57] M. Genetzakis, V. Siris, "A contention-aware routing metric for multi-rate multi-radio mesh networks," *IEEE SECON*, 2008.

[58] M. Lacage, M. Manshaei, and T. Turletti, "IEEE 802.11 Rate Adaptation: A Practical Approach," in *MSWiM04*, 2004.

[59] ONOE, "http://sourceforge.net/projects/madwifi."

[60] J. C. Bicket, "Bit-rate Selection in Wireless Networks," Master's thesis, Massachusetts Institute of Technology, 2005.

[61] MIT Roofnet., "http://pdos.csail.mit.edu/roofnet."

[62] IEEE802.11, "http://standards.ieee.org/getieee802/download/802.11g-2003.pdf."

[63] ISI, 2006. [Online]. Available: "http://www.isi.edu/nsnam/ns/"

[64] Madwifi. [Online]. Available: "http://madwifi.org/"

[65] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs. [Online]. Available: "http://dast.nlanr.net/Projects/Iperf/"

[66] K. Jain, J. Padhye, V. Padmanabhan , L. Qiu, "Impact on interference on Multi-hop wireless network performance," *MOBICOM*, 2003.

[67] J. Padhye, S. Agarwal , V. Padmanabhan , L. Qiu , A. Rao, and B. Zill, "Estimation of Link Interference in Static Multi-hop Wireless Networks," *In Proc. of Internet Measurement Conference, IMC*, 2005.

[68] S. Biaz and S. Wu, "Rate adaptation algorithms for IEEE 802.11 networks: A survey and comparison," *IEEE Symposium on Computers and Communications*, 2008.

[69] The Netfiler Project. "http://www.netfilter.org/"