MITIGATING THE IMPACT OF PHYSICAL LAYER CAPTURE AND ACK INTERFERENCE IN WIRELESS 802.11 NETWORKS

WANG WEI

B.Eng. & M.Eng., NTU

A THESIS SUBMITTED

FOR THE DEGREE OF PH.D. IN COMPUTER SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

NATIONAL UNIVERSITY OF SINGAPORE

2014

DECLARATION

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Wei Wang
31 July 2014

Acknowledgement

First and foremost, I would like to express my sincere gratitude to my advisor, Dr. Ben Leong, for his guidance and mentoring through my graduate study. With his enthusiasm, his inspiration, and his patience, he helped me to learn how to conduct proper scientific research and also how to live a meaningful life. I simply could not wish for a better advisor.

I am grateful to Dr. Wei Tsang Ooi for his insightful suggestions to my research work and also for his great support during my graduate study. I also learned a lot from Dr. Ooi about teaching when working as his TA for one semester.

I would like to acknowledge my collaborators, Wai Kay Leong and Qiang Wang, for their contributions to the work presented in this thesis. Without their assistance to my research, completing it single-handedly is unimaginable. I would also like to thank the other cheerful people in the lab for their support and friendship: Yin Xu, Aditya Kulkarni, Daryl Seah, Zixiao Wang, Raj Joshi, James Yong, Guoqing Yu, Ali Razeen, Xiangyun Meng, Yan Hao Tan, Eugene Chow, Kartik Muralidharan, Youming Wang, Jian Gong, Yu Chen, Hao Li, Hongyang Li, Pratibha Sundar, and Yi Li.

Finally, I am indebted to my parents for their selfless sacrifice and support since the day I was born. I am blessed with a wonderful wife, Xiaohan Mu, who has provided me enormous support and has been the driving force of my life. Words cannot express my gratitude to her.

Publications

- Wei Wang, Wai Kay Leong, and Ben Leong. "Potential Pitfalls of the Message in Message Mechanism in Modern 802.11 Networks." In *Proceedings of the 9th ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*, Sep. 2014.

- Wei Wang, Ben Leong, and Wei Tsang Ooi. "Mitigating Unfairness due to Physical Layer Capture in Practical 802.11 Mesh Networks." IEEE Transactions on Mobile Computing, to appear.

- Wei Wang, Qiang Wang, Wai Kay Leong, Ben Leong, and Yi Li. "Uncovering a Hidden Wireless Menace: Interference from 802.11x MAC Acknowledgment Frames." In *Proceedings of the 11th IEEE International Conference on Sensing, Communication and Networking*, Jun. 2014.

- Wei Wang, Raj Joshi, Aditya Kulkarni, Wai Kay Leong and Ben Leong. "Feasibility Study of Mobile Phone WiFi Detection in Aerial Search and Rescue Operations." In *Proceedings of the 4th ACM Asia-Pacific Workshop on Systems*, Jul. 2013.

- Guoqing Yu, Wei Wang, James Yong, Ben Leong, and Wei Tsang Ooi. "Adaptive Antenna Adjustment for 3D Urban Wireless Mesh Networks." In *Proceedings of the 10th IEEE International Conference on Sensing, Communication and Networking*, Jun. 2013.

Abstract

As both the deployment density and traffic volume of 802.11 networks are increasing rapidly, the interference among 802.11 devices is expected to become more and more serious, thereby adversely affecting the network performance. In this thesis, we address two major sources of interference that have received little attention in the literature: i) physical layer capture and ii) MAC Acknowledgment (ACK) frames.

Physical layer capture is a common phenomenon in wireless networks where the frames with stronger signal strength can still be decoded in the event of a collision. This is typically helpful, but it can sometimes cause MAC unfairness. Existing solutions that attempt to mitigate MAC unfairness either fail to correctly identify the sender that needs to be throttled or are too aggressive in reducing the sending rate. Our key insight is that the nodes that cause an unfair situation to arise and can act to remedy it are often distinct from the ones that can accurately assess the degree of unfairness. We developed a distributed *CWmin* adjustment protocol, called *FairMesh*, which is the first attempt at decoupling the detection and assessment of unfairness from the remedial action. In FairMesh, the nodes with accurate assessment of unfairness are distributedly elected as *coordinators* to slow down the nodes causing unfairness (called *offenders*) by adjusting their *CWmin*. FairMesh is shown to achieve approximate max-min fairness for arbitrary set of links in 802.11 mesh networks.

We also investigated a special case of physical layer capture for the 802.11n Message In Message (MIM) mechanism, which refers to the capability of a receiver to abandon ongoing reception and shift to receive another frame with a higher signal strength. While MIM is supposed to improve the robustness of receiver against interference, we showed that MIM could be detrimental to the reception of aggregate frames when the interference is stronger. We proposed and evaluated a simple yet effective method to dynamically toggle MIM to achieve near-optimal throughput. The key idea is to monitor the frame receptions and to determine whether MIM should be enabled from the observed collision patterns.

The second source of interference we address in this thesis is the interference due to MAC ACK frames. While most existing works are exclusively focused on the interference due to Data frames, we showed that the interference from the MAC ACK frames

can potentially reduce throughput by several fold. We propose *Minimum Power for ACK (MinPACK)*, a distributed MAC ACK power control protocol that can minimize ACK interference without affecting the original throughput. Starting from the default ACK power, MinPACK gradually reduces ACK power until the level just before the ACK success rate starts decreasing. In addition to mitigating ACK interference, MinPACK is complementary to existing data frames power control algorithms and adapts rapidly to dynamic environments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The IEEE 802.11 standard and its associated products (also known as WiFi technology) have become completely ubiquitous in the past 15 years. Nowadays, almost every mobile electronic device supports WiFi capability, e.g., mobile phones, tablets, digital cameras, or even SD cards. It was recently reported that the WiFi hotspot market would continue to grow at an annual rate of 84% in the next few years [4]. In addition, wireless 802.11 mesh networks, as a complementary configuration to the conventional WLAN, has also been introduced commercially [2].

With such a high demand for WiFi connectivity, we expect the deployment of access points (AP) or mesh nodes will become denser, potentially leading to more inter-flow interference among WiFi devices. Furthermore, interference is likely to become more serious because of the increasing volume of the Internet traffic. For example, Cisco predicted that in 2017 the bandwidth-hungry video traffic will make up 69% of the Internet traffic, more than half of which is carried by WiFi [3]. Therefore, it is important to develop practical and effective solutions to mitigate the inter-flow interference in 802.11 networks.

Interference mitigation has been an active area of research in the literature. However, through a measurement study on a 802.11 testbed, we have identified two aspects that have received thus far received limited attention in the wireless research community. One

Table 1.1: Commercial 802.11 adapters with capture effect.

Chipset	Adapter	Source
N.A.	Lucent WaveLAN-II	[79]
Prism 2	LinkSys WPC11 Compaq WL100 Demarc Tech	[49]
Prism 2.5	SMC 2532-B	[21, 20]
	Senao 2511CD plus Ext2	[43]
Atheros AR5112	Wistron NeWeb CM9	[52]
Atheros AR5212	N.A.	[32]
Atheros AR5213	N.A.	[57]

aspect is *the impact of physical layer capture effect*, whereas the other is *the interference due to MAC Acknowledgment (ACK) frames*.

Physical layer capture effect refers to the phenomenon where, when two 802.11 frames collide at a receiver, the frame with stronger signal strength can still be successfully decoded. This is contrary to the conventional wisdom and assumption that both frames will be lost in the collision. The capture effect is extremely common in commercial 802.11 adapters. Table 1.1 lists several commercial 802.11 adapters that exhibit capture effect. Note that capture effect also exists in other wireless technologies like sensor networks [80] and cellular system [85].

In this thesis, we present the solutions for three problems associated with interference mitigation in wireless 802.11 networks: the first two problems are related to the physical layer capture effect, while the last problem is caused by MAC ACK interference. In particular,

1. we designed and implemented *FairMesh* to mitigate the MAC unfairness arising from physical layer capture effect under the setting of mesh networks [76];

2. we characterized the impact of the Message In Message (MIM) mechanism (a special case of physical layer capture effect) on the reception of Aggregate MPDU (A-MPDU), and developed an adaptive method to dynamically turn on/off the MIM mechanism [77]; and

Figure 1.1: An example of MAC unfairness due to capture effect in a mesh network.

3. we propose a *Minimum Power for ACK (MinPACK)* protocol to improve efficiency by mitigating the interference due to MAC Acknowledgment frames in WLAN [78].

In this thesis, we investigate techniques to improve the MAC performance of 802.11 networks that are constrained by interference. While MAC layer (or link level) problems have been studied in the literature, they are not well solved in practice especially the unfairness problem due to capture effect and the interference problem due to MAC ACK frames. On the other hand, the impact of MIM on A-MPDU is a new problem, and it will become more important as 802.11n hardware become more common.

1.1 Mitigating unfairness due to capture effect

While the capture effect would typically appear to be beneficial since one frame would survive a frame collision, instead of both frames being lost, we found that the capture effect could potentially cause serious MAC unfairness or even complete starvation to the weaker sender. Such MAC unfairness problem is particularly detrimental in mesh networks. For example, in Figure 1.1, the traffic from a strong mesh node Y towards the gateway could completely annihilate the traffic from a weak mesh node X to the gateway as well as all the subsequent mesh nodes that use X as a forwarder to the gateway. It turns out that the unfair situation in Figure 1.1 is quite common in practice, as it has been shown that an RSSI difference of 1 dB is sufficient for capture effect to occur [52]. Most existing works on MAC unfairness in the literature did not consider the impact of capture effect and assumed that mesh nodes X and Y have equal opportunity to transmit to the gateway regardless of their received signal strength.

The challenges of mitigating MAC unfairness in mesh networks lie in the complexity of the problem. First, there are multiple factors that could potentially contribute to MAC

unfairness such as capture effect and unfair topology. For example, unfairness could occur at different places in a mesh network where the topologies are arbitrary and which usually does not have a central management entity. It is also difficult to achieve both fairness and efficiency (i.e., maximum total throughput) at the same time.

We address these challenges with FairMesh, a new practical *CWmin* adjustment protocol to mitigate MAC unfairness in mesh networks. First, we identified three canonical scenarios of MAC unfairness, two of which are due to capture effect and one of which is due to unfair topology. The idea is to simplify a complex unfair situation by breaking it down into the combination of these three canonical scenarios. Second, we observed that the nodes that cause an unfair situation to arise and can act to remedy it are often distinct from the ones that can accurately assess the degree of unfairness. To this end, we decoupled the action of unfairness assessment from the remedial action and proposed a mechanism to distributedly elect a set of nodes (called *coordinators*) to slow down the nodes that cause unfairness (called *offenders*). Third, in view that the classic water-filling algorithm to achieve max-min fairness is only applicable for wired networks, we develop an analogous *CWmin* adjustment algorithm called *water-discharging algorithm* for wireless mesh networks. The idea of our algorithm is to let the coordinators gradually search for the max-min throughput allocation by increasing the offenders' *CWmin* step by step and improving the worst observed throughput in each step. The algorithm does not guarantee max-min fairness but can achieve throughput allocation that is sufficiently close to the optimal.

We show via simulation and with experiments on a 20-node outdoor 802.11 wireless mesh testbed that FairMesh has many desirable properties. First, it is fully distributed and has negligible control overhead. Second, it achieves approximate max-min fairness, and can be modified to support a different notion of fairness (e.g., proportional fairness). Third, it can handle multiple (more than two) competing links and can scale up to mesh networks with tens of nodes. Fourth, it remains efficient under high data rates and high loss rates. Finally, FairMesh interacts well with TCP and maintains good fairness when a

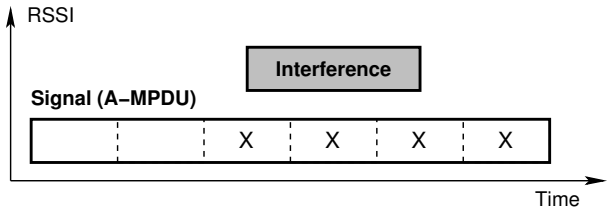


Figure 1.2: Detrimental effect of MIM.

multi-hop flow competes with a single-hop flow.

1.2 Mitigating potential pitfalls of 802.11 MIM mechanism

The Message In Message (MIM) mechanism [52, 68] is a feature of modern wireless receiver which allows a frame with stronger signal to "knock out" the frame that is being received. It is a special case of the general physical layer capture effect, as the MIM mechanism is activated when the stronger frame arrives later than the weaker one. The MIM mechanism also exists in the adapter hardware for sensor networks [80].

While the MIM mechanism has been utilized to improve spatial reuse in 802.11g WLANs [57] and also to reduce packet loss in sensor networks [31], we found that the MIM mechanism could cause performance degradation when the desired signal is an Aggregate MPDU (A-MPDU) and is subject to strong interference. As shown in Figure 1.2, when the MIM mechanism is disabled, the interfering frame will corrupt three frames in the desired A-MPDU (i.e., the third, fourth and fifth frames) and the receiver is still able to decode the last frame. When the MIM mechanism is enabled, however, the receiver switches to receive the interfering frame and thus is unable to decode the last frame in the A-MPDU. Furthermore, the receiver will not reply Block ACK (BA) frame to the A-MPDU sender, which may have to retransmit the whole A-MPDU if it does not support BA Request (BAR) frame.

This potential side effect of the MIM mechanism can be very harmful in modern WLANs, where frame aggregation is widely adopted. For example, an 802.11n sender

can aggregate more than 40 1500-byte frames in a single A-MPDU as the maximum A-MPDU size is 64 KB for 802.11n. The problem would be even worse for the upcoming 802.11ac standard, which employs a maximum A-MPDU size of 1 MB, i.e., equivalent to more than 600 1500-byte frames per A-MPDU. A small but strong interfering frame can potentially destroy a whole A-MPDU, which otherwise could have been partially received if the MIM mechanism is not enabled.

In view of the potential detrimental effects of the MIM mechanism, we develop an algorithm for the receiver to intelligently turn on/off MIM based on ongoing traffic. The basic idea is to let the receiver continuously monitor the frame receptions and assess the effect of the MIM mechanism in a short history. MIM will be enabled only when its benefits outweigh its harmfulness. We also performed a comprehensive set of experiments to characterize the exact impact of turning on/off MIM has on the reception of A-MPDU using commercial 802.11n adapters. We considered different scenarios by varying A-MPDU size, interfering frame air time, and received signal strength difference. We also investigated the scenarios with channel bonding and adjacent-channel interference.

1.3 Mitigating ACK Interference

Due to the shared nature of wireless medium, the system efficiency (or total throughput) of 802.11 networks is fundamentally limited by cross-flow interference. As the deployment density and traffic intensity in 802.11 WLANs are both expected to increase, the negative impact of interference on throughput performance will likely be more serious for 802.11 WLANs in the near future.

Power control has been shown to be an effective solution for interference mitigation in 802.11 WLAN. Existing work on interference mitigation have focused almost exclusively on regulating the transmission power of *MAC Data frames* from access points (APs) [11, 58, 64]. Our key observation is that the MAC Acknowledgment (ACK) frames from clients can also cause serious interference to other flows. Our experiments in a cam-

pus WLAN show that, by reducing the transmission power of MAC ACK frames, i) the throughput of two competing 802.11n UDP flows could be increased by more than 100%; ii) the fairness between two competing TCP flows can be improved; and iii) an 802.11n flow will not be starved by a competing 802.11a flow.

MAC ACK power control is challenging because of several reasons: First, the aggressive and excessive power reduction for the ACK frames could lead to throughput degradation since the Data sender has to retransmit the Data frame if it fails to receive the ACK frame. Second, the level of ACK power adjustment needs to adapt to potential client mobility. Finally, for ease of deployment, the implementation of the ACK power control algorithm should only involve the receiver of Data frames (i.e., the clients) but not the sender of Data frames (i.e., the AP).

To address these challenges, we propose the *Minimum Power for ACK* (MinPACK) protocol that dynamically adjusts the ACK power level of clients in 802.11 WLANs. One key challenge is to accurately and rapidly estimate the success rate of MAC ACK frames. To this end, we developed two estimation methods: a feedback-based method that is accurate but needs to modify AP, and a passive method that does not require AP modification yet is still sufficiently accurate in practice. The rationale of the passive method is that the transmission status of ACK can be approximately inferred by the sequence number of the following Data frame received. Another key challenge is to decide how to adjust the ACK power. We adopted a conservative approach and let the ACK sender gradually reduce the ACK power until the point just before the success rate of ACK starts decreasing.

Through extensive experiments over both our 20-node testbed and campus WLAN, we showed that MinPACK is able to greatly improve the overall throughput by mitigating ACK interference for various scenarios. We also showed that power control of the Data frames alone is insufficient to fully mitigate interference from the ACK frames, and MinPACK can complement existing Data frame power control protocols to achieve better performance. In addition, we also demonstrate that MinPACK is adaptive to moderate client mobility.

1.4 Contributions

The key contribution of this thesis is the thorough investigation of interference caused by the physical layer capture effect as well as the interference due to MAC ACK frames, leading to the development of practical solutions to three common problems identified:

- FairMesh is a new distributed *CWmin* adjustment protocol that is able to comprehensively mitigate MAC unfairness in 802.11 mesh networks. The key mechanism of FairMesh consists of accurate traffic assessment, identification of the degree of unfairness, and *CWmin* adjustment of relevant nodes. The major contributions and insight include: i) improving traffic assessment accuracy through per-neighbor sequence number; ii) decoupling the remedial action from the assessment action; and iii) adjusting the *CWmin* based on a proposed algorithm called the *water-discharging algorithm*. In addition to max-min fairness, the design of FairMesh can also be easily adapted to support proportional fairness.

- To the best of our knowledge, we are the first to discover and investigate the potential pitfalls of the 802.11 MIM mechanism. Our characterization work also reveals that the MIM mechanism could be activated even if the interfering signal is on an adjacent channel. Our main contribution is a simple yet effective method to adaptively turn on/off MIM to achieve near-optimal throughput.

- To the best of our knowledge, MinPACK is the first power control protocol to mitigate the interference due to MAC ACK frames in 802.11 WLAN. Our studies show that MAC ACK interference can limit overall throughput, exacerbate TCP unfairness, and cause 802.11n starvation. The key contributions of MinPACK are two ACK success rate estimation methods (namely feedback and passive methods), and an ACK power adjustment algorithm. The passive estimation method is solely based on the existing sequence number in Data frames (thus no AP modification) and is sufficiently accurate in pracitce. To cater for 802.11n with frame aggrega-

tion, we also enhance the passive method to estimate the success rate of Block ACK frames.

Overall, our work makes very few assumptions, and the majority of our investigations are conducted using commercial WiFi hardware. In this sense, we believe our work has real and immediate impact to the practical deployment and operation of 802.11 networks.

1.5 Thesis organization

The rest of this thesis is organized as follows. In Chapter 2, we present an overview of the existing work on interference mitigation in 802.11 networks. In Chapter 3, we present the design, implementation, and evaluation of FairMesh. In Chapter 4, we investigate the potential pitfalls of the MIM mechanism in 802.11n networks with frame aggregation. In Chapter 5, we present the design, implementation, and evaluation of MinPACK. Finally in Chapter 6, we summarize the work in this thesis and also discuss several interesting future research directions.

Chapter 2

Related Work

In this chapter, we provide a survey of the literature that is relevant to our research work. Specifically, we will discuss existing work on the link characteristics and capture effect of 802.11 adapters, the MAC unfairness problem, the impact of frame aggregation, and the power control protocols for interference mitigation.

2.1 Characteristics of 802.11 Links

The IEEE 802.11 standard has become a popular enabling technology for wireless networks. Unlike its Ethernet counterpart, 802.11 links have a non-negligible packet loss rate that has serious impact to the system performance. Before we embark on discussing the more complicated issues, the first part of our survey is to provide a good understanding on the very basic performance of 802.11 links, particularly the *packet delivery probability*.

2.1.1 Understanding Delivery Probability

Perhaps the earliest comprehensive evaluations of 802.11 link performance was done by Aguayo et al. [10] on the MIT Roofnet [67], which consists of 38 nodes distributed over six square kilometers in the suburb Cambridge, Massachusetts. The adapter is 802.11b running at 2.4GHz ISM band, the antenna is omni-directional and mounted on roof, and

the node is conventional PC with Linux. The quality of links is assessed by measuring the delivery probability of broadcast packets that do not require MAC ACK. Aguayo et al. made several interesting observations, e.g. link quality or delivery probability does not have much correlation with distance and there is a diverse range of loss burstiness among different links. They also observe that the delivery probability of a link has certain correlation with *Signal to Noise Ratio* (SNR) but the correlation is not strong. Such weak correlation between delivery probability and SNR is attributed to the presence of multi-path fading, which induces unpredictable packet loss. As a result, SNR is not recommended as accurate link quality indicator for mesh, at least for Roofnet.

In contrast to the conclusion made in [10], another work on link-level measurement [65] finds that it is *external interference* but not multi-path fading that causes the unpredictable packet loss. The wireless testbed used in [65], called FRACTEL, is similar to Roofnet, i.e. the antennas are mounted on the roof of buildings of a few storeys height and most links have clear Line-of-Sight (LOS). Both FRACTEL and Roofnet use similar 802.11b adapters (with Prism 2.5 chipset) running at 2.4GHz ISM band. One major difference between Roofnet and FRACTEL is that there are two distinct types of environment in FRACTEL—one environment is with external WiFi source in vicinity (e.g. university campus) and the other has no external WiFi interference (e.g. rural village). In the former environment, the correlation between delivery probability and SNR is as weak as in Roofnet, while in the latter environment, it is found that the correlation is strong. In details, the plot between delivery probability and SNR in the latter environment shows a clear threshold of SNR, above which the delivery probability is nearly 100% and below which almost no packet can be delivered. The authors of FRACTEL thus concluded that external interference is the major factor that weakens the correlation between delivery probability and SNR. In addition, they also re-examined the data collected in Roofnet and investigate why the impact of external interference is neglected in [10].

Although multi-path fading is downplayed by the authors of FRACTEL, it does not mean that its impact can be completely ignored in other types of channel conditions, con-

sidering that the multi-path fading in FRACTEL is mild. In a more recent work [37], Halperin et al. investigate the impact of frequency-selective fading (as a result of multi-path fading) on the link-level performance of 802.11n. Their two testbeds under investigation are both indoor, whereby inducing much more severe frequency-selective fading than the outdoor testbeds in Roofnet and FRACTEL. In addition, there is no external WiFi interference since the testbeds run at 5GHz, which is relatively clean. Unlike 802.11b, 802.11n employs Orthogonal Frequency Division Multiplexing (OFDM) scheme that utilizes a large number of orthogonal sub-carriers to transport data simultaneously. Depending on the channel condition from sender to receiver, different sub-carriers would incur different degrees of fading. Consequently, even though two links have the same SNR, they would produce diverse performance of delivery probability. In other words, when frequency-selective fading is severe, there does not exist a clear SNR threshold for predicting delivery probability.

With a good understanding on the influencing factors of the delivery probability of 802.11 links, we are able to predict the delivery probability based on those factors. An accurate prediction on delivery probability is crucial to the *adjustment of data rate* of a link and also to the *selection of routes* in mesh networks.

As we have seen in the previous section, SNR is a fundamental factor in determining delivery probability, but the SNR-based prediction could be greatly perturbed by other factors such as external interference and frequency-selective fading, which are generally difficult to model. In this sense, one of the simplest ways to predict delivery probability, as adopted by the Roofnet team, is to statistically measure it using artificial packets. In details, the Roofnet team uses two types of artificial packets with different packet sizes: large one (1500-byte) for emulating normal Data frame whereas small one (60-byte) for MAC ACK frame. The artificial packets are periodically broadcasted by each node, and by recording the reception status of these packets, nodes are able to estimate a link's delivery probability of both Data and ACK frames. Based on estimated delivery probability, the Roofnet team proposes a routing metric, called Expected Transmission Count

(ETX) [26], for selecting high throughput route. ETX is later enhanced to Expected Transmission Time (ETT) [18] by taking data rate into consideration. ETT is further revised to Weighted Cumulative Expected Transmission Time (WCETT) [28] for multi-radio mesh.

In spite of its simplicity and universality, the statistical estimate method using artificial packets has an obvious drawback which is the communication overhead. In order to accurately and promptly estimate delivery probability, the artificial packets have to be sent very frequently, thereby inducing excessive communication overhead. Another drawback is its inability to distinguish packet loss due to hidden-node collision from that due to poor link quality [66]. Take the application of data rate adaptation for example. If a link starts incurring low delivery probability due to collision, sender would presume the link quality has degraded and thus switch to lower data rate. As a result, the air time is elongated, thereby exacerbating collision. Besides, the study in [25] has demonstrated that, with multiple flows running in a mesh, ETT starts showing meaningless value and fails to capture the true delivery probability. Although one may say that ETT (or ETX) is able to reflect the packet loss due to collision to some extent, our argument is that it is not specifically designed to do so.

In the work of the FRACTEL testbed [65], the authors recommend to use SNR to directly predict delivery probability since FRACTEL is nearly free of multi-path fading and has little interference in rural environment. In the 802.11n indoor testbed in [37], frequency-selective fading is the source of perturbation on the correlation between delivery probability and SNR. Since different sub-carriers suffer from different fading, the authors of [37] exploit the individual SNR value of each sub-carrier and propose a model to calculate *Effective SNR* (ESNR), which can be used to directly predict delivery probability under different channel conditions of fading. Their model also takes into consideration the Multiple Input Multiple Output (MIMO) mechanism used by 802.11n. All inputs to their model are obtained from the Channel Status Information (CSI) as reported by their Intel 802.11n card.

2.1.2 Physical Layer Capture Effect

In this section, we discuss the existing research work on physical layer capture effect of 802.11 radio, which states that a stronger signal (in terms of SNR or RSSI) at receiver is able to survive collision. Capture effect also exists in many other types of radio, e.g. the Chipcon CC1000 transceiver [80] and CC2420 transceiver [55] used in wireless sensor networks, the receivers in FM radio [53], as well as the hardwares used in cellular systems [85]. Capture effect contradicts the commonly held belief that both packets get lost in collision [48].

Perhaps the earliest work on investigating capture effect in real 802.11 radio is the one in [79], with Lucent WaveLAN-II adapters used. The network topology is that two hidden senders simultaneously transmit TCP traffic to the same receiver. By varying the SNR difference of the two senders at the common receiver, their experiments demonstrate that the stronger sender is dominant whereas the weaker sender gets starved. In addition, the effect of RTS/CTS is examined. It is found that RTS/CTS could not prevent TCP starvation. Although not explicitly stated in [79], the reason could be because capture effect takes effect for RTS frames as well.

The work in [49] investigates how the impact of capture effect is perceived at transport layer (using UDP or TCP) and how such impact is exacerbated by some mechanisms at MAC layer such as BEB. Another evaluation work on the unfairness problem arising from capture effect can be found in [32], which is limited to UDP traffic. An important contribution of [32] is that some remedies are proposed to mitigate the unfairness due to capture effect, including changing transmission power, adjusting MAC retry limit, and tuning 802.11e QoS parameters. However, the remedies are not adaptive to arbitrary traffic conditions and topologies. TCP starvation due to physical layer capture effect is also partly investigated in [21].

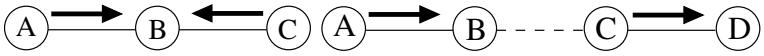
Since the impact of capture effect is directly determined by the signal strength difference (e.g., SNR difference) between desirable signal and interfering signal, it is necessary to quantify the correlation between such SNR difference and the delivery probability of

the stronger signal. The comprehensive measurement study in [43] evaluates such correlation for 802.11b adapters with Prism 2.5 chipset under different conditions, which include varying data rate and varying delay from interested signal to interfering signal. The results show that, at higher data rate, the occurrence of capture effect requires larger SNR difference and longer delay between desired signal and interfering signal.

Another dedicated study on capture effect is the work presented in [52], which focuses on 802.11a adapters with Atheros 5112 chipset. It studies not only the case where the desirable signal arrives earlier than the interfering signal (as in [43]) but also the case where desired signal comes later. The capability that stronger signal is able to "knock out" a weak signal that is being received is called *Message in Message (MIM) mechanism* [68].

The results in [52] show that at least 10 dB SNR difference is required in order for MIM to take effect. On the other hand, if desired signal comes earlier, a smaller SNR difference is enough for it to survive from collision. For example, at 6 Mbps, any positive SNR difference would make the desired signal immune to collision. The work in [52] also addresses the situation where interfering signal arrives earlier than desired signal but somehow does not trigger the receiver to receive it. In this case, for 6 Mbps data rate, about 4 dB to 5 dB SNR difference is needed for the late desired signal to be correctly received. Notice that the result observed in [52] is very similar to that in our 802.11 adapters.

Unlike above-mentioned works that treat capture effect as detrimental [79, 49, 32, 21], the work in [57] capitalizes capture effect to improve spatial concurrency in infrastructure WLAN. It is based on the observation that, in order to be correctly decoded, the desired signal requires smaller SNR difference (4 dB) if it precedes interfering signal than the other way round (10 dB). When applied to infrastructure WLAN, overall throughput from APs to their clients can be improved if the order of APs' transmissions is carefully selected. Their proposed scheduling protocol, called *Shuffle*, is implemented in a controller that is connected to various APs through Ethernet. Measurement results on a simple 3-AP testbed shows that both aggregate throughput and fairness are improved with



(a) Fair topology (b) Unfair topology

Figure 2.1: The fair and unfair topologies. Solid arrow refers to actual transmission, whereas dashed line for overheard transmission.

Shuffle as compared with both 802.11 and TDMA schemes.

In summary, we make two observations about physical layer capture effect: 1) it can cause serious MAC unfairness and even starvation in 802.11 networks; 2) if carefully managed, capture effect can be potentially utilized to improve the spatial diversity in 802.11 networks.

2.2 Unfairness of 802.11 MAC

The IEEE 802.11 standard was originally designed for infrastructure-based WLAN, where multiple clients compete for a single AP. As shown in Figure 2.1(a) which represents a typical WLAN, two clients (*A* and *C*) are hidden nodes with each other and compete for a single AP (*B*). RTS/CTS exchange is able to mitigate potential hidden-node collision between *A* and *C* at *B*. What is more important, *A* and *C* have the same medium access opportunity to reserve the channel if there is no capture effect. The measurement work in [33] has shown that similar topologies like the one in 2.1(a) exhibit long-term fairness.

Things get much complicated in mesh networks, where the 802.11 standard is no longer fair in some problematic topologies. A representative problematic topology is shown in Figure 2.1(b). *A*'s transmissions to *B* might collide with the radio signal from *C*, whose intended receiver is *D*. Note that *C*'s transmissions do not incur any collision at *D*, and thus always use the smallest MAC contention window. As a result, there is little chance for *A* to correctly "insert" its data packets (or even RTS packet) into the small inter-packet gap of *C*'s transmission. In other words, *A* is inferior to *C* in terms of medium access opportunity. In this section, we are going to review some representative works on the unfairness of IEEE 802.11 MAC.

2.2.1 MACA, MACAW and Representative Topologies

MACA. The above-mentioned intrinsic unfairness of IEEE 802.11 MAC was first reported in the work of MACAW [16], which is based on the design of MACA (Multiple Access Collision Avoidance) protocol [44]. The main contribution of MACA is the idea of RTS/CTS exchange, which is later adopted in IEEE 802.11. One difference is that MACA employs only 3-way handshake (RTS/CTS/DATA) rather than the 4-way handshake (RTS/CTS/DATA/ACK) used in IEEE 802.11, but the omission of ACK in MACA makes it unsuitable in fading channel. Another salient feature of MACA is the omission of carrier sense. Without carrier sense and with RTS/CTS, the classic exposed node problem could be partially solved. For example in Figure 2.1(b), assume B is sending to A when C attempts to send to D. In MACA, C only need wait for a short duration for B to receive A's CTS, and then C can start sending anything to D [1].

MACAW. MACAW is modified from MACA and incorporates a few new designs. Firstly, in view of the lossy nature of wireless medium, MACAW adds ACK as the final step into MACA's RTS/CTS/DATA sequence. The resulting 4-way handshake is later adopted in IEEE 802.11 MAC. Secondly, MACA considers per-stream fairness. A node keeps separate transmission queues for each stream and runs back-off for each queue independently.

Thirdly, MACAW argues that the conventional Binary Exponential Backoff (BEB) is unfair because the BEB always favors the last successful transmission. To solve the unfairness of BEB, MACAW proposes to let each packet piggyback its current back-off value, and any other node who gets the packet would set to the same back-off value as in the packet. Another problem of BEB is the rapid change and large variation of back-off value. To mitigate the oscillation of back-off value, MACAW revises the back-off mechanism to Multiplicative Increase Linear Decrease (MILD). When packet loss is detected, back-off value increases by a multiplicative factor of 1.5; when a packet is successfully sent, back-off value decreases by one. Simulation results show that the two

[1] Of course, after C sends RTS and expects CTS from D, D's CTS might be collide with B's radio signal.

revisions of back-off mechanism can both achieve better fairness than BEB in single-AP scenario.

Fourthly, MACAW takes special care of a few topologies where unfairness easily occurs. One problematic topology is similar to Figure 2.1(b), in which *A* sends to *B* and *D* sends to *C*. Suppose *B* accepts *A*'s RTS and replies with CTS, which prohibits *C* from sending anything. In this case, *C* would ignore the RTS from *D*, unless *D* is able to insert its RTS into the small inter-packet gap of *A*. To solve such unfairness to *D*, *C* would contend for *D* by sending Request-for-Request-to-Send (RRTS) after *A*'s transmission finishes. Upon receiving/overhearing RRTS, *D* immediately sends RTS to *C*, and *B* defers for a short while so as to leave chance for *D* to complete its RTS/CTS exchange. Another problematic scenario is, as shown in Figure 2.1(b), when *B* and *C* want to send to *A* and *D*, respectively. Suppose *B* sends RTS first but somehow fails to receive CTS from *A*. Then *C* would be deferred unnecessarily. To save air time for this case, *B* would send a Data-Sending (DS) packet after receiving *A*'s CTS. *C* is deferred only after receiving DS. In fact, here DS has similar usage to carrier sense, but it does not require carrier sense hardware. The last problematic scenario is the above-mentioned flow-pair in 2.1(b). Unfortunately, no solution was provided in MACAW. Solving the unfairness of IEEE 802.11 MAC in this scenario is one of our critical tasks in the future.

12 Representative Topologies. We have seen a number of representative topologies as mentioned in MACAW, and it is possible to generalize them further. Garetto et al. [33] enumerate and analyze 12 representative topologies that involve two *single-hop flows*, *Aa* and *Bb*, as shown in Figure 2.2. These representative topologies can be used to constitute more complex topologies in mesh, and thus they are worth investigating. Simple UDP traffic is used to understand the interaction between the two competing flows. According to their fairness performance obtained from `ns-2` simulation, these topologies can be grouped into three categories. Topology 1 to Topology 7 belong to a category where the two competing flows almost have equal throughput. The reason of such fair performance is because the two senders (node *A* and node *B*) can sense each other and thus are able to

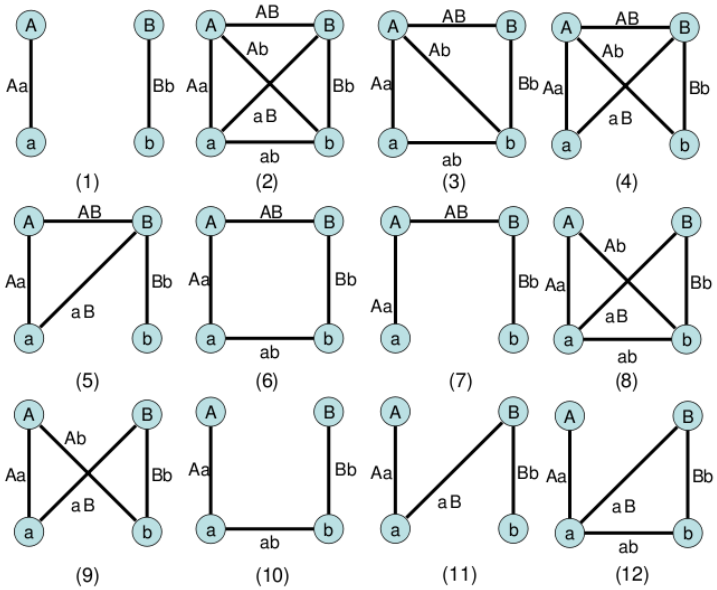


Figure 2.2: 12 representative topologies in [33].

coordinate their transmissions very well using CSMA.

The second category includes Topology 8, Topology 9 and Topology 10. These topologies show severe throughput unfairness in short-term (within one second) but enjoy fairness in long-term. The culprit of short-term unfairness is the exponential backoff mechanism, which exacerbates the inferiority of the flow that loses in contention. The long-term fairness is attributed to the fact that, on average, the two senders have equal chance to win over each other.

Topology 11 and Topology 12 belong to the third category, and both topologies show severe long-term unfairness (flow *Aa* is starved). The starvation is owing to the different situations faced by the two receivers (node *a* and node *b*). Node *a* cannot correctly decode the packet from node *A* because of the radio from node *B*, whereas node *b* is able to enjoy nearly error-free transmission from node *B*. Since almost no error occurs on flow *Bb*, node *B* does not experience exponential back-off, thereby further increasing the chance of hidden-node collision at node *a*. Topology 11 is exactly the same one in Figure 2.1(b).

Similar performance of above 12 topologies is observed when RTS/CTS is enabled, except that the aggregate throughput is a bit smaller owing to RTS/CTS overhead. Although not explicitly stated in [33], the ineffectiveness of RTS/CTS in avoiding short-

term starvation (such as in Topology 8) is because the default value of *CWmin* is too small, thereby inducing collision between RTS packets. RTS/CTS cannot prevent long-term unfairness (such as in Topology 11), since the two senders intrinsically do not have equal opportunity of successful transmission.

2.2.2 Unfairness Detection and Reaction

To solve the unfairness arising from the unfair topologies in general wireless multi-hop networks, we have to first detect the existence of the unfairness, and then take action to mitigate it. In this section, we are going to review two works that follow this principle. They use different unfairness detection techniques and their proposed reactions to unfairness are not the same either: one using contention window adjustment, and the other using sending rate adjustment. We will compare both of them with FairMesh later in Chapter 3.

Probably one of the earliest works in this category is the measurement-based scheme proposed in [14]. In details, each node keeps overhearing all packets transmitted from neighboring nodes (including RTS/CTS exchange). Based on the timing information specified in the header of received packets, a node is able to estimate how much air time its neighbors have consumed. Each node treats all its neighbors as a whole and keeps a single value of their overall consumed air time, as denoted by T_o. The air time consumed by local node itself can be easily obtained, as denoted by T_i. Together with two predefined shares of air time, ϕ_i for itself and ϕ_o for others, T_i and T_o can be used to compute a fairness index as $FI = max(T_i/\phi_i, T_o/\phi_o)/min(T_i/\phi_i, T_o/\phi_o)$. Then a node compares its latest fairness index with two threshold values, C and $1/C$, and generates three decisions on adjusting contention window: doubling, no change, and halving.

Using OPNET simulations, it is demonstrated that the scheme is effective in improving channel access fairness in some simple and representative topologies, including the one in Figure 2.1(b). The other advantage of the scheme is its simplicity, and it can potentially be implemented using available hardware. However, there are several drawbacks that limit its operation in practice. Firstly, it assumes that the desired air time shares, ϕ_i

for itself and ϕ_o for others, are already available and do not change. In fact, the air time share in mesh is not only unpredictable but also highly dynamic. In an extreme case when all nodes stop sending except just one node, that node would guess its transmission is too aggressive and thus keeps doubling its congestion window unnecessarily. Secondly, the scheme does not distinguish different neighbors but simply treats them as a whole. This treatment is too coarse, because different neighbors have different traffic demand. Thirdly, the traffic model used in simulation follows Poisson distribution, which cannot capture the aggressiveness of node with high channel access opportunity. Besides, the simulated topologies are too simple to evaluate the proposed scheme in a comprehensive way. Another work that studies similar scheme can be found in [30].

A more recent work is the Additive Increase Synchronized Multiplicative Decrease (AISD) scheme proposed in [42]. It is well known that TCP's Additive Increase Multiplicative Decrease (AIMD) rate control mechanism is able to converge to both fairness and efficiency of competing TCP flows. Basically, the purpose of AISD is to apply similar mechanism of AIMD to improve the fairness of IEEE 802.11 MAC. The authors argue that conventional AIMD achieves fairness only when competing flows do multiplicative decrease in a synchronized manner, and thus AIMD cannot be directly applied in mesh.

The details of AISD scheme as follows. Each node keeps a rate of packets that MAC layer can take from upper layer (denoted by r). r increases linearly with time, i.e. additive increase. A node also keeps measuring its transmission queue length q. If q is larger than some threshold δ_q, the node would infer that it does not get fair share of air time. In other words, unfairness is detected according to a node's own states, rather than by overhearing other nodes' transmissions as in [14]. To gain more chance of channel access, the node reduces its own contention window to the minimum value and tries to send out many backlogged packets in a batch. The purpose of such massive sending is twofold: reducing its own queue size, and jamming other nodes so that they would react accordingly. When receiving jamming signal, a node could hardly send and thus join the jamming eventually due to large q. After jamming signal finishes, a node reduces r by some percentage, i.e.

synchronized multiplicative decrease. In this way, AISD makes AIMD work at IEEE 802.11 MAC layer since competing flows now decrease rate in a synchronized manner.

Thanks to the well-known efficacy of AIMD mechanism, AISD the scheme in [42] shows better fairness when applied to some simple problematic topologies, according to `ns-2` simulation results. However, similar to the drawback of [14], the simulated topologies in [42] is primitive. In addition, the simulation measurement does not evaluate the impact of the detrimental jamming signal, which regularly occurs in the proposed scheme. When the number of nodes is large, the impact of jamming signal might be devastating because a jamming node could cause another node to send jamming signal.

While the work discussed above are able to mitigate the MAC unfairness problem to some extent for certain scenarios, these techniques do not work for MAC unfairness arising from physical layer capture effect, which is common in modern 802.11 hardware.

2.3 Impact of Frame Aggregation

Frame aggregation was first developed in the IEEE 802.11e standard and have been revised in the IEEE 802.11n standard. One type of frame aggregation is Aggregate MPDU (A-MPDU). An A-MPDU includes a single physical header and one or multiple MAC Data frames, each of which is protected by 4-byte CRC checksum. When an 802.11n receiver receives an A-MPDU, it replies with a Block ACK (BA) frame informing the A-MPDU sender which Data frames have been lost if any.

A number of studies showed that frame aggregation is able to greatly reduce transmission overhead and improve throughput for 802.11n links. The work by Skordoulis et al. [72] was one of the earliest attempts to quantify the impact of frame aggregation in terms of transmission overhead reduction for 802.11n links. Through OPNET simulations, the authors showed that A-MPDU is able to achieve throughput that is approximately 4.5 times higher than the no aggregation method as a result of overhead reduction. They found that the use of Aggregate MSDU (A-MSDU) could further improve

the throughput of A-MPDU albeit not significantly. Using analytic model, Ginzburg and Kesselman [34] showed that A-MPDU aggregation achieves channel utilization of 95% in the ideal case as compared with the 33% channel utilization when there is no frame aggregation. They also observed that, when link data rate and loss rate are high, A-MPDU aggregation outperforms A-MSDU aggregation.

Paul et al. [63] reported similar throughput gain due to A-MPDU using real experiments in an outdoor environment. They observed that the two new mechanisms proposed in 802.11n standard, channel bonding and short guard interval, would not achieve significant throughput improvement unless frame aggregation is used. A similar measurement work by Kriara et al. [51] showed that frame aggregation is always beneficial when the link loss rate is small. The authors also observed that frame aggregation could lead to MAC unfairness as the sender with excessive aggregation will occupy the channel longer than other senders. We will address this issue in Chapter 3.

The upcoming 802.11ac standard employs a frame aggregation scheme that is more aggressive than that of 802.11n (maximum A-MPDU size of 1 MB for 802.11ac vs. 64 KB for 802.11n). Ong et al. [62] analytically evaluated the performance gain due to frame aggregation for 802.11ac links and compared it with 802.11n. They found that, with frame aggregation, an 802.11ac link with the configuration of 80 MHz bandwidth and single spatial stream achieves 28% higher throughput than 802.11n with the configuration of 40 MHz bandwidth and two spatial streams. Similar like the result in [72], it was shown that 802.11ac links would have the maximum MAC efficiency when A-MPDU and A-MSDU are both enabled. Throughput gain due to frame aggregation for IEEE 802.11ad at 60 GHz can be found in [84].

While the above-mentioned works mainly focused on the link-level performance of frame aggregation, Gubner and Lindemann [35] investigated the impact of frame aggregation on video streaming performance over multi-hop mesh networks using commercial 802.11n adapters (Atheros AR9223 chipset). They found that A-MPDU aggregation greatly improves both the delay and quality of the streamed video. For example, with

A-MPDU aggregation, a Full-HD video can be streamed over a 6-hop path with little degradation to the video quality in terms of PSNR, and the average end-to-end delay is smaller than 100 ms. They showed that both the delay and quality of the streamed video will get worse as the limit on the aggregation size becomes smaller. The reason is twofold: the link-level overhead is reduced (as discussed earlier), and the level of channel contention becomes mild as a result of fewer attempts to transmit.

Camp-Mur et al. [22] studied the interaction between frame aggregation and two 802.11 power saving protocols: 802.11 Power Saving Mode (PSM) and 802.11e Unscheduled Automatic Power Save Delivery (U-APSD). PSM is the design for power saving in the original 802.11 standard, where the client periodically wakes up every Beacon interval to receive frames from AP. U-APSD was developed in 802.11e to support delay sensitive application, by allowing the client to wake up itself to trigger the AP. The authors found that while U-APSD achieves shorter delay than PSM when A-MPDU aggregation is not used, PSM has much higher throughput than U-APSD when A-MPDU aggregation is enabled. The reason is that both the AP and client wait longer in PSM to buffer frames, thereby creating more aggregation opportunities.

The work presented in this section suggest that A-MPDU is an important mechanism for improving throughput. However, the serious problem of throughput reduction when A-MPDU interacts with the MIM mechanism has to the best of our knowledge, not been reported and investigated in the literature.

2.4 Methods for Interference Mitigation

One of the most effective methods for interference mitigation in wireless networks is transmission power control. Specifically, adjusting transmission power affects not only the reception at the desired receiver but also the interference level to other receivers. Power control also helps conserve energy, but it is not a concern in WLAN or mesh. In this section, we will focus on a number of practical power control works that were developed

to mitigate interference in 802.11 networks by adjusting the transmission power of MAC Data frames. We will also cover several other types of works proposed for interference mitigation.

2.4.1 Power Control of Data Frames

Broustis et al. [19] performed a comprehensive measurement study on how power control affects the performance of two competing links. They broadly classified link pairs into three categories: overlapping, hidden-terminal, and potential disjoint. When the two senders can sense each other (i.e. the overlapping category), power control has no positive impact because the senders coordinate nicely using carrier sense. Power control was found to be beneficial to the other two categories—being able to improve the fairness for the hidden-terminal case, and both the efficiency and fairness for the disjoint case. The experiments in [19] were conducted by enumerating all the possible combinations of power levels, but the paper did not address the problem on how to find the optimal combination.

One of the seminal work on practical power control protocol is the one by Akella et al. [11], which focused on the scenario of unplanned WLANs. The power control algorithm proposed in [11] is employed by AP to minimize the transmission power without affecting the transmission data rate. The decision of power adjustment at AP is based on either the past transmission status or the average SNR value at the receiver. Similar design idea for power control in WLAN can also be found in [64], where rate adaptation and client mobility were considered. Kowalik et al. [50] applied similar power control idea to general wireless ad hoc networks. The overall throughput can be improved by 15% as compared with fixed maximum power settings.

The side effect of power control is the emergence of asymmetric links. Specifically, a weak sender may hear a strong sender but not the other way around. As a result, due to carrier sensing, the weak sender has much less transmission opportunity than the stronger sender. To mitigate this problem of asymmetric links, Mhatre et al. [58] proposed an algorithm that jointly adjusts transmission power and carrier sensing threshold for AP in

WLAN. They developed both a centralized and a distributed solutions, but most evaluations were performed in simulation. The work by Kim et. al [47] also jointly adjusted the power and carrier sensing threshold, but it was based on an ideal unit-disk graph model.

Despite the apparent benefit of power control, there are several constraints to its implementation in practice. Shrivastava et al. [70] found that, in the indoor environment, the variation of RSSI is so large that it is no longer feasible for the fine-grained power control mechanisms to operate. As a result, a typical 802.11 adapter has only about three to five distinguishable power levels. Similar observation of large RSSI fluctuation in indoor environment can also be found in [8]. The measurement study in our 20-node outdoor testbed shows that this problem is less serious in the outdoor environment due to less multi-path fading. In addition, as we will see in Chapter 5.3.1, we are able to achieve 0.5 dBm granularity of power adjustment by directly setting the power control register in the adapter hardware.

Kowalik et al. [50] reported that some commercial 802.11 adapters do not accurately set to the designated transmission power, i.e., the relationship between the designated power and the actual power is not linear. A positive finding in [50] was that the transmission power of current hardware can be adjusted with negligible latency, thereby making it possible to do per-packet power control.

The limitation of these prior work is that they only considered the power control of MAC Data frames but not that for MAC ACK frames. We show in Chapter 5 that MAC ACK frames can also cause significant interference to neighboring cells.

2.4.2 Other Interference Mitigation Methods

In addition to power control, another effective method to mitigate interference in 802.11 networks is to schedule the transmissions of contending links.

Acharya et al. [9] improved the design of MACA by adding "enhanced parallelism" and proposed MACA-P to increase the number of concurrent transmissions by alleviating the exposed node problem in general multi-hop networks. The basic idea of MACA-P is

to insert a control phase interval between RTS/CTS exchange and Data frame so that two competing senders can negotiate their common transmission instant. Simulation showed that MACA-P can improve throughput by almost 200% for a concentric ring scenario, but the performance of MACA-P will degrade as the number of senders becomes higher. The reason is because the control phase interval may not be long enough to accomodate the negotiations of transmissions among senders. In addition, the evaluation of MACA-P was only performed in simulation, and it is unclear how it will perform in real testbed as it is not trivial to precisely synchronize the senders in practice.

Shrivastava et al. [71] proposed Centaur to improve the spatial concurrency in practical enterprise WLANs. The basic design idea of Centaur is simple: a centralized server schedules the transmissions of each AP in a precise manner such that their transmissions do not interfere each other. In other words, Centaur can eliminate the impacts of both hidden and exposed node problems in enterprise WLANs. The key contribution of Centaur is to realize the above basic design idea using commercial hardware, without modifying the clients. Specifically, the authors proposed fixed backoff, packet staggering, and epoch-based scheduling as the key design components of Centaur. Centaur was evaluated in two practical 802.11 testbeds, and was shown to achieve significant improvement in throughput, delay, as well as the audio quality of VoIP traffic. The limitation of Centaur is that it can only be deployed in enterprise WLANs but not in unplanned 802.11 networks.

The design of Shuffle [57], as discussed earlier in Chapter 2.1.2, shared similar design principle as Centaur. The difference is that Shuffle does not rely on precise scheduling of transmissions from AP but utilizes physical layer capture effect to mitigate the impact of interference and improve spatial concurrency.

Recently, some researchers have attempted to mitigate interference by proposing new physical layer techniques, such as successive interference cancellation [36], chain decoding [73], and directional antenna based on phased array system [54]. The potential throughput gains of these proposals are promising, but they require complex and costly hardware when being deployed in practice.

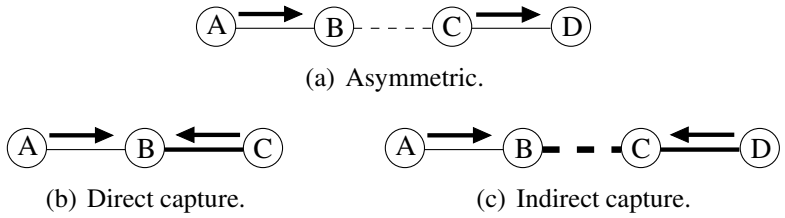
Chapter 3

Mitigating Link Layer Unfairness with FairMesh

In this chapter, we describe FairMesh, our proposed solution to address the MAC layer (or link layer) unfairness problem due to capture effect for wireless 802.11 mesh networks.

As discussed in Chapter 1, there are two basic causes of the unfairness at 802.11 link layer—the unfair topologies (or the asymmetric topologies) and the physical layer capture effect. Figure 3.1(a) illustrates a typical asymmetric topology. The arrows indicate the directions of the data flows, and the dotted lines indicate overhearing links. When node *C* has a backlog of data to send to *D*, node *A* has little chance of successfully sending packets to *B* as the data (or RTS) packets from *A* would likely collide with the data packets from *C*. Prior work has shown that the unfairness arising in this topology can be mitigated by adjusting the contention window *CWmin* [41, 42].

In our measurement study of a large number of flow pairs in a 20-node wireless mesh testbed, we found that the link layer unfairness arising from physical layer capture effect is also common. We broadly classify the capture effect-induced unfairness into two categories—*direct capture* and *indirect capture*. Figures 3.1(b) and 3.1(c) illustrates them, respectively, where the bold lines indicate the capturing links. In our 20-node outdoor wireless mesh testbed, some 92.6% (87/94) and 18.6% (19/102) of the possible 3-



(a) Asymmetric.

(b) Direct capture.

(c) Indirect capture.

Figure 3.1: Topologies that can result in link layer unfairness. The arrows indicate the directions of the data flows, the bold lines indicate the captured links, and the dashed lines indicate overheard.

and 4-node configurations exhibit direct capture and indirect capture, respectively. While MAC unfairness arising from the capture effect was investigated in [32], the authors did not consider topologies with hidden-node collisions that are very common in wireless networks.

In the direct capture scenario, node *B* is able to capture the packets from node *C*, and can successfully decode *C*'s packets even if they collide with the packets from node *A*. This means that the link *AB* could be starved if node *C* has backlogged packets to node *B*. With RTS enabled, node *A* would have a lower throughput than *C* because *C* always wins in the RTS collision at *B*. In the indirect capture scenario, node *C* is able to capture node *D*'s packets, and node *B* is able to capture node *C*'s packets. As such, node *C* can receive *D*'s RTS with high probability; also, the CTS from node *C* has a high probability of "overriding" any RTS from node *A* and preventing node *B* from sending CTS. This causes the throughput of the link *AB* to be significantly lower compared with that of link *DC*. Even if there is capture effect at node *C* but not at node *B*, link *AB* still has smaller throughput than link *DC*, albeit to a lesser degree.

For the rest of this chapter, we will refer to link *AB* (in all the three topologies in Figure 3.1) as the *victim* link and the link *CD* (in Figure 3.1(a)), link *CB* (in Figure 3.1(b)) and link *DC* (in Figure 3.1(c)) as the *offending* links. The sender on the offending link is referred to as the *offender*.

Our extensive evaluation of two existing solutions [41, 42] reveals fundamental and practical shortcomings. As neither of them are explicitly designed to handle capture effect, we found that they are able to achieve reasonable fairness in some, but not for all

three, topologies in Figure 3.1. Even in some scenarios where fairness is achieved, total throughput is reduced.

Motivated by our findings and the need for a simple, practical and distributed solution to mitigate unfairness in 802.11 mesh networks, we designed and implemented *FairMesh*, a new algorithm for adjusting the contention windows among competing flows to achieve approximate max-min fairness. Our key insight is that the nodes that cause an unfair situation to arise and can act to remedy it are often distinct from the ones that can promptly detect and accurately assess the unfairness. To the best of our knowledge, we are the first to decouple the detection and assessment of unfairness from the remedial action. A key strength of our approach is its *simplicity*, which makes it amenable for immediate deployment in practical 802.11 mesh networks using off-the-shelf commodity adapters.

FairMesh has many desirable properties: (i) it is fully distributed with negligible overhead; (ii) it works not only in all the three scenarios identified with two competing flows, with and without Binary Exponential Backoff (BEB), but is also scalable to more than two competing flows in a network with tens of nodes; (iii) it incorporates practical techniques to improve the performance in the presence of lossy links, high data rate, and TCP traffic, and (iv) it can be extended to other notions of fairness besides max-min fairness.

We show with a comprehensive set of experiments, both in simulation and on our 20-node outdoor 802.11 wireless mesh testbed, that FairMesh is not only more fair than 802.11 and prior work, but also achieves near-optimal max-min fairness allocation. The major contribution of FairMesh is thus a comprehensive, yet simple and practical solution to the longstanding problem of unfairness in 802.11 mesh networks. Before describing the details of FairMesh in Section 3.2, we present some results to quantitatively understand the degree of link layer unfairness in the following section.

We organize this chapter as follows: in Section 3.1, we study the degree of unfairness and its correlation with *CWmin*. In Section 3.2, we discuss the design details of FairMesh. Section 3.3 present the evaluation results of FairMesh.

3.1 Understanding Link Layer Unfairness

Unfairness among competing links arises only when the competing links have *backlogged traffic*, i.e., when there is an accumulation of packets at the transmission queue. Such a scenario is not uncommon, especially considering the prediction that the bandwidth-hungry video traffic is expected to make up 69% of the Internet traffic in 2017, more than half of which is carried by WiFi [3]. Thus, we are only concerned with backlogged-traffic scenarios in this thesis.

3.1.1 Degree of Unfairness

To understand the degree of unfairness for the topologies cited in Figure 3.1, we simulated the three topologies with the `ns-2` simulator with backlogged UDP flows using the default 802.11 parameters (*CWmin*= 15, *CWmax*= 1023, and 1,500-byte packets). We implemented the capture effect in our `ns-2` simulation model, by building on the `ns-2` enhancements by Chen et al. [23]. Since Atheros network adapters do not double the backoff window after failed transmissions [39], we also investigated the impact of not using BEB. We verified on our Atheros-based 802.11 testbed that the results without BEB are similar to that produced by the `ns-2` simulations, validating the simulation model. BEB cannot be enabled on our Atheros-based adapters, thus we can only use `ns-2` to evaluate scenarios with BEB enabled.

In Figure 3.2, we compare the throughput of the victim link with the offending link on all three topologies, for data rates of 6 Mbps and 24 Mbps. With BEB enabled, the victim link in all three topologies is starved (i.e., has throughput values that are close to zero). With BEB disabled and RTS/CTS enabled, the victim link is not starved for the asymmetric and direct capture topologies (though unfairness still exists), but is still starved for the indirect capture topology. Without RTS/CTS, the victim links are starved for the first two topologies and the performance is poor for the indirect capture topology.

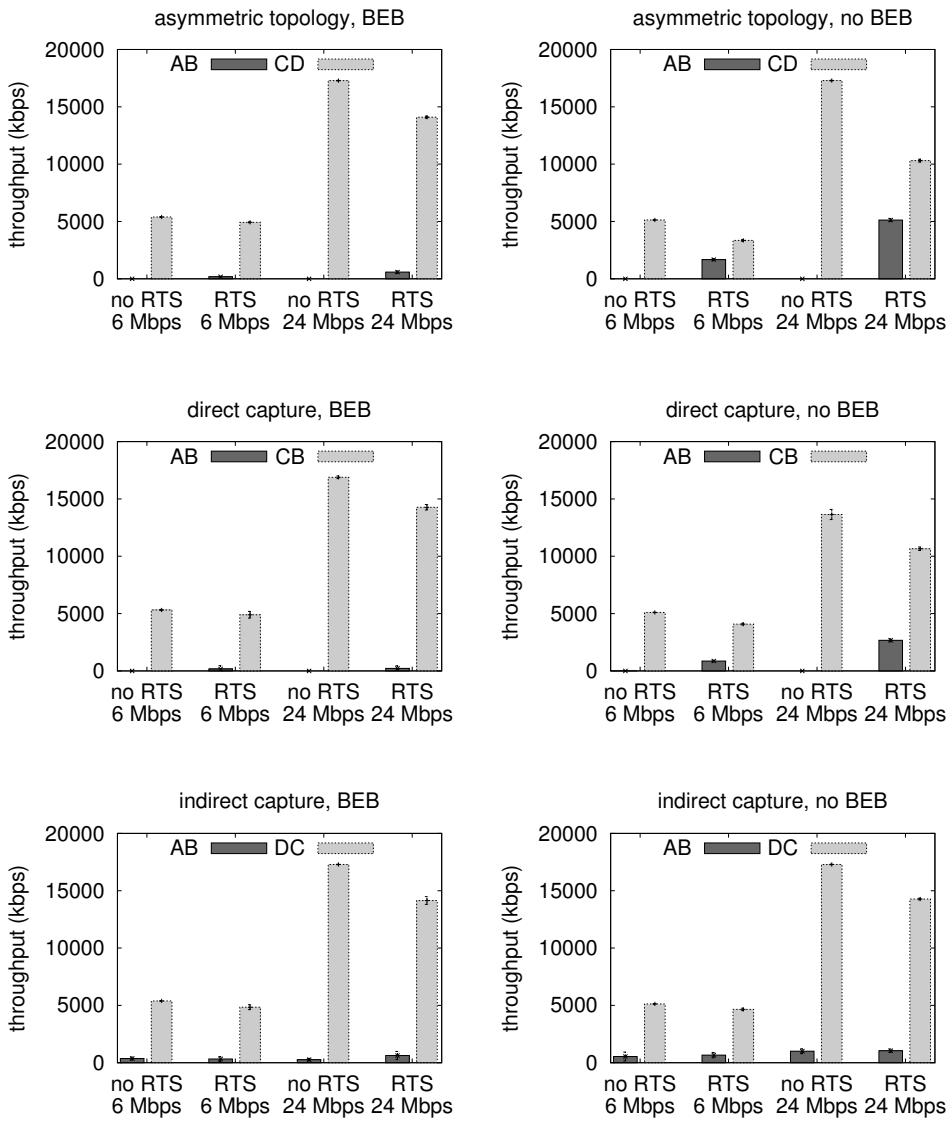


Figure 3.2: MAC unfairness in Figure 3.1.

3.1.2 Design Decisions

A straightforward way to improve fairness is to adjust the waiting time between consecutive packets, so that the opportunity of the victim to access the channel can be increased. There are two important design decisions that we shall carefully address. The first design decision is whether to slow down the offender (i.e., increasing the offender's waiting time between its consecutive packets) or to speed up the victim (i.e., reducing the victim's waiting time). Through both testbed experiment and simulation, we found that speeding up the victim is not sufficient to ensure fairness and sometimes it could exacerbate the

unfairness due to an increased likelihood of hidden-node collisions. We will show that slowing down the offender is more effective in achieving fairness. Note that the 802.11e standard employs a similar "reduced waiting time" mechanism to increase channel access priority, and is thus not too effective either.

The second design decision is which MAC parameter to adjust in order to slow down the offender. There are two possible options: Arbitrary Inter-Frame Space (AIFS) [17, 38] and *CWmin* [41]. The former specifies the minimum waiting time of a sender after the channel becomes idle, whereas the latter determines the random backoff time. Note that, unlike DIFS, AIFS is not fixed, e.g., from 2 to 7 in the 802.11e standard. We found that it is possible to mitigate unfairness by adjusting either AIFS or *CWmin* in the three topologies in Figure 3.1. However, the drawback of adjusting AIFS is that it is possible for a sender with large AIFS value to be completely starved if a neighboring node starts transmitting with the default MAC parameters. This is because, when a sender is waiting for the AIFS period to expire and the channel becomes busy, the sender will start a fresh AIFS period later after the channel becomes idle. With a large AIFS, the AIFS period for the sender might not expire before it is repeatedly renewed, thereby causing starvation. On the other hand, a sender with large *CWmin* will not experience starvation because the backoff counter is frozen when the channel is busy and resumes decrementing after the channel becomes idle. Therefore, we choose to adjust *CWmin* instead of AIFS.

3.1.3 Impact of *CWmin*

To understand the effect of *CWmin* adjustment, we plot the average throughput of backlogged UDP traffic for the three different topologies with different combinations of *CWmin* obtained from `ns-2` simulations in Figure 3.3. The *CWmin* ranges from 15 (default) to 1023 (default value of *CWmax*). The area of each pie is proportional to the total throughput. The white and black sectors represent the throughput of the victim and offending links, respectively.

We make several observations. First, as expected, increasing the *CWmin* of the two

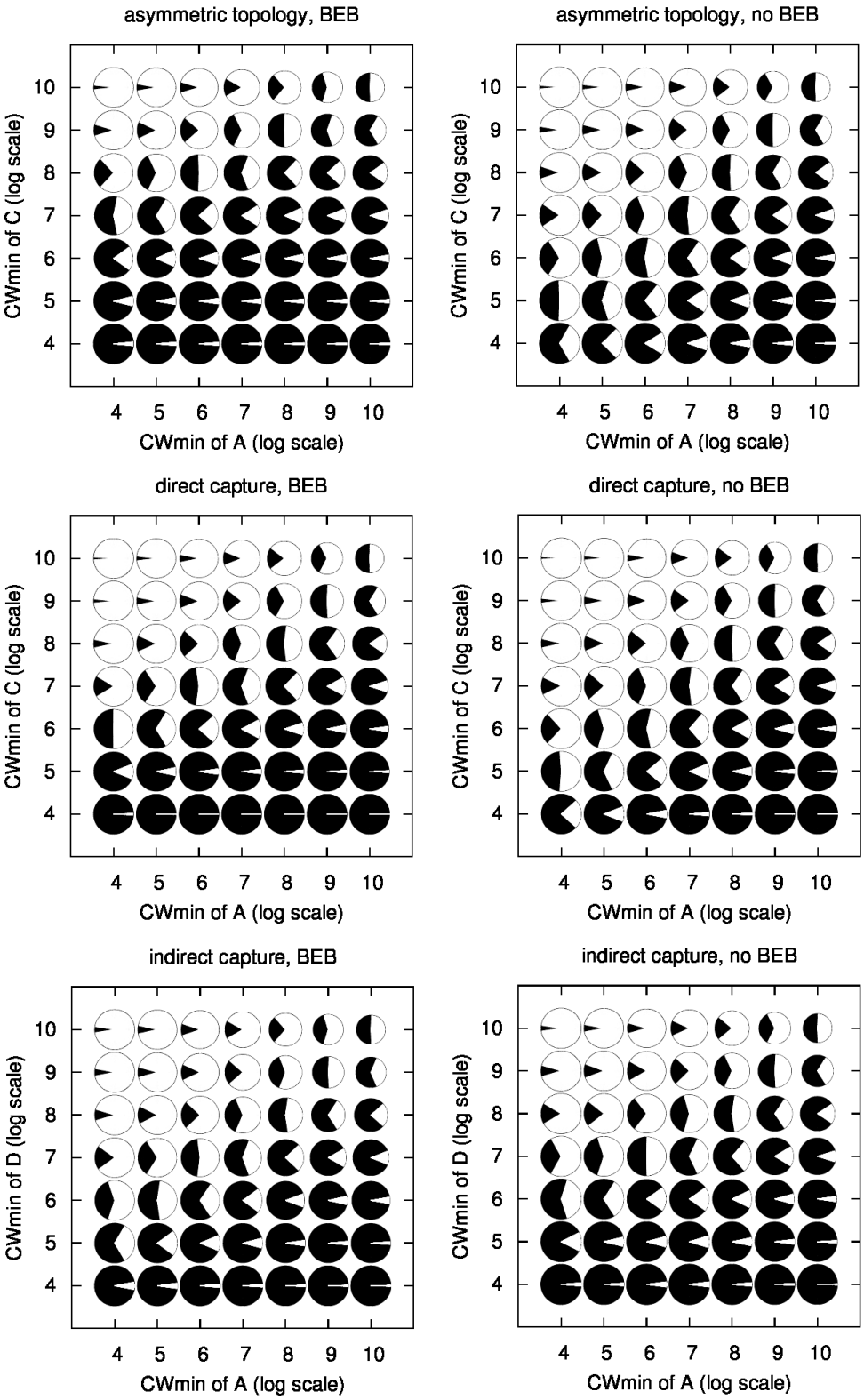


Figure 3.3: Throughput under different combinations of *CWmin*, for the topologies in Figure 3.1, with RTS/CTS. The values on the axes are the logarithms of *CWmin*. Larger pie size indicates better overall throughput. Identical size of black and white sectors means perfect fairness.

flows makes their throughput more equal but hurts the total throughput, i.e., the size of the pie reduces. To allow us to trade off the total throughput against the fairness between the two flows, some notion of fairness is required. In this thesis, we focus on max-min fairness [15]. In Section 3.3.6, we show how we can extend our work to proportional fairness.

Second, the pie graph in Figure 3.3 makes it easy to find the optimal *CWmin* combination for max-min fairness, but it is generally difficult to obtain similar figures for arbitrary topologies. Since 802.11 adapters often allow *CWmin* to be set only to a value of $2^k - 1$, where k is an integer, it is in principle possible to generate Figure 3.3 via offline simulation/measurement. However, each figure is specific to the interaction between a given pair of flows. In practice, the number of possible interacting flows/links could be huge and it is not feasible to probe all possible *CWmin* values to determine the optimal values for each and every topology. In addition, although there are existing proposals [33, 83] to model the asymmetric topology in Figure 3.1(a), it is not trivial to extend these models to arbitrary topologies.

Third, we note that the pies can be loosely divided into two contiguous regions: a black-dominant region and a white-dominant region. The optimal *CWmin* values to achieve max-min fairness must lie on the boundary between the two regions. This observation suggests a simple approach for approximating the optimal *CWmin* values: *increase the* CWmin *of the offender until we reach the boundary between the two regions and once we get there, we move along the boundary until we find the allocation that achieves max-min fairness.* While this idea is simple, implementing it in a distributed way for arbitrary pairs of links is not entirely straightforward and it involves many details, which we describe in the next section.

3.2 FairMesh Design

Unfairness can fundamentally only be mitigated by slowing down the offender(s) that are sending too fast, in order to provide the victim node(s) with a fair chance at accessing the wireless media. Since unfairness does not occur all the time and depends on both the network topology and traffic patterns, we should only intervene when unfairness arises, to avoid interfering with the normal operation of the 802.11 MAC.

In this light, the general approach to solve this problem would involve (i) detecting the existence of unfairness in a timely manner, (ii) identifying the offending link(s) accurately; and (iii) reducing the transmission rate of the offending link(s) appropriately. In a wireless mesh network, it is also preferable to achieve the above in a distributed manner without a central coordinator.

FairMesh detects unfairness and identifies the offending link(s) by continuously tracking and overhearing the transmissions to and from the nodes within its one-hop neighborhood. It turns out that simply overhearing the transmissions of one-hop neighbors is generally not sufficient to provide a node with an accurate view of the transmissions in its locality. As explained in Section 3.2.1, we insert an additional short header (between MAC and IP headers) and piggyback additional information in order to allow nodes to infer the transmissions of packets that are not successfully overheard.

A problem, not commonly considered in previous work, is the coordination among multiple parties that could detect unfairness concurrently in a practical wireless mesh network with multiple flows. To avoid multiple nodes from acting on the same problem and causing the throughput to be reduced excessively, FairMesh uses a distributed algorithm (described in Section 3.2.2) to elect a unique coordinating node. Also, we have found situations where the nodes that can detect an unfair situation are distinct from the ones that can remedy the problem.

Once an offending link is identified, it remains for FairMesh to throttle the offender to improve fairness. There are several possible notions of fairness. FairMesh adopts

max-min fairness [15] because it is simple to implement and practical. We show in Section 3.3.6 that FairMesh can, in principle, be modified to support other notions of fairness, such as proportional fairness.

The performance of FairMesh depends critically on the amount of throttling, as it affects both the eventual efficiency and the fairness of the system. Our investigations into BEB and *CWmin* tuning, briefly described in Section 3.1, suggested that the adjustment of *CWmin*, according to the general principle of moving to the boundary of two unfair regions and then searching along the boundary, can achieve a good solution. In this light, we propose the following algorithm to throttle the offender: the coordinator sends a control message to the offender to double its *CWmin*. Once a change in *CWmin* is detected, the coordinator checks if the change improves the situation. If so, it informs the offender to further double the *CWmin*; otherwise, it informs the offender to roll back the earlier instruction. As we will explain in Section 3.2.3, it is challenging to get this algorithm to work in a general multiple-flow scenario, because after an offender is slowed down, another node may become the new offender.

We have considered different solutions based on these ideas and the final algorithm that we describe in this section represents a solution that achieves approximate max-min fairness in a distributed way, guided by the principles of simplicity and practicality. Our primary motivation is to develop a reliable algorithm that can allow an arbitrary number of flows to operate concurrently in a fair and consistent way, without reducing overall efficiency.

3.2.1 Estimating Throughput Accurately

The degree of unfairness observed at a node is measured using the number of *successfully* transmitted packets per second, which reflects the channel access opportunities of different senders. If the packet size is constant, the packet sending rate is directly proportional to throughput. We track the unfairness on a *per-link* basis rather than a *per-node* basis, as the latter would tend to penalize the nodes with many neighbors. Alternatively, we

can enforce fairness on a per-node basis by aggregating per-link information, if desired. Correspondingly, each outgoing link from a node to a different neighbor also maintains an independent value of *CWmin*, which would be adjusted according to the algorithm to be described in Section 3.2.3.

Each node has knowledge about all the nodes within two hops, which can be maintained at negligible cost, since mesh networks are stationary and relatively stable. Each time a node transmits, receives, or overhears a data packet, the node updates the throughput estimates of all its observed traffic. A *sliding-window*-based approach is used to estimate the throughput: for a link *AB*, the number of packets successfully transmitted from node *A* to node *B* during a given time window is used as an estimate of the throughput for link *AB*. We investigated different time window sizes in our mesh testbed, ranging from several hundred milliseconds to a few seconds, and found that 0.5 to 1 s works well in practice, so we set the window size to 1 s in our algorithm.

While a node can accurately track the number of packets it has successfully transmitted to or received from a neighbor, the same cannot be said for overheard packets. Overheard packets are often corrupted or missed, since we are operating in a regime where collisions are common. This problem causes the throughput of an overheard link to be under-estimated. To address the issue, we introduce a new per-neighbor sequence number and piggyback this information in the packets by adding it into the FairMesh header, which is between the IP and MAC headers. This sequence number is incremented after every successful packet transmission and a node can estimate the throughput of an overheard link from the sequence number instead of by counting packets. If the packets are not of uniform size, it is straightforward to modify the implementation to piggyback the number of successfully transmitted bytes instead. Our approach works even if *rate adaptation* [81] is enabled because fairness is measured by the amount of data transmitted, not by the underlying transmission rate.

3.2.2 Detecting Unfairness

All nodes constantly monitor the traffic to and from its one-hop neighbors. It remains for us to use this information to (i) detect unfairness and situations where we can potentially improve fairness; and (ii) elect a coordinator to coordinate the *CWmin* adjustment.

Since unfairness only arises when competing links are backlogged, FairMesh needs only to consider backlogged links. To identify such links, we piggyback an additional bit in the FairMesh header to indicate if the transmission queue for a link is backlogged.

Identifying the Victim. The backlogged links with the lowest throughput values among all the flows observed within each node's neighborhood are the potential victim links. From the perspective of an observing node, a *potential victim* is defined as the link with the smallest throughput among all the observed links (i.e., among all the outgoing, incoming, and overheard links); and the offending link is a link that has a higher throughput than the victim and also has a *conflict* with the victim. By *conflict*, we mean that their senders cannot successfully transmit packets at the same time, i.e., a link has at least one node within the transmission range of either the sender or receiver of the other link [61]. Note that, if two links share the same sender, they are not considered conflicting links, since their packets share the same transmission queue and do not affect each other's probability of channel access.

Whether a potential victim link is really a victim depends on the notion of fairness. In the case of max-min fairness, which is what FairMesh implements by default, we consider the link with the largest throughput (denoted with l_O) that has conflict with the backlogged link with the lowest throughput (denoted with l_V). If the throughput of l_O exceeds that of l_V by more than a threshold value δ, we consider l_V to be a victim and execute the *CWmin* adjustment algorithm to reduce the rate of l_O. The threshold δ is determined by the estimated error in the throughput estimation, which depends on the size of the sliding window. If a victim link is not found, then a node does nothing and continues to monitor the flows within its neighborhood.

Coordinator Election. It is likely that several nodes simultaneously detect the same

unfair situation (i.e., same l_O and l_V). In FairMesh, one of these nodes will become the coordinator and initiate the *CWmin* adjustment algorithm described in Section 3.2.3. To ensure that there is a unique coordinator for each offender-victim pair, a node determines whether it should become the coordinator according to whether the victim and offending links are incoming, outgoing or overheard links as well as rules $R1$ and $R2$, shown in the following table:

		l_V		
		Incoming	Outgoing	Overheard
l_O	Incoming	YES	$R1$	$R2$
	Outgoing	YES	Impossible	NO
	Overheard	YES	$R1$	NO

A cell with "YES" indicates that the node will become a coordinator; a cell marked $R1$ and $R2$ means that the node will check the following two additional rules to decide if it should be a coordinator. According to Rule $R1$, a node will become the coordinator if the receiver of the victim link cannot hear the offender. According to Rule $R2$, a node will become the coordinator if both the sender and receiver of the victim link cannot hear the offender. An example is shown in Figure 3.4, where it is assumed that AB and CD are l_V and l_O, respectively. Nodes A, B, and C all detect the same unfair situation between AB and CD. From A's perspective, l_V (AB) is outgoing and l_O (CD) is overheard, and thus Rule $R1$ is checked. A is not elected since the receiver of l_V (B) can hear the offender (C). C is not elected either, and only B will be elected. The proof of uniqueness for the coordinator in other scenarios is relatively straightforward by enumeration. Our approach works even when the network topology changes, because the neighborhood information required for $R1$ and $R2$ is obtained from and kept up-to-date by the periodic Hello beacons. As we focused on static mesh networks, such Hello beacons do not need to be transmitted frequently and thus introduce very little overhead.

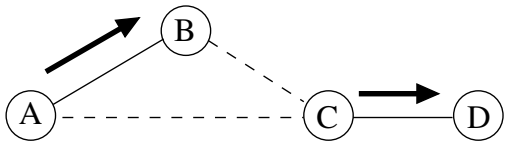


Figure 3.4: Example of multiple nodes detecting the same unfair situation between l_V (AB) and l_O (CD).

3.2.3 CWmin Adjustment Algorithm

FairMesh's *CWmin* adjustment algorithm aims to achieve the max-min fairness between the identified victim and offending links. The coordinator sends a control message to the offender in order to adjust its *CWmin*. In addition to the per-neighbor sequence number (for throughput estimation) and backlog bit (to identify backlogged flows), each packet also contains the *CWmin*. This information allows the coordinator to determine whether its control message has been executed at the offender.

The classic solution to max-min fairness is the *water-filling* algorithm [15], originally proposed for wired networks. It is not feasible to apply this algorithm in our context, because we cannot increase the throughput of all conflicting links at the same rate. Instead, FairMesh takes a simpler and more practical approach inspired by our observations in Figure 3.3: we gradually slow down the offender by doubling its *CWmin* to increase the victim's effective transmission opportunity, until the minimum throughput among all the links (denoted by MIN) cannot be improved. We call this the *water-discharging* algorithm, and the pseudocode for the case where there are two competing links is shown in Algorithm 1.

We illustrate the execution of the water-discharging algorithm for a sample asymmetric topology (see Figure 3.1(a)) found in our 20-node wireless testbed in Figure 3.5. The algorithm is manually activated after 7 s, and the coordinator B starts slowing down offending link CD. One time window (1 s) later, node B observes a better MIN and starts slowing down link AB, which then becomes the offending link. After another time window, a lower MIN is observed, so node B rolls back the previous decision and the algorithm terminates. The resulting *CWmin*'s turn out to be the optimal ones according to

Algorithm 1 Water-discharging *CWmin* Adjustment

```
1:  prev_MIN ← 0
2:  while true do
3:      MIN ← the throughput of l_V
4:      if MIN > prev_MIN then
5:          l_O ← offender
6:          prev_MIN ← MIN
7:          Double the CWmin of l_O, using a small control message
8:          Wait for one time window
9:      else
10:         Undo the previous doubling, using a new control message
11:         Break
12:     end if
13: end while
```

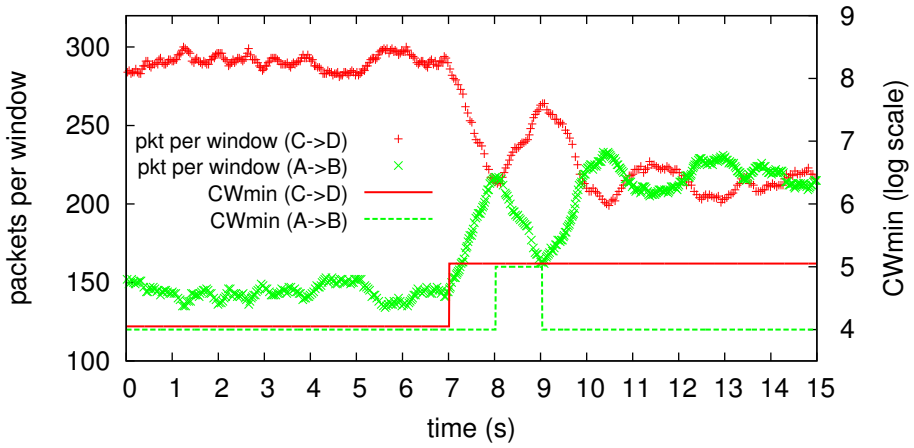


Figure 3.5: The evolution of *CWmin* and the corresponding packet count per window, for a window size of 1 s. The *CWmin* values are in logarithm.

Figure 3.3, and it took only three messages and about 3 s to find the *CWmin*.

Multiple Offending Links. If multiple offending links exist, the coordinator will slow down the *fastest* offender. As the fastest offender is slowed down, another offender might end up taking its place as the new fastest offender, while the victim remains the same without any throughput improvement. In this case, the coordinator does not roll back the *CWmin* of the previous offender, but proceeds to slow down the new fastest offender. This process repeats until either (i) the MIN is improved, and then the coordinator moves to next iteration; or (ii) all offenders have been slowed down without any improvement to the MIN, and the coordinator rolls back all their *CWmin* in a batch before terminating.

Possible Abort. Note that if a coordinator detects an unexpected change in the

CWmin for any of its monitored links when running the water-discharging algorithm, the algorithm will abort and the coordinator will roll back to the previous state with the best MIN. Again, this step avoids several coordinators from modifying the network state simultaneously.

Hysteresis. We found that the natural variation in the throughput estimates for one link is approximately 10% with our chosen window size of 1 s. In other words, the actual throughput for a link can be expected to vary as much as 10% from the estimated value, so we set δ, the threshold for l_O and l_V, at 10%. We also introduce a 10% hysteresis in the water-discharging algorithm and only consider a step to have improved the MIN when the new minimum value is more than 10% larger than the previous MIN. After each *CWmin* doubling, the coordinator will monitor the offending link to check that its control message is executed. Once executed, the coordinator will wait for one window of time to get a updated view of the throughput of the various links that it is monitoring before it executes the next iteration of the water-discharging algorithm.

Periodic Updates. After increasing *CWmin*, a link does not stay at this new *CWmin* value indefinitely, otherwise in the long term, it would itself suffer from unfairness. Rather, the link will halve its *CWmin* every τ time windows if the *CWmin* is larger than the default value of 15. This automatic decay of *CWmin* allows links to return to the default state over time after the original backlogged victim link stops transmissions. If an offender halves its *CWmin* prematurely and causes unfairness, the *water-discharging* algorithm would kick in and cause the *CWmin* to be inflated again. In our implementation, τ is set at 10.

3.2.4 Handling Indirectly Overheard Links

The basic solution presented above is based on the implicit assumption that, between two conflicting links, at least one node (of these two links) can hear both the senders. In other words, at least one node is aware of the actual degree of unfairness between the two links, using the throughput estimation technique described in Section 3.2.1. It turns out that

for the indirect capture topology shown in Figure 3.1(c), this assumption does not hold. Under this unfair situation, none of the nodes overhear both senders simultaneously and hence cannot detect the unfairness.

To detect the unfairness due to indirect capture, we enhance the throughput estimation algorithm to estimate the throughput of an *indirectly* overheard link by counting the ACK from the receiver of the indirect link. In particular, for the topology in Figure 3.1(c), node *B* estimates the throughput of *DC* by counting the ACK from node *C*. A minor technical challenge is that an ACK message only contains the receiver's MAC address and not the sender's MAC address. To circumvent this problem, we use the RSSI reading of a node as a *signature* to identify the ACK sender. The RSSI reading is obtained from either the periodic Hello beacon or a recently transmitted data packet, and the RSSI changes on a time scale that is slow enough for this signature method to be relatively accurate. Note that this unknown-sender problem may not exist for the 802.11n standard, in which the Block ACK frame includes both the sender's and receiver's MAC addresses.

A drawback of this ACK counting method is that it tends to under-estimate the throughput, which makes it difficult to achieve the required max-min fairness. In addition, to address the indirect capture scenario, the coordinator cannot send control messages to the offender directly, but instead will need to have a common neighbor forward the control messages. For example in Figure 3.1(c), when node *B* finds that link *AB* is the victim link and link *DC* is the offending link, it would slow down link *DC* by sending the control message to node *D* through node *C*.

To improve the accuracy of ACK counting, we also implemented a per-neighbor sequence number in the ACKs, like in the data packets. Our current implementation is naive and only works for a fixed packet size, but it is straightforward to support different packet sizes by incrementing the sequence number in byte count. Unfortunately, we are not able to modify the ACK in our 802.11 testbed directly, and are only able to implement this feature in our `ns-2` simulations.

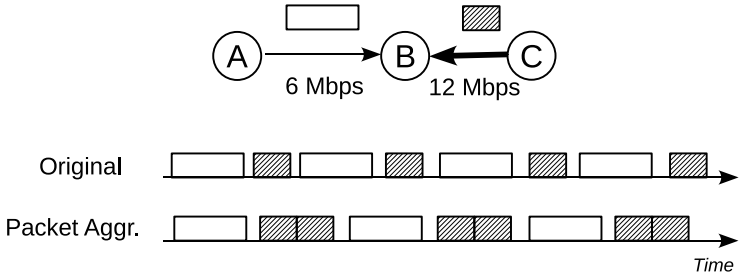


Figure 3.6: Illustration of packet aggregation.

3.2.5 Optimizations

In this section, we describe the optimizations to handle high data rates and also to improve TCP performance.

Handling High Data Rates. FairMesh enables RTS/CTS to mitigate hidden node collisions, which can lead to significant overhead at high data rates. Because we are working with backlogged transmissions, we implemented *packet aggregation* in order to improve efficiency at the higher data rates. This idea is not new and a similar technique has been incorporated in the 802.11n standard: instead of sending one packet per RTS/CTS, we send several packets per RTS/CTS for the higher rates to improve overall throughput.

In addition to reduced overhead, packet aggregation also allows FairMesh to achieve approximate max-min fairness in terms of *transmission air time* instead of throughput. As illustrated in Figure 3.6, link *CB* transmits multiple packets during one RTS/CTS, so that the packets from *AB* and *CB* have comparable transmission times in the air even though they are sending at different data rates. Although with packet aggregation, the lower rate link *AB* sends fewer packets, the time it gives up is more efficiently utilized by the higher rate link *CB*, thereby improving the overall throughput (in *kbps*). In other words, with packet aggregation, FairMesh is able to achieve approximate max-min fairness in terms of transmission air time and does not unduly penalize the link with a higher data rate.

Preventing TCP Starvation. Significant contention in a wireless mesh often leads to high packet loss, causing TCP to go into exponential backoff [60] and to starve. We found that we can prevent TCP starvation if we retransmit failed packets up to three more

times using a second hardware queue, even after 802.11 gives up. This method keeps the packet loss rate below 5%. We are mindful that additional retransmissions will increase latency and may cause TCP timeouts, but we found that three retransmissions achieves a good trade off between latency and reliability in practice.

3.3 Evaluation

We now present our evaluation of FairMesh through an extensive set of experiments, both on our 802.11 wireless mesh testbed and in `ns-2` simulation. We begin with evaluating FairMesh using the three basic problematic topologies in Section 3.3.2. We then evaluate FairMesh with more complex topologies, consisting of multiple competing links, in Section 3.3.3. In Section 3.3.4, we compare FairMesh with two existing proposals to mitigate unfairness. Section 3.3.5 evaluates the interaction between links with different data rates under FairMesh. The performance of FairMesh in the presence of high packet loss rates is evaluated in Section 3.3.6. We then extend the evaluation to large topologies (on our 20-node testbed and a 50-node Berlin topology [59] in simulation) in Section 3.3.7. Finally, the impact of FairMesh on multi-hop TCP is evaluated in Section 3.3.8.

3.3.1 802.11 Wireless Mesh Testbed

Our testbed is deployed in a college dormitory at the National University of Singapore [6] over an area that is approximately 350 m $\times$ 200 m. The dormitory consists of tall and dense buildings, and thus 802.11 radio signals experience significant attenuation. The network has an average node degree of 4.5 and the network diameter is 6 hops (see Figure 3.7).

Hardware. The testbed consists of 20 ALIX boards [1]. We use Compex IEEE 802.11abg adapters [5], with the Atheros AR5414 chipset. All the experiments are conducted using the 802.11a mode. The Atheros adapters have several characteristics that are worth highlighting. First, the BEB mechanism in our adapters cannot be enabled [39].

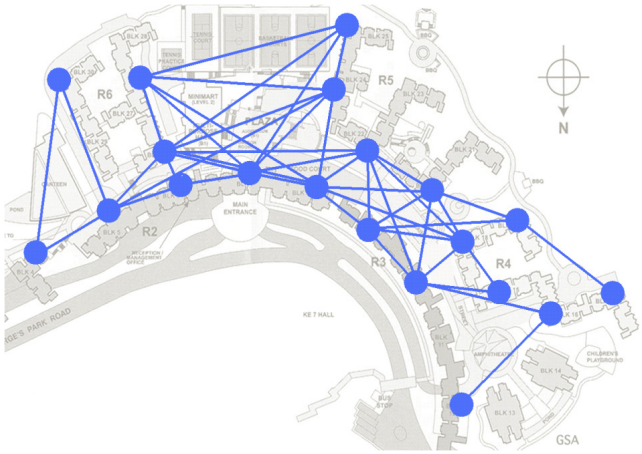


Figure 3.7: Deployment map of the testbed.

As such, we have to resort to `ns-2` simulations to investigate the behavior of FairMesh with BEB. Second, the capture effect in our Atheros adapters is similar to that of the adapters in [52]. Third, like the TFA testbed [21], our Atheros adapters do not employ energy detection in the Clear Channel Assessment (CCA) mechanism. Instead, the channel is considered busy if the hardware is able to successfully decode the preamble and the PLCP header. In other words, the carrier sense range is the same as the transmission range at 6 Mbps.

Software. The board runs OpenWRT Kamikaze 7.09 with kernel version 2.6.25. The driver for the wireless adapter is MadWifi (version 0.9.4) with a default MAC retry limit of 10 and runs in monitor mode. The small control messages for *CWmin* adjustment were sent at the lowest data rate of 6 Mbps, and we found its losses were negligible in practice. We modified the MadWifi driver to insert a small 4-byte FairMesh header between MAC and IP headers (as shown in Figure 3.8), and also to support per-packet *CWmin* adjustment. To avoid the performance degradation described in [74], both the Transmit Antenna Diversity and the Ambient Noise Immunity features were disabled. FairMesh was implemented using the Click modular router [24] (version 1.6.0) in user space. Iperf was used to generate traffic and the default packet size is 1,500 bytes. The transmission queue size in our FairMesh implementation is 50 packets.

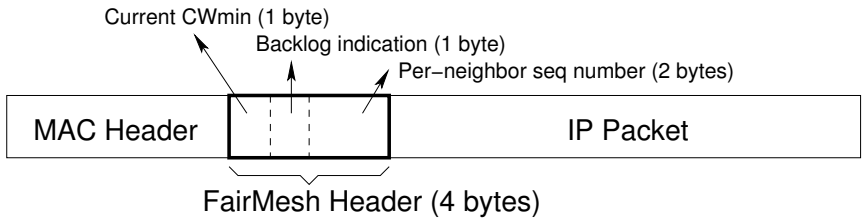


Figure 3.8: Format of FairMesh header in our implementation. FairMesh header has only 4 bytes, and its overhead is negligible as compared with the 1,500-byte payload.

3.3.2 Basic Scenarios

We first run FairMesh on the three problematic topologies in Figure 3.1 to understand its basic behavior, and compare it with 802.11 with and without RTS/CTS, and with and without BEB. Note that since the testbed does not support BEB, the evaluation without BEB is done using the testbed, while the evaluation with BEB is done in `ns-2` simulation using a configuration similar to that on the testbed. In all our experiments in this chapter (except in Section 3.3.8), all source nodes send saturating UDP traffic at the maximum data rate because our goal is to evaluate unfairness under worst case conditions. The average throughput values are summarized in Figure 3.9. The error bars indicate the magnitude of the standard deviation in the data points, each of which is the average throughput over a 1-s interval. Each experiment lasts for 300 s and produces a total of 300 data points.

FairMesh vs. 802.11. As shown in Figure 3.9, there is significant unfairness in all three topologies without RTS/CTS. The victim link *AB* is starved in most cases. As before, we see that BEB will also exacerbate the unfairness. With BEB disabled, RTS/CTS can significantly improve the throughput for the asymmetric topology and somewhat improve that for the two capture topologies. On the other hand, FairMesh is able to achieve significantly better fairness in all scenarios, without affecting the total throughput. In view that the performance of 802.11 without RTS/CTS is so bad, 802.11 will be used with RTS/CTS enabled by default for the remainder of this chapter.

To BEB or not to BEB? FairMesh seems to be able to achieve similar or better fairness without BEB in all cases. For 802.11 with RTS/CTS, the improvement in fairness

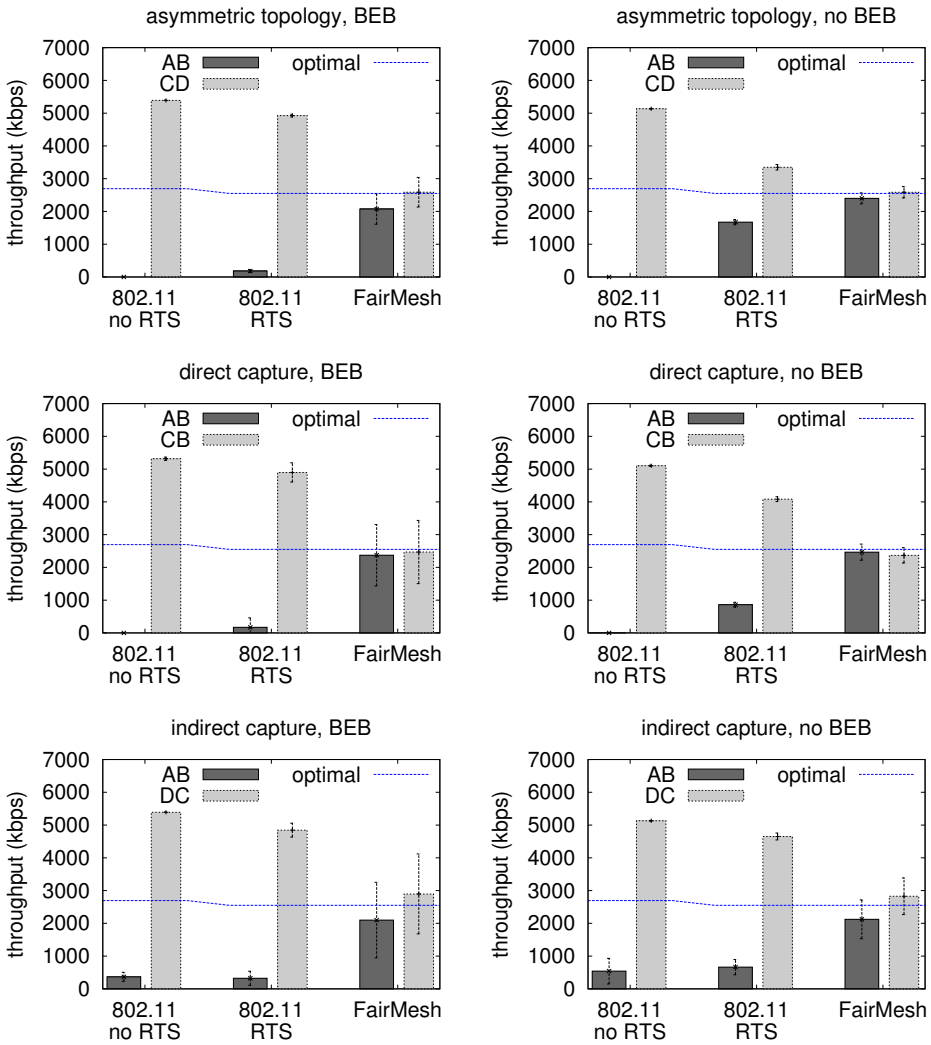


Figure 3.9: Comparison between 802.11 and FairMesh: BEB enabled in simulation, and BEB disabled in testbed.

is especially significant when BEB is disabled. This makes it tempting to completely disable BEB.

It turns out, however, that if we disable BEB, there are scenarios which can result in catastrophic failure and complete starvation. One such scenario is illustrated in Figure 3.10. In this example, A is trying to send packets to B, and C is trying to send packets to D. The packets from A and C are captured by D and B, respectively.

When RTS/CTS is enabled and BEB is disabled, B and D will keep overhearing RTS packets from C and A respectively. This will cause the NAV at both B and D to increase progressively and prevent them from sending CTS packets to A and C. Consequently,

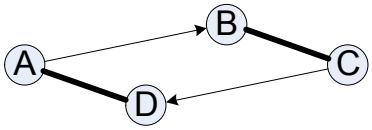


Figure 3.10: Scenario where disabling BEB results in catastrophic failure. The packets from *A* and *C* are captured by *D* and *B*, respectively.

both *AB* and *CD* will get starved.

When RTS/CTS and BEB are both disabled, each sender will continuously send data packets that collide and corrupt the data packet of the other. Since the default inter-packet gap is small (because BEB is disabled) compared to the data packet, the packets will always collide, which again results in starvation.

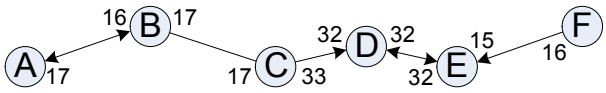
We confirmed that this scenario indeed results in catastrophic failure when BEB is disabled on both our testbed and in simulation. Unfortunately, FairMesh is not able to improve the situation for this scenario because we cannot even detect the traffic to begin with. Overall, our view is that although it is possible to mitigate unfairness to some extent by disabling BEB, caution has to be exercised when doing so in practice.

3.3.3 Optimal Capacity & Multiple Links

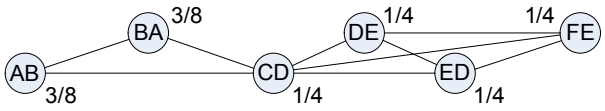
Having shown that FairMesh performs well in the three basic topologies with two competing links, we evaluate FairMesh in a more complex scenario with multiple competing links. We chose a 6-node topology from our testbed (see Figure 3.11(a)) and configured each node to transmit to a neighbor. We replicated this topology in the `ns-2` simulator to evaluate the performance of the same topology with BEB enabled.

To compute the optimal max-min allocation of throughput for this topology, we used a classic *conflict-graph*-based algorithm [13, 41]. The vertices in a conflict graph are the links with backlogged traffic, and there exists an edge between two vertices if the associated links are in conflict with each other[1]. Each *maximal clique* in the conflict graph initially has a nominal capacity of 1. The algorithm iteratively computes the nominal

[1]On our testbed, we consider two links to be in conflict if one link has a node (either sender or receiver) that has a packet delivery ratio of at least 50% to either the sender or receiver of the other link at 6 Mbps.



(a) Network topology.



(b) Conflict graph.

Figure 3.11: Network topology and its conflict graph: the numbers in (a) are RSSI values, and the numbers in (b) are the nominal capacity for the links.

capacity assigned to each node in the conflict graph (or each link in the actual graph). To convert to the absolute capacity, the nominal value is multiplied by the maximum link throughput of 5,100 kbps. The corresponding conflict graph of the 6-node topology and the computed optimal nominal capacity for each link are shown in Figure 3.11(b).

Figure 3.12 compares the performance of FairMesh with 802.11 in the above topology both on the testbed (with BEB disabled) and `ns-2` simulator (with BEB enabled). The computed optimal throughput is plotted as well. The results show that the throughput of FairMesh is much closer to the optimal capacity compared with 802.11. This is especially true for the case with BEB, where *CD* and *FE* are starved for 802.11. We verified that `ns-2` yielded similar results as our testbed for the case with BEB disabled.

Table 3.1 summarizes the median *CWmin* values of the six links and the total number of control messages sent by each node during the experiment. The *CWmin* values with BEB enabled are generally larger than those with BEB disabled, as the victim links contend much less aggressively with BEB enabled. We make two observations about the number of control messages initiated by each node. First, every node was elected at some point as a coordinator, since we can see that they each sent some control messages. Second, compared with the number of data packets (approximately 33,000 per node for the whole experiment), the number of control messages is insignificant, demonstrating that FairMesh incurs very little overhead.

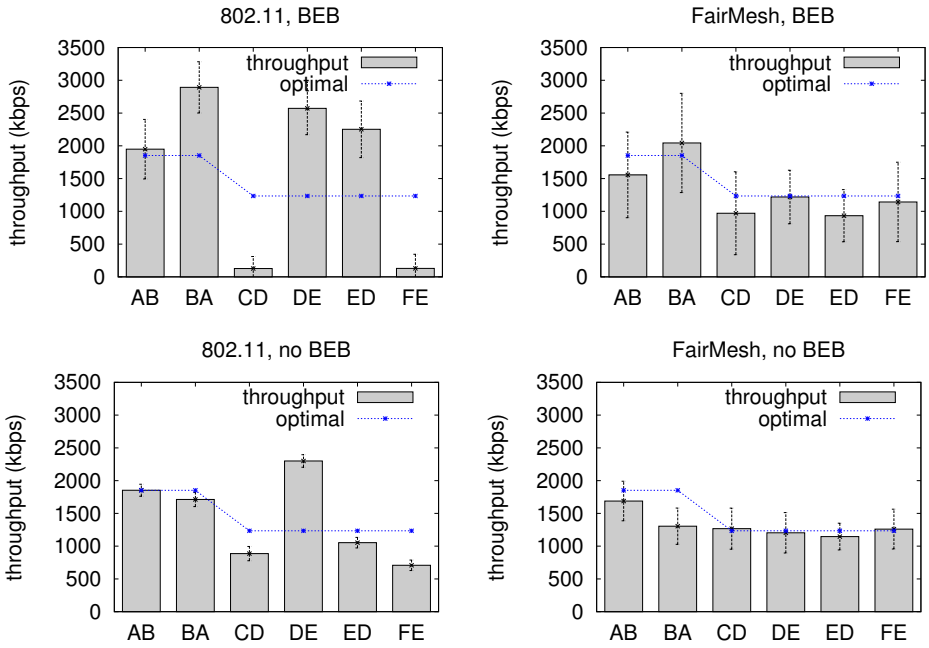


Figure 3.12: Actual throughput and the optimal allocation: with BEB in simulation, and without BEB in testbed.

Table 3.1: Summary of median *CWmin* values for each link and total number of control messages from each node.

CWmin	links	*AB*	*BA*	*CD*	*DE*	*ED*	*FE*
	BEB	15	63	63	255	511	255
	no BEB	31	31	63	31	31	15
Overhead	nodes	*A*	*B*	*C*	*D*	*E*	*F*
	BEB	28	36	26	32	21	22
	no BEB	10	77	42	23	37	45

3.3.4 Comparison with Prior Work

Next, we compare FairMesh with two previous proposals that also address MAC unfairness, one by Huang and Bensaou [41] (denoted with HB), and another by Jian and Chen [42] (called PISD), to demonstrate that FairMesh performs better than existing solutions for mitigating MAC unfairness. The HB method uses naive packet counting to estimate throughput, and does not consider throughput at the granularity of individual links like FairMesh. Besides, for the HB method, a node only adjusts its own *CWmin*, and thus is unable to handle the indirect capture scenario. PISD tries to do away with overhearing by emulating the AIMD mechanism in 802.11 networks. Links with queue

lengths larger than a threshold would use a small *CWmin* to jam the channel, thereby creating a congestion signal. We found, however, that such jamming signals are not always propagated to the appropriate nodes, especially when BEB is enabled.

Both HB and PISD were implemented for the `ns-2` simulator. Our implementation of HB is an approximate one—the original proposed algorithm has a component that estimates the max-min fair throughput; in our simulation, we implemented an oracle that feeds the exact values for the max-min fair throughput into the main HB algorithm. Our HB implementation is thus equally or more accurate than what the proposed HB can achieve. Both HB and PISD operate with RTS/CTS enabled.

Figure 3.13 shows the comparison of FairMesh to both HB and PISD, using the three topologies in Figure 3.1. HB performs well for the asymmetric topology, but is less efficient for the direct capture topology. It is also unable to react correctly for the indirect capture topology. PISD is unable to achieve fairness in the asymmetric topology with BEB enabled. This is because the jamming signal from node *A* does not propagate promptly to node *C*, and link *AB* reduces its rate more frequently than link *CD*. We discovered that most of the evaluations in [42] were conducted with the senders all *within* the carrier sense range of each other, which is why jamming was reported to be effective. PISD performs better in the direct capture scenario because the CTS from node *B* to node *C* can be overheard by node *A*, which prevents node *A* from "blindly" accessing the channel as in the asymmetric topology case.

If we compare the results of FairMesh for the indirect capture scenario without BEB in the testbed (shown in Figure 3.9) with that in the `ns-2` simulation (shown in Figure 3.13), we see that the improvement in the fairness for FairMesh is better for the testbed. We found that this is because node *B* overheard fewer of node *C*'s ACKs in the simulation, while node *B* was able to successfully overhear more of these ACKs in the testbed. Nevertheless, in both cases, FairMesh is still slightly more fair than both HB and PISD.

In Section 3.2.4, we explained that we can improve the accuracy of ACK counting

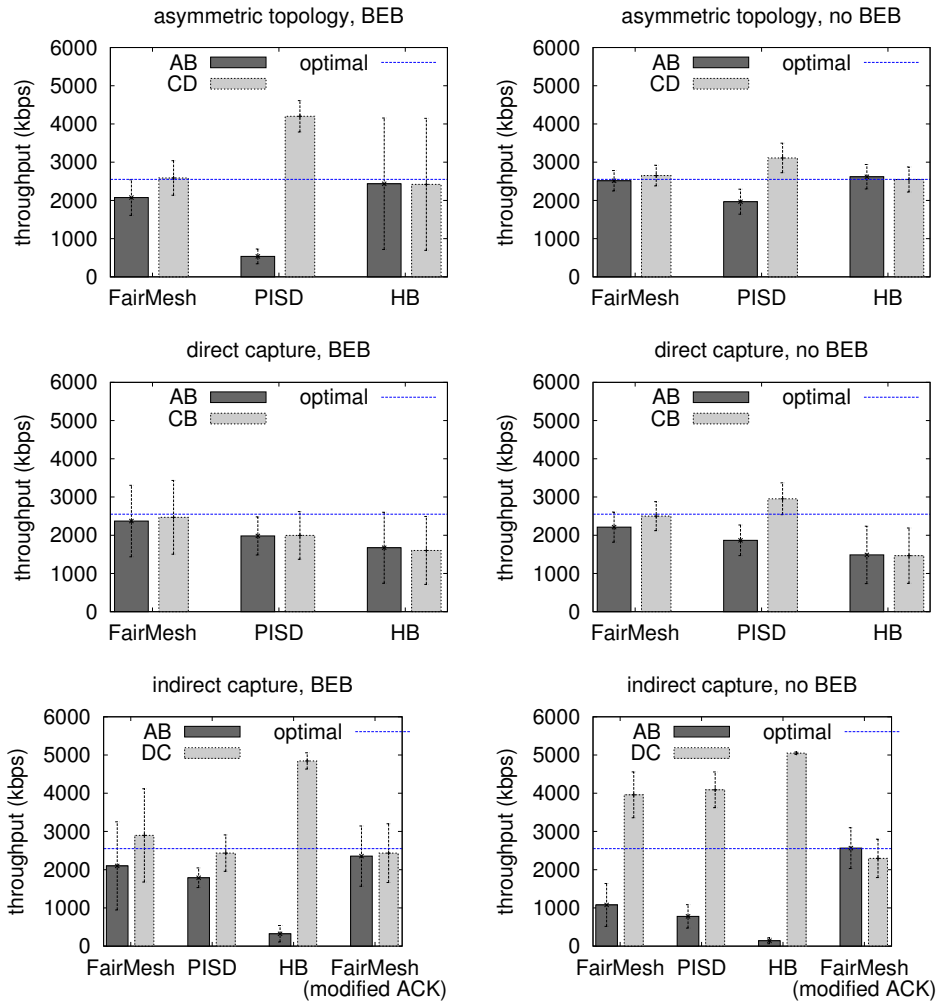
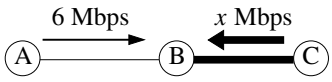
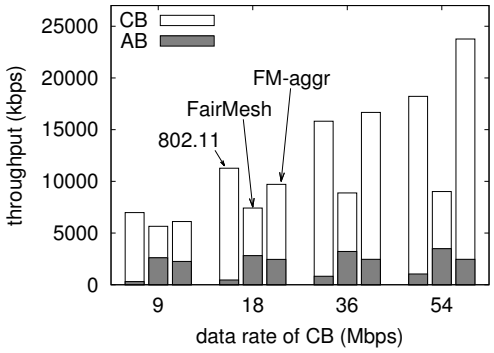


Figure 3.13: Comparison of FairMesh to HB and PISD for all three problematic topologies in simulation.

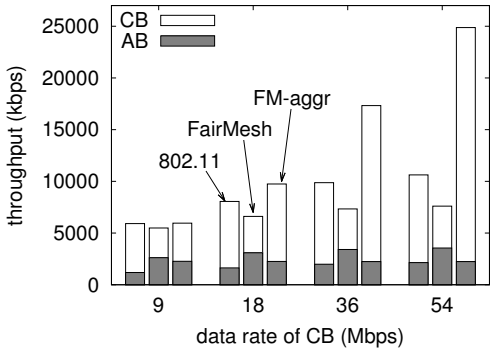
by introducing a per-neighbor sequence number in ACKs, but because we cannot modify hardware ACKs, we can only implement this ACK modification in simulation. In Figure 3.13, we also include the results of FairMesh with the modified ACK for the indirect capture topology. Because the sequence number in ACK enables node *B* to accurately assess the throughput of *DC*, FairMesh is now able to achieve almost equal share between *AB* and *DC*. Note that the modified ACK only provides additional information to assist in the throughput assessment of indirectly overheard links, and it does not interfere with the performance of FairMesh in other topologies. For the rest of the chapter, we will not consider the modified ACK optimization to keep the FairMesh implementation for the testbed consistent with that for the `ns-2` simulation.



(a) Higher data rates scenario.



(b) BEB



(c) no BEB

Figure 3.14: Evaluation of 802.11, FairMesh, and FairMesh with packet aggregation (FM-aggr) under selected higher data rates via simulation.

3.3.5 Higher Data Rates

To evaluate FairMesh with higher data rates, we consider the direct capture scenario shown in Figure 3.14(a), where link *AB* uses 6 Mbps and the stronger link *CB* uses higher data rate of x Mbps, where $9 \leq x \leq 54$. To implement the packet aggregation technique in Section 3.2.5, we aggregate multiple packets of link *CB* into one so that links *AB* and *CB* have the same air time per packet. In this way, FairMesh ensures fairness in terms of *transmission air time* of each link, instead of the original *throughput*-based fairness.

Figure 3.14(b) and Figure 3.14(c) show the results of 802.11, FairMesh, and FairMesh with packet aggregation (labelled as "FM-aggr"), via simulation (we are unable to do packet aggregation on the testbed due to the limited MTU size of our hardware). As compared with 802.11, the original FairMesh achieves almost the same throughput between *AB* and *CB* (i.e., better fairness). However, the total throughput of FairMesh is smaller than that of 802.11 (i.e., reduces overall efficiency). This is because FairMesh re-allocates certain amount of transmission air time from the fast link *CB* to the slow link *AB*. With packet aggregation, FairMesh significantly increases the total throughput, with only a slight drop in the throughput for *AB*.

3.3.6 Lossy Links & Proportional Fairness

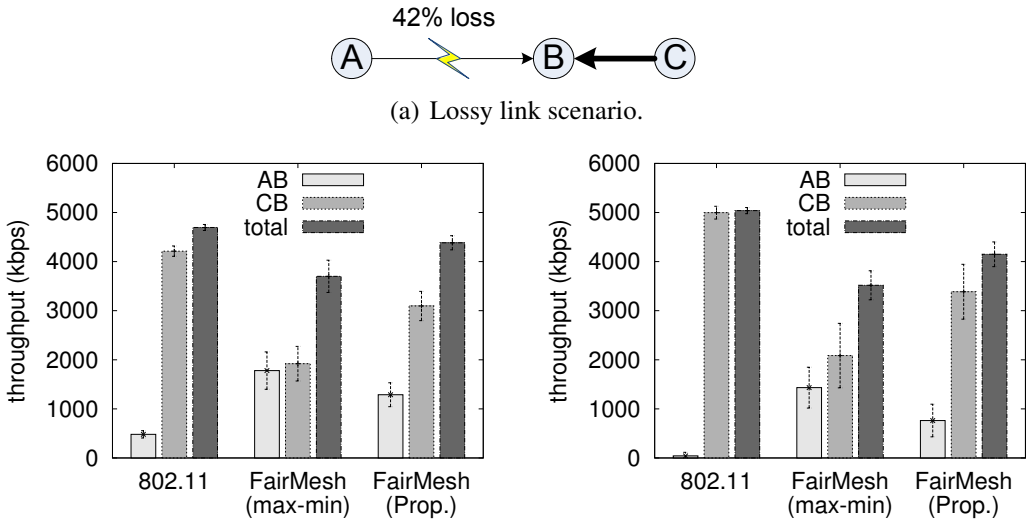
Next, we demonstrate that FairMesh works well even with lossy links, and that FairMesh can be easily modified to work with a different notion of fairness besides max-min fairness.

Take the sample testbed topology in Figure 3.15(a) for example, where node *B* can capture *C*'s packets and link *AB* has a loss rate of 42% due to poor RSSI. FairMesh is still able to produce comparable throughput for the two links. However, the transmission opportunity given up by link *CB* does not translate into the same number of effective transmissions for link *AB*. With 42% losses, for every 5 transmissions given up by link *CB*, link *AB* would only gain approximately 3 successful transmissions. This leads to a significant loss in overall efficiency and this is one situation where proportional fairness [45] would allow us to trade off fairness for better efficiency.

FairMesh can be easily modified to support proportional fairness, by checking whether a *utility function* $\sum ln(r_i)$ improves in the water-discharging algorithm, rather than the MIN, where r_i is the throughput of an observed link *i*. Figure 3.15(b) and Figure 3.15(c) show the results of FairMesh (max-min), FairMesh (proportional fairness) and 802.11 for the scenario in Figure 3.15(a). Compared with FairMesh (max-min), FairMesh (proportional fairness) achieves a higher total throughput at the expense of slightly less equal allocations between *AB* and *CB*.

3.3.7 Large-Scale Experiments

Having shown that FairMesh performs well in the three basic topologies with two competing links, we evaluated FairMesh in a more complex scenario with *large-scale arbitrary competing links*. We used our 20-node testbed to study the case without BEB, and used `ns-2` simulation to study the case with BEB. In these experiments, each node sends saturating UDP traffic to one randomly chosen neighboring node. The point is not to attempt to simulate real traffic, but to investigate how FairMesh and existing algorithms hold up



(a) Lossy link scenario.

(b) no BEB (testbed).

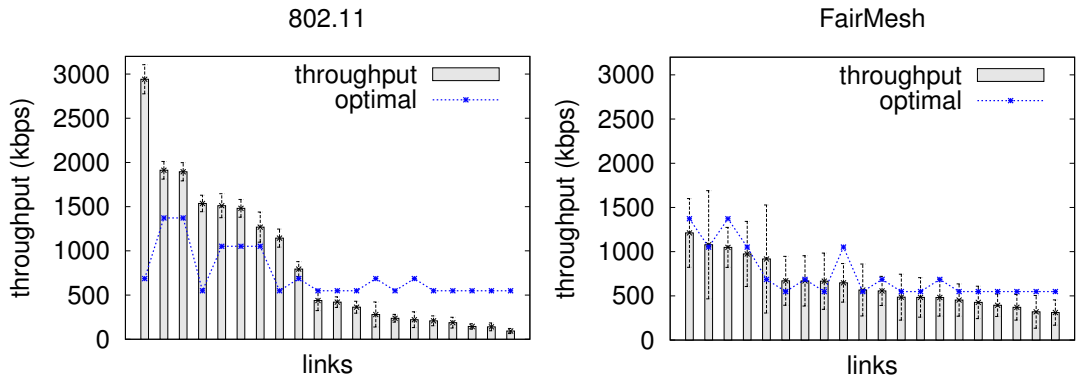
(c) BEB (simulation).

Figure 3.15: Scenario with lossy link: max-min and proportional fairness have different impacts.

under conditions of severe contention, where unfairness is most likely to manifest. As a benchmark for comparison, we used the classic *conflict-graph*-based algorithm [41] to compute the optimal max-min allocation of throughput.

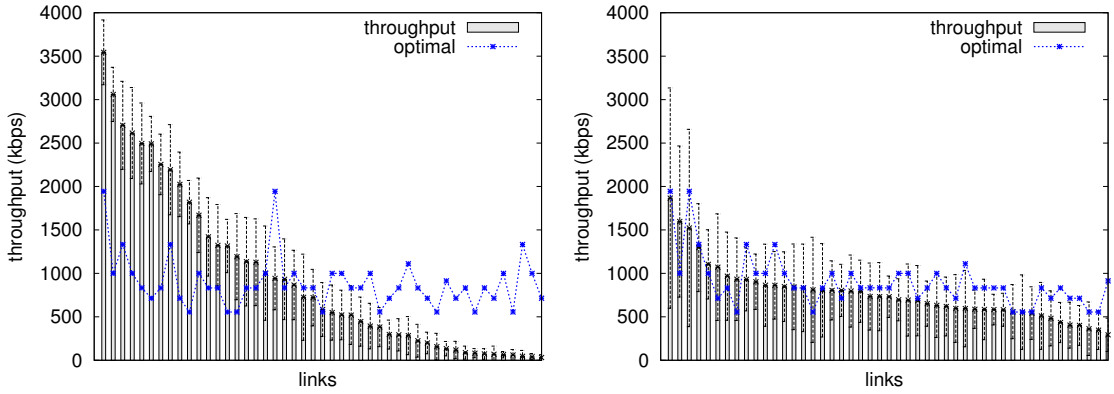
Testbed. We randomly selected 20 links in the 20-node testbed to evaluate FairMesh with multiple concurrent flows. Figure 3.16 shows the performance of 802.11 and FairMesh on the testbed. We plot the optimal max-min throughput allocation with a blue dashed line. The links are sorted in descending order of throughput. We observed that the distribution of the link throughput for FairMesh is significantly closer to the optimal max-min fair allocation. Also, the overall throughput of 802.11 is slightly higher than that of FairMesh, as the throughput reduction of one link can cause a throughput increase in multiple links, i.e., 802.11 exploits spatial diversity more efficiently at the expense of fairness. The total rate of the control messages for the entire network is about 1.48 packets per second, which is negligible compared to the data throughput.

Simulation. We next performed a large-scale simulation study of FairMesh on a 50-node topology with an average degree of 3.6, generated using the Berlin network methodology [59]. This simulation allows us to study the behavior of FairMesh with BEB in a large network, as well as to compare FairMesh with HB and PISD. Like before, each



(a) 802.11 (b) FairMesh

Figure 3.16: Comparing FairMesh to 802.11 in the real 20-node testbed. The links are sorted according to their throughput results in descending order.



(a) 802.11 (b) FairMesh

Figure 3.17: Comparing FairMesh to 802.11 (with BEB) via simulation. The links are sorted according to their throughput results in descending order.

node randomly selects a neighbor as receiver and sends saturating UDP traffic. We plot the resulting throughput in Figure 3.17. Again, FairMesh achieves a throughput allocation that is much closer to the optimal max-min fair allocation than 802.11, which has a large number of nearly starved links. To help us visualize the differences in fairness among FairMesh, HB and PISD, we also plot the CDF of the throughput ratio between the achieved throughput and the optimal max-min throughput, for the 50 links in Figure 3.18. FairMesh achieves a throughput allocation that is closer to the optimal max-min allocation than both PISD and HB. In particular, PISD does not perform well in terms of fairness improvement. On the other hand, while HB achieves better fairness, it has poorer overall efficiency.

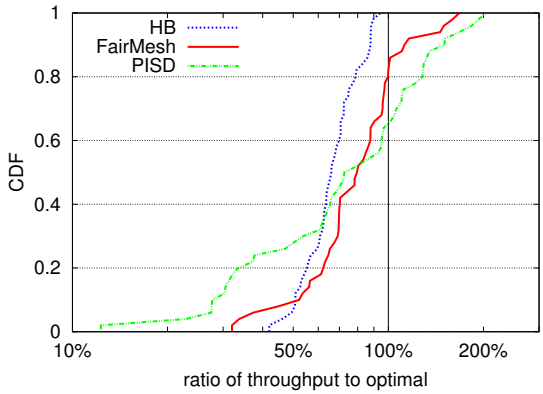


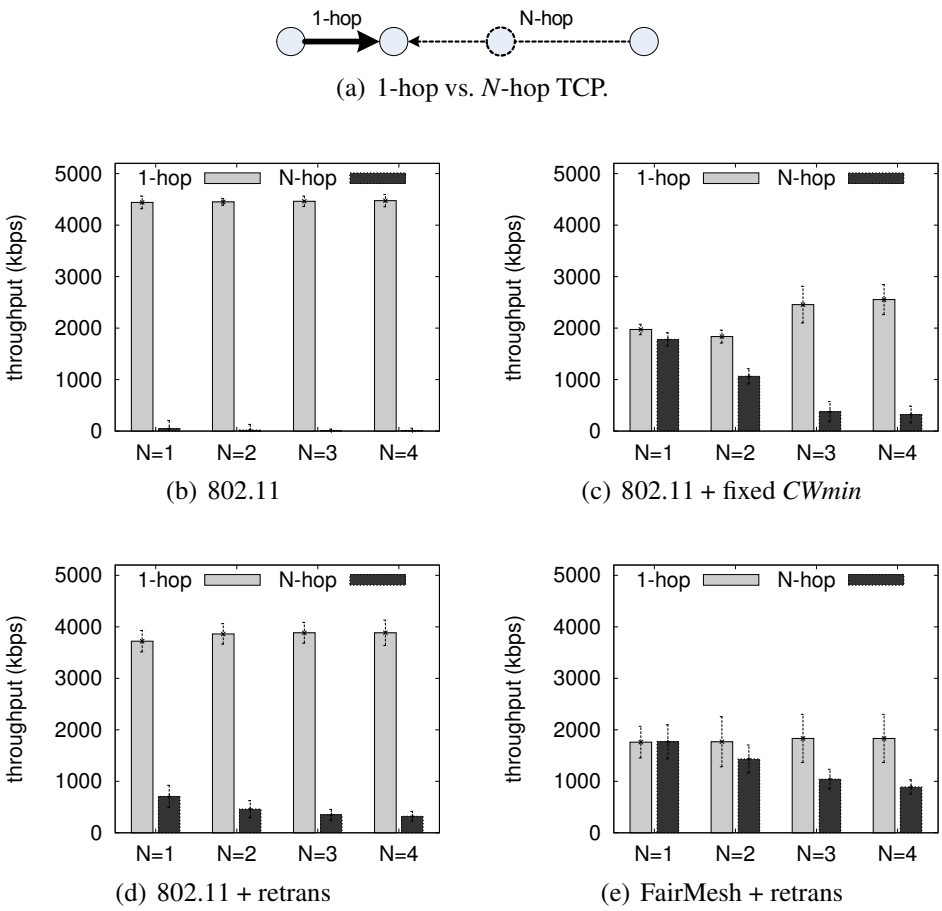
Figure 3.18: CDF of throughput ratio to optimal, of the large-scale simulation experiment.

We investigated the convergence performance of FairMesh. As a link in FairMesh periodically halves its *CWmin* to increase throughput, its *CWmin* does not converge to a fixed value but varies within certain range of the possible seven *CWmin* levels (i.e., from 4 to 10 in logarithmic value). To evaluate the variation of *CWmin* in FairMesh, we recorded the interdecile range (i.e., from 10% to 90%) of each link's *CWmin* logarithmic values during the 300-s experiment above. We found that the majority of the links (88%) have their interdecile range of *CWmin* equal to or less than 3. This implies that FairMesh does not cause significant fluctuations in the *CWmin*.

3.3.8 TCP & Multi-Hop Flows

Finally, we show that FairMesh can improve the fairness perceived by TCP. To evaluate the efficacy of the additional retransmissions in FairMesh (see Section 3.2.5), we compared FairMesh with 802.11 and the *counter-starvation policy* proposed by Shi et al. [69] (denoted as "fixed *CWmin*"). To implement Shi et al.'s counter-starvation policy, we set the *CWmin* for the nodes near the gateway to 255. For our evaluation, we randomly selected various "1-hop vs. N-hop" topologies ($1 \leq N \leq 4$) from our testbed (see Figure 3.19(a)), and ran two concurrent TCP flows on them. Typically, the N-hop flow not only experiences contention within itself but also gets overwhelmed by the 1-hop flow, because of the capture effect at the destination (which acts like a gateway).

Figure 3.19 summarizes the results. As expected, all the N-hop TCP flows are starved



(a) 1-hop vs. *N*-hop TCP.

(b) 802.11

(c) 802.11 + fixed *CWmin*

(d) 802.11 + retrans

(e) FairMesh + retrans

Figure 3.19: Impact of FairMesh on TCP: 1-hop TCP vs. *N*-hop TCP. The 1-hop flow has higher RSSI and its packets can be captured by the common destination. Maximum 1-hop TCP throughput is 4,600 kbps.

for vanilla 802.11 due to frequent TCP exponential backoff. The counter-starvation policy improves fairness for $N = 1$ and $N = 2$, but not for $N > 2$. In fact, we found that the 4-hop TCP flow occasionally got starved, as the packet loss rate was still relatively high. While 802.11 with retransmissions can prevent starvation, the N-hop TCP flows suffer significant throughput degradation because the underlying MAC unfairness is not addressed. FairMesh with retransmission improves the TCP throughput for the N-hop TCP flows significantly, especially for large N. An interesting observation is that, because FairMesh seeks to achieve per-link fairness, it achieves almost equal throughput for $N = 1$ and $N = 2$, i.e., where all the links compete with each other and end up sharing the air time equally.

3.4 Summary

In this chapter, we propose and evaluate FairMesh to mitigate the link layer unfairness arising from physical layer capture effect. Our key insight is that the nodes that cause an unfair situation to arise and can act to remedy it are often distinct from the ones that can accurately assess the degree of unfairness. By intelligently selecting a set of nodes to coordinate *CWmin* adjustment, FairMesh is able to achieve approximate max-min fairness in a distributed manner for arbitrary sets of links. In addition, we show that FairMesh remains efficient under high data rates and high loss rate and interacts well with TCP in multi-hop scenarios. In the next chapter, we will investigate the impact of Message In Message (MIM) mechanism, which is a special case of capture effect, in 802.11n networks with frame aggregation.

Chapter 4

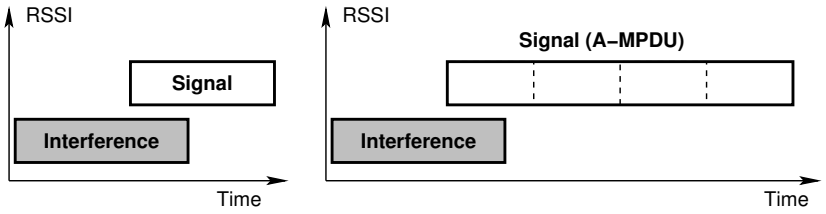
Potential Pitfalls of the Message In Message Mechanism

In this chapter, we study the performance impact of the Message in Message (MIM) mechanism, a special case of capture effect, in 802.11n networks with frame aggregation.

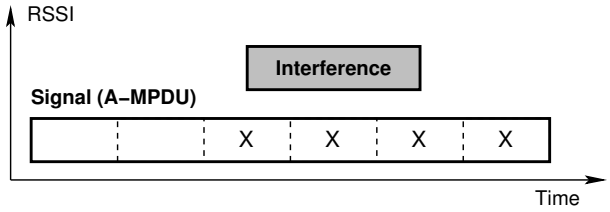
4.1 Motivation

Message in Message (MIM) is a physical layer mechanism in modern 802.11 adapters [52, 68]. When MIM is enabled, the adapter will abandon the ongoing reception of an 802.11 MAC frame to start decoding another frame with a stronger signal. In other words, a frame with a higher signal strength can "knock out" a frame with lower signal strength. This mechanism is also present in the adapter hardware for sensor networks [80].

It has been shown that MIM can help to improve spatial reuse in 802.11g WLANs [57] and also reduce packet loss in sensor networks [31]. Figure 4.1(a) serves to illustrate why MIM can often be helpful in terms of throughput. In this example, the target data frame is stronger, but arrives in the midst of a weaker interfering frame. In this situation, MIM will cause the interfering frame to be "knocked out" and switch to receive the target frame. Without MIM, both frames would have been corrupted and the sender of the target frame



(a) The cases where MIM is helpful.



(b) The case where MIM is detrimental.

Figure 4.1: MIM may not always be helpful.

will have to retransmit the lost frame.

The 802.11n standard now includes a new feature called aggregate MAC Protocol Data Unit (A-MPDU) to reduce transmission overhead [72, 63, 51], where several frames are aggregated into one physical data frame. What we have found is that MIM can be detrimental when a short but stronger interfering frame collides midway with a long, but weaker A-MPDU. We illustrate the problem in Figure 4.1(b). Without MIM, the receiver would successfully decode the first two and also the last frame of the A-MPDU and reply with a Block ACK (BA) indicating that the middle three frames were corrupted. The sender would then only have to retransmit the corrupted frames. However, with MIM enabled, the A-MPDU would be knocked out by the interfering frame and thus the last frame in the A-MPDU cannot be decoded. In addition, the receiver would not send a BA. This will cause the sender to either time out and retransmit the entire A-MPDU, or to send a Block ACK Request (BAR) frame. The 802.11 standard [7] does not mandate that the BAR be sent and we found that the Cisco 1140 AP in our campus WLAN does not do so. This means that full retransmissions of the entire A-MPDU will be sent, incurring unnecessary overheads.

To better understand how MIM could potentially degrade performance under unfavor-

able scenarios, we conducted a comprehensive set of experiments on the MIM mechanism using commercial 802.11n adapters. In particular, we investigated various scenarios by varying A-MPDU size, interfering frame air time, and received signal strength difference. We also studied the effect of channel bonding and adjacent-channel interference. Interestingly, we found that strong interference at an adjacent channel is still able to knock out an A-MPDU that is being received. While our observation is simple to understand, to the best of our knowledge, we are the first to comprehensively study how enabling and disabling the MIM mechanism can affect the reception of A-MPDU under interference.

Our study suggests that instead of necessarily always being helpful, the MIM mechanism is a double-edged sword, and it should be enabled judiciously. To this end, we proposed and evaluated a simple yet effective method to adaptively decide when to enable MIM by monitoring the received data frames. We show that our proposed adaptive MIM scheme is able to achieve near-optimal throughput with commodity 802.11 adapters.

4.2 Impact of MIM: a qualitative study

In this section, we describe a measurements study where we investigate the impact of MIM with modern 802.11n adapters. The experiments were conducted with our campus enterprise WLAN. First, we identified two adjacent access points (AP) that were hidden from each other, and placed a client at three different positions as illustrated in Figure 4.2. Position-1 has a stronger received signal strength (RSS) from AP-1 than that from AP-2, while Position-3 has a stronger RSS from AP-2 than that from AP-1. Position-2 was a point where the RSS from both AP-1 and AP-2 were similar. The corresponding RSS values at each position are shown in the following table:

Position	1	2	3
RSS from AP-1 (dBm)	-59	-66	-73
RSS from AP-2 (dBm)	-73	-65	-57

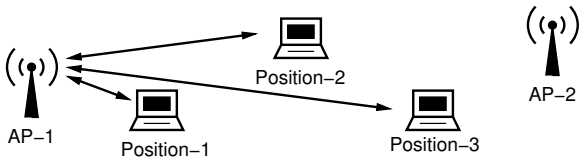
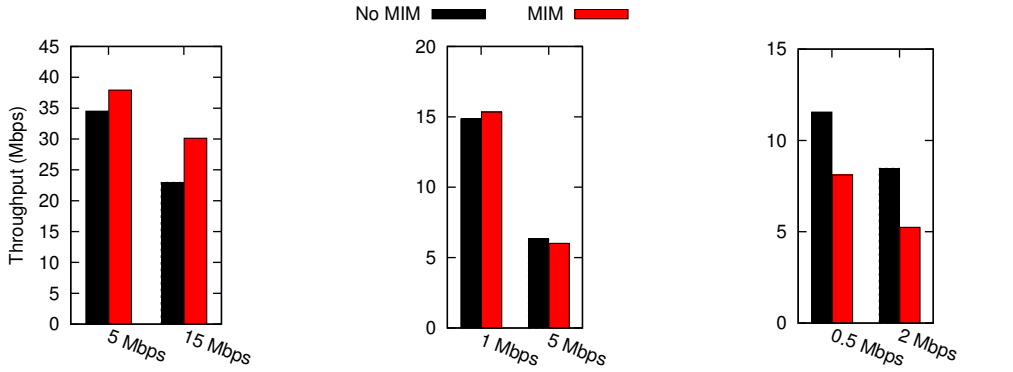


Figure 4.2: Campus WLAN experiment setup.

The client is an Advantech board with a Compex WLE350NX adapter in which we were able to enable and disable the MIM mechanism (more details in Section 4.3.1). The experiment was conducted during the vacation period to minimize interference from user traffic. The first client is associated with AP-1 and a saturating UDP flow with 1,500-byte payload was sent to the client from a machine in our laboratory. To generate interference from AP-2, we connected another client (not shown in Figure 4.2) to AP-2 and sent a 1,500-byte UDP flow to this interfering client from another machine in our laboratory. To mitigate MAC ACK interferences from the interfering client, the interfering client was positioned further away from AP-1's client (i.e. on the "other" side of AP-2) and also operated at reduced transmission power. For each test, the UDP flows ran for 30 s and we measured the throughput obtained by AP-1's client. We repeated the test three times at each position to mitigate the impact of external factors.

Figure 4.3 shows the average UDP throughput of the client that was connected to AP-1, for both with MIM enabled and with MIM disabled. At Position-1, the average throughput was higher with MIM enabled than with MIM disabled, which shows that MIM is indeed helpful as expected when the RSS of the interfering signal is lower than the target frame. At Position-3, the average throughput with MIM enabled was much lower (more than 30%) than that when MIM was disabled. When examining the first trace at Position-3, we found that about 28% of the A-MPDU that the client had started to decode from AP-1 were "knocked out" by the interfering frames from AP-2 as these frames had a higher RSS than the target A-MPDUs. At Position-2, enabling or disabling MIM had little effect since both the target and interfering frames had similar RSS. We conclude from this simple experiment that enabling MIM is not always helpful, and can



(a) Advantageous (Position-1) (b) No effect (Position-2) (c) Detrimental (Position-3)

Figure 4.3: Impact of MIM mechanism for three different scenarios. The x-axis refers to the throughput of the interfering frames from AP-2. We did not impose the same level of interference as the one for Position-1, as it would cause the throughput from AP-1 to drop to nearly zero for the other two positions regardless of MIM.

in fact, potentially be detrimental when the interfering signal is stronger than the target A-MPDU.

4.3 Effect of MIM on A-MPDU Reception

In this section, we investigate how the MIM mechanism affects the reception of A-MPDUs under interference in commercial 802.11n adapters. In particular, we study the impact of A-MPDU size, interfering frame air time, and received signal strength difference on the reception of A-MPDU when MIM is turned on/off. We also consider the scenarios when channel bonding is used and when there is adjacent-channel interference. The Frame Delivery Ratio (FDR) of A-MPDU is used as the performance metric.

4.3.1 Experimental Methodology & Setup

Three physical WiFi adapters were used to conduct our experiments. They were placed a few meters apart in our laboratory according to the topology shown in Figure 4.4, with the interfering node placed closer to the receiver so the receiver will receive a stronger signal from the interfering node than that from the sender. The receiver is an Advantech board with a Compex WLE350NX adapter (Atheros AR9580 chipset). The sender

Table 4.1: Data rate and the corresponding size of the A-MPDU.

MCS index	0	1	2	3	4	5	6	7
Data rate (Mbps)	6.5	13	19.5	26	39	52	58.5	65
A-MPDU size	2	4	6	8	12	16	18	20

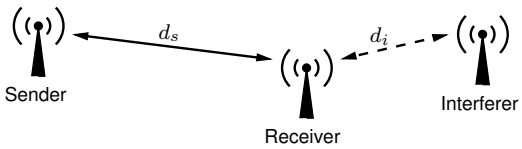


Figure 4.4: Experiment setup for MIM characterization. $d_s > d_i$ ensures that the interfering node has a stronger signal than the sender at the receiver.

and interfering node are both ALIX boards with a Wistron DNMA92 adapter (Atheros AR9220 chipset). All the three boards run OpenWRT (Attitude Adjustment release) with the ath9k WiFi driver. The MIM mechanism is enabled by default at the receiver and can be disabled by clearing a bit in the `AR_PHY_RESTART` hardware register. We set the adapters to operate in *monitor* mode as other modes like *managed* mode and *ibss* mode will implicitly transmit some management frames, which may affect the FDR results. All the experiments (except the ones in Section 4.3.5 and Section 4.3.6) were conducted over channel 56, which we had verified to have little interference from our campus WLAN.

Figure 4.5 illustrates a simple scheme we used to ensure that the interfering frame arrives later than the target A-MPDU at the receiver. The receiver first broadcasts a small poll message at the lowest data rate which will trigger the sender to immediately transmit an A-MPDU (with no retry) to the receiver. The interfering node with carrier sensing disabled, will wait for a random time t before sending the interfering frame with a broadcast MAC destination address. t is uniformly distributed between zero and the air time duration of the A-MPDU. This ensures that the interfering frame will always collide with the A-MPDU that is being received at the receiver. In our experiments, the poll messages were sent every 50 ms, i.e., we collected 20 samples every second, over 100 s. To mitigate the impact of any external factors, we toggled MIM between enabled and disabled every 10 s. We also conducted a separate experiment to verify that the frame delivery ratio of

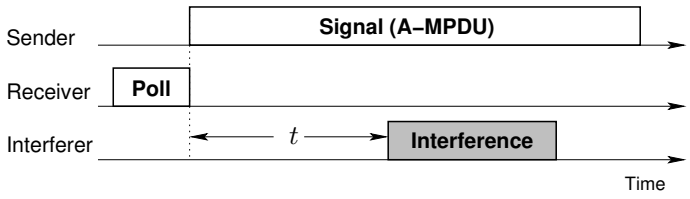


Figure 4.5: Polling scheme ensures that the interfering frame arrives t time later than the A-MPDU. t is uniformly distributed between zero and the air time duration of the A-MPDU.

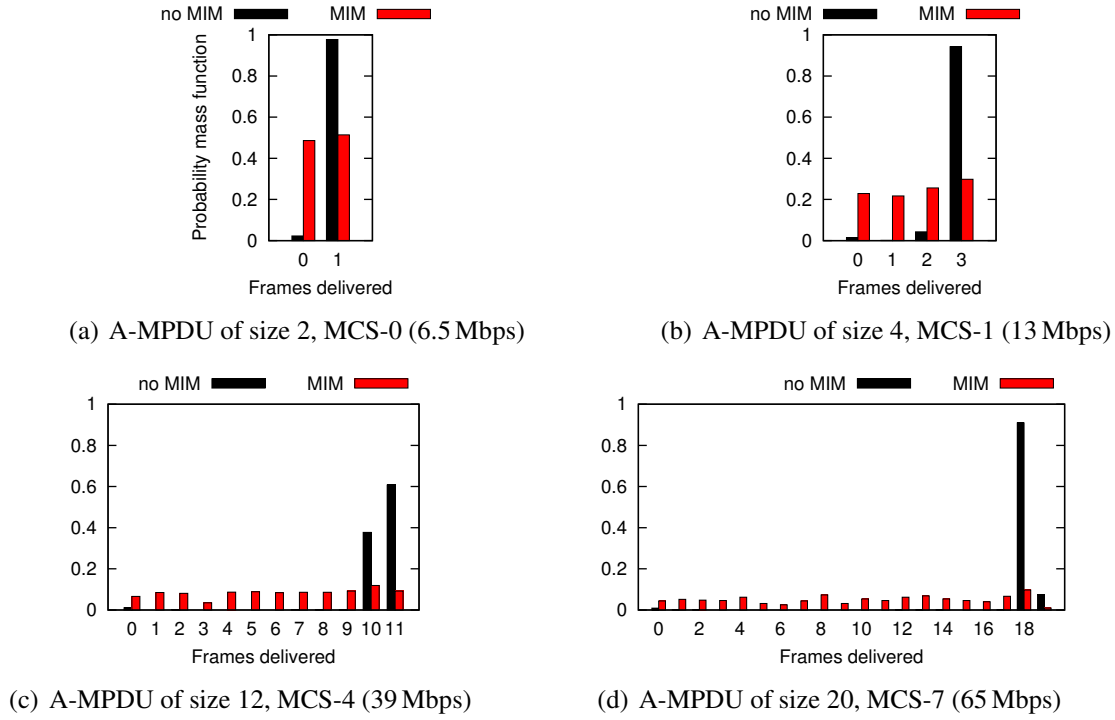
A-MPDU was close to 1 when there was no interference.

We modified the ath9k driver to allow the sender to set the data rate and the number of frames aggregated for each A-MPDU. In our experiments, each frame in an A-MPDU has a 1,500-byte payload, and the number of frames aggregated into an A-MPDU was chosen to achieve a total air time duration of about 3.8 ms, which is slightly smaller than the maximum 4 ms allowed by the ath9k driver. Table 4.1 summarizes the number of frames in an A-MPDU (A-MPDU size) according to the data rate. For the rest of the thesis, we will refer to data rate using MCS index since different MCS indices may use the same data rate (in Mbps), and there might be ambiguity if we specified the data rate in Mbps.

4.3.2 A-MPDU Size

In this section, we investigate the impact of MIM by varying the A-MPDU size. The payload of the interfering frames was set at 50 bytes, which is the size of a typical TCP-ACK packet. The interfering frames were transmitted at data rate MCS-2 (19.5 Mbps) and had a stronger signal strength of about 14 dB higher than the A-MPDU. The air time duration of an interfering frame under these settings was about 60 μs.

Figure 4.6 shows the normalized distribution of the number of frames delivered per A-MPDU for A-MPDUs of size 2, 4, 12 and 20. With MIM enabled, the probability density of the number of frames delivered per A-MPDU is evenly distributed for all the graphs. This is because a collision in one frame of an A-MPDU causes all subsequent frames to be lost, and the interfering frame collides uniformly across the A-MPDU (see



(a) A-MPDU of size 2, MCS-0 (6.5 Mbps)

(b) A-MPDU of size 4, MCS-1 (13 Mbps)

(c) A-MPDU of size 12, MCS-4 (39 Mbps)

(d) A-MPDU of size 20, MCS-7 (65 Mbps)

Figure 4.6: Distribution of the number of frames delivered per A-MPDU.

Figure 4.5). With MIM disabled, there is a small non-zero probability that no frames were received due to the interfering frame colliding with the PLCP header of the A-MPDU, causing a failure to decode the entire A-MPDU. The remaining probability density is skewed towards losing only one frame for A-MPDUs of size 2 and 4 (Figures 4.6(a) and 4.6(b)), and is approximately two frames as the A-MPDUs size increases (Figures 4.6(c) and 4.6(d)).

Intuitively, it seems likely that the two-frame loss is caused by the interfering frame "straddling" two consecutive frames when colliding with an A-MPDU. For A-MPDU size of 20, each frame in an A-MPDU takes about 190 μs air time, as compared with the 60 μs of interfering frame. Theoretically, the interfering frame would straddle two frames in an A-MPDU at a probability of 0.3. However, in Figure 4.6(d), we see that the majority of A-MPDU lose two frames. We suspect that the reason for this higher than expected loss ratio is because the impact of an interfering frame is more than its total airtime and it takes receiver some additional amount of time to re-adjust itself to receive the original

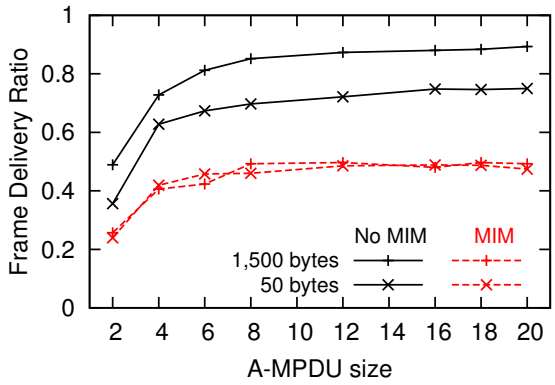


Figure 4.7: Impact of A-MPDU size with interfering frame payload of 50 and 1,500 bytes.

A-MPDU after being interrupted by a strong interfering frame. Unfortunately, we do not possess the equipment that would allow us to make the measurements required to verify this hypothesis.

We plot the average frame delivery ratio (FDR) for each A-MPDU size in Figure 4.7, and confirmed that with MIM disabled, the average FDR for interference with 50-byte payloads converges to a value of about 0.9, which is the approximate ratio of the air time duration of the interfering frame to the total A-MPDU. With MIM enabled, the FDR converges to 0.5, which is expected as the interfering frame would collide uniformly across the A-MPDU. When the payload of the interfering frame is increased to 1,500 bytes, the results are similar to that of 50-byte payload when MIM is enabled. But with MIM disabled, the average FDR now converges to a much lower value of about 0.75. This shows that with MIM disabled, the air time duration of the interfering frame will affect the average FDR of the A-MPDU.

4.3.3 Interfering Frame Air Time Matters

It turns out that the achieved frame delivery ratio for an A-MPDU depends on the air time duration of the interfering frame. To demonstrate this, we performed two experiments: in one, we increased the air time of the interfering frame by increasing its payload, while in the other, we reduced the data rate. In the first experiment, we vary the payload of the interfering frame from 1 to 1,500 bytes, keeping the data rate at MCS-0 (6.5 Mbps). In

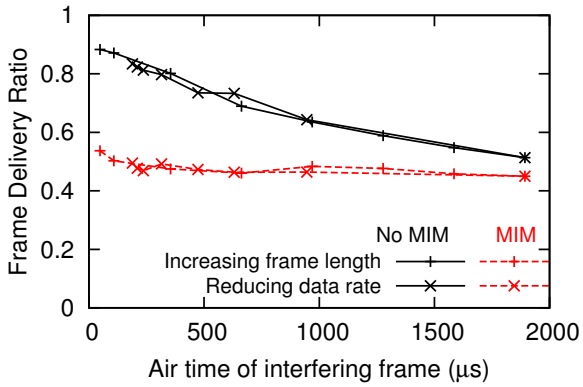


Figure 4.8: Impact of the air time of interfering frames. The A-MPDU size is 20.

the second experiment, we fixed the payload of the interfering frame at 1,500 bytes and varied the data rate of the interfering frame from MCS-7 (65 Mbps) to MCS-0 (6.5 Mbps).

Figure 4.8 shows the achieved average FDR against the air time duration of the interfering frame from both experiments. The results show that the air time duration has a direct effect on the FDR of the A-MPDU and that there is no difference between increasing the payload or reducing the data rate of the interfering frame. With MIM enabled, the FDR remains at a stable value just below 0.5 even as the interfering frame air time increases. With MIM disabled, the FDR decreases as the interfering frame air time increases.

Our results suggest that as the air time duration of the interfering frame decreases, the negative impact of enabling MIM unnecessarily increases. Therefore, to understand the impact of enabling MIM in real networks, we investigated the distribution of the air time duration of frames "in the wild." Figure 4.9 shows the distribution of the air time duration of frames captured at different locations in a university library over a 2-hour period during office hours. The results show that most of the frames have a very short air time duration. In particular, the median air time is only about 30 μs, and 90% of the frames have air time durations shorter than 300 μs. We can also see two distinct bandings with about 25% of the frames having an air time of 20 μs and 20% at 190 μs. Upon closer inspection, we found that the former 20 μs frames were MAC ACK frames and the latter frames were multicast frames arising from the Neighbor Discovery Protocol of IPv6. We

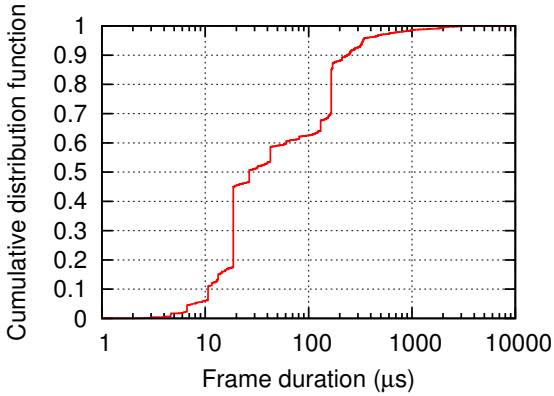


Figure 4.9: Distribution of frame air time duration in a university library.

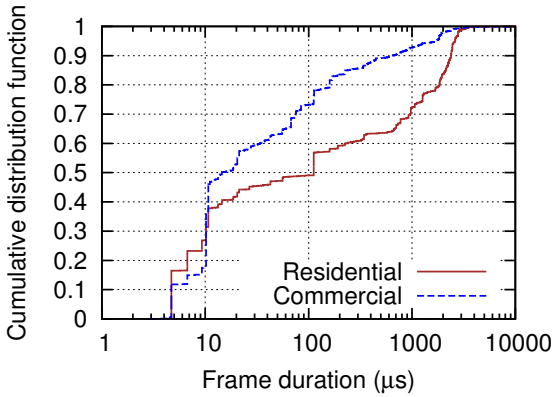


Figure 4.10: Distribution of frame air time duration in a residential area and a commercial mall.

also measured similar frame air time duration in a residential area and a commercial mall, as shown in Figure 4.10. We can see that the median air time duration is also quite short, which is no more than 100 μs. There is a banding at around 10 μs, which is due to MAC ACK frames. Note that this banding is smaller than the one in Figure 4.9 because we found most MAC ACK frames are sent at a higher data rate of 12 Mbps.

4.3.4 Impact of Received Signal Strength Differences

In the previous experiments, the received signal strength of interference was set to about 14 dB higher than that of the A-MPDU signal. In this section, we investigate and characterize the impact that the differences in received signal strength (RSS) have on the MIM mechanism. The transmission power of A-MPDU sender was held constant while the `iwconfig` command was used to change the transmission power of the interfering node

to set the RSS difference.

Figure 4.11 shows the average FDR against the RSS difference with A-MPDUs of size 6 and 20, and interfering frames with 1,500-byte payload sent at MCS-7 (65 Mbps) and MCS-15 (130 Mbps). With MIM disabled, the FDR decreases slightly as the RSS difference increases. This effect is more pronounced when the air time duration of the interfering frame is longer (i.e., using MCS-7). Like the results for interference highlighted in Section 4.3.2, we suspect that it takes time for the receiver to re-adjust itself to receive the original A-MPDU after being interrupted by an interfering frame. Again, we do not currently have the means to verify this and leave this as future work.

With MIM enabled, the FDR drops steeply to below 0.5 when the RSS difference is larger than 10 dB, regardless of the size of the A-MPDU. This shows enabling MIM is effective only when the RSS difference is larger than 10 dB and corroborates an earlier study on Atheros 802.11a adapters [52].

In addition, the 10 dB threshold of RSS difference remains the same when the interfering frame was sent at MCS-7 and MCS-15. Note that the interfering frame with MCS-15 uses two spatial streams for its payload, as compared with one for MCS-7. In our experiments, the interfering node was in direct line-of-sight of the receiver, and the receiver could hardly decode the payload of the interfering frames sent at MCS-15 [27]. However, the preamble of the interfering frame is still sent in a single spatial stream even when the payload is sent across two. Thus, in terms of the capability to knock out A-MPDU, there is no difference when the interfering node uses multiple spatial streams for its frame payload.

4.3.5 Channel Bonding

The 802.11n standard specifies an extended channel bandwidth of 40 MHz (which is twice that of the conventional 20 MHz bandwidth) to increase network capacity by doubling the data rate. This use of an extended channel bandwidth is known as *channel bonding* [27]. In this section, we investigate the impact of the MIM mechanism when channel bonding

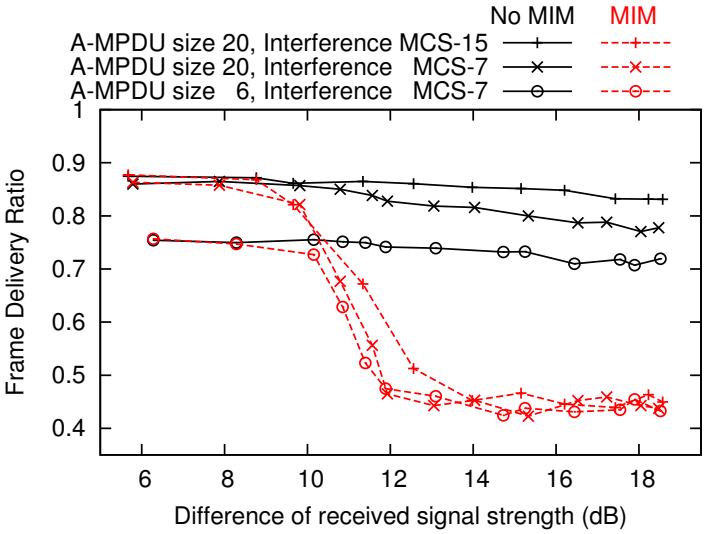


Figure 4.11: Impact of received signal strength. The payload of interfering frame is 1500-byte.

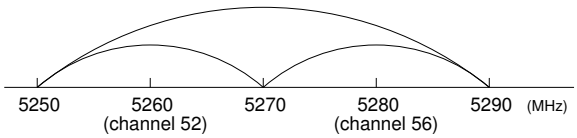


Figure 4.12: The two channels used in the channel bonding experiments.

is used.

In this experiment, the 40 MHz-wide channel used comprises channel 52 and channel 56 (see Figure 4.12). We consider all the possible permutations of channel widths for the three nodes as shown in Table 4.2. Note that for the receiver to receive the sender's signal, the receiver channel cannot be narrower than the sender. The data rate MCS-7 (65 Mbps) was used for both the sender and interfering node, and the payload of the interfering frame was 50 bytes.

In Figure 4.13, we plot the resulting frame delivery ratio against RSS difference for the cases listed in Table 4.2. Our results show that there is no apparent difference among the different cases. For example, case-1 shows that the interfering frames sent at bonded channel can still knock out A-MPDU when the sender and receiver use 20 MHz-wide channel. Similarly, case-4 shows that when the interfering node uses a 20 MHz-wide channel, it can disrupt the sender and receiver on a bonded channel.

Table 4.2: Combinations of channel width (MHz) used in the channel bonding experiments. The 20 MHz channel refers to channel 56.

	Sender	Receiver	Interfering Node
Case-1	20	20	40
Case-2	20	40	20
Case-3	20	40	40
Case-4	40	40	20
Case-5	40	40	40

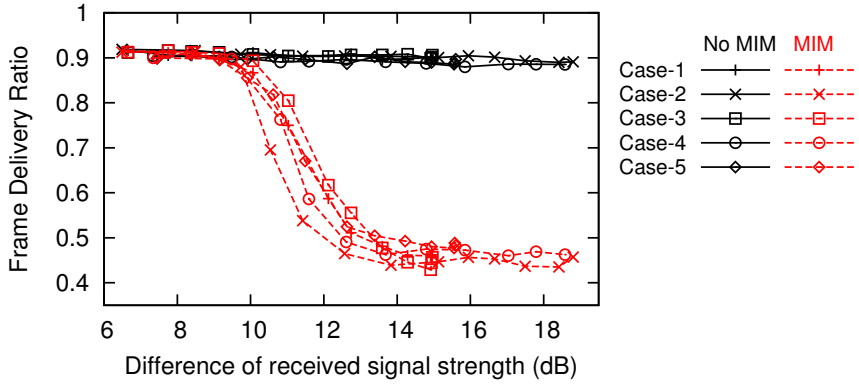


Figure 4.13: Effect of MIM with channel bonding. The payload of interfering frame is 50-byte.

4.3.6 Adjacent-channel Interference

In the experiments described in the previous sections, all the interference occurred in the same (or overlapped) channel as the sender. In this section, we investigate the impact of adjacent-channel interference when the interfering frame is sent on a channel adjacent to the A-MPDU. Adjacent-channel interference is common as the spectral mask in the radio hardware is not perfect. In this experiment, the sender is set at channel 56 and the interfering node is set at channel 52 (see Figure 4.12). We investigated both cases where the receiver listens only on channel 56 and where it uses channel bonding and listens to both channels 52 and 56.

One minor complication in our experiment is that the original method of sending a poll message to clock the interfering frames no longer works as the sender and interfering node are now listening on different channels. Thus, we employed a different approach to generate the interference, by having the interfering node continuously broadcast an

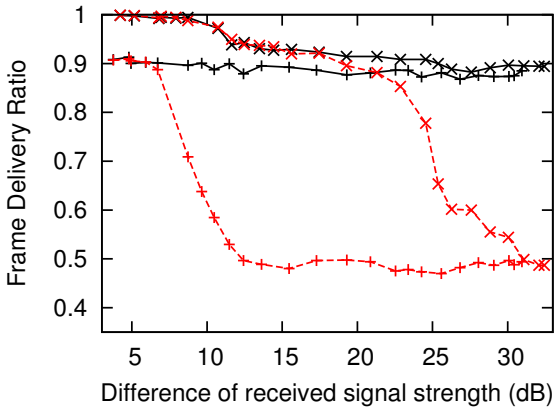


Figure 4.14: Receiver's channel width is 20 MHz.

50-byte interfering frame at MCS-7 (65 Mbps), at 4 ms intervals. At the same time, the sender sends the A-MPDU at an interval that is uniformly distributed from zero to 100 ms. The carrier sensing mechanism at both the sender and interfering node was disabled. In this way, we ensure that the A-MPDU (with air time duration of about 3.8 ms) will collide with a single interfering frame for most of time. This technique cannot guarantee that the A-MPDU will always arrive before the interfering frame. However, since the air time duration of A-MPDU is much longer than that of interfering frame (several tens of μs), the probability that the A-MPDU will be received later than interfering frame is very small.

Figure 4.14 shows the FDR when the channel width of receiver is set to 20 MHz. For comparison, we also included the scenario where the sender and interfering node are on the same channel using the same experimental setup. In order to get a larger RSS difference, we also reduced the transmission power of the sender, while ensuring that the FDR of the A-MPDU remains close to 1 when there is no interference.

When MIM is disabled and the sender and interfering node are on adjacent channels, the FDR is close to 1 when the RSS difference is small but gradually decreases when the RSS difference increases. This is expected as adjacent-channel interference becomes more pronounced at larger RSS differences. As the RSS difference increases further, the FDR appears to converge to the case where the sender and interfering node are on the same channel.

Next, we examine the case where MIM is enabled and the sender and interfering node

are on adjacent channels. At RSS differences smaller than 20 dB, the FDR is similar to when MIM is disabled as the leaked signal from the adjacent channel is weak. However, from an RSS difference of about 20 dB onwards, the receiver begins to pick up and decode the preamble of the interfering frame and kick out the A-MPDU. Thus the FDR begins to decrease sharply until it becomes equivalent to the case where the sender and interfering node are on the same channel when the RSS difference is greater that 30 dB. This shows that when the interfering node has a sufficiently higher signal strength than the sender, MIM can still take effect and be detrimental even though the sender and interfering node are on adjacent channels. The exact threshold at which this happens is likely to be dependent on the spectral mask of the adapter hardware.

Figure 4.15 shows the FDR when the receiver uses channel bonding and listens to both channels. With MIM disabled, the FDR is similar to the case in Figure 4.14 where the receiver only listens on one channel, as the interfering frame appears to be noise to the receiver. On the other hand, with MIM enabled and RSS difference above 10 dB, the receiver can clearly pick up the interfering frame and abandon the reception of A-MPDU, resulting in a sharp decline of the FDR. In other words, the receiver behaves similarly to when the sender and interfering node are on the same channel. Therefore, we can see that listening on the 40 MHz-wide channel could cause the receiver to be more susceptible to MIM effect due to adjacent-channel interference, as compared with the one listening on a 20 MHz channel.

4.4 Adaptive MIM

Our results clearly suggest that the MIM mechanism can be helpful or detrimental under different conditions. The natural question is whether it is possible to adaptively toggle MIM on and off in a manner such that MIM is turned on only when it is helpful and not when it is harmful. Our key insight is that it is possible to determine whether MIM is helpful by observing the interference patterns on the received A-MPDU frames as illustrated

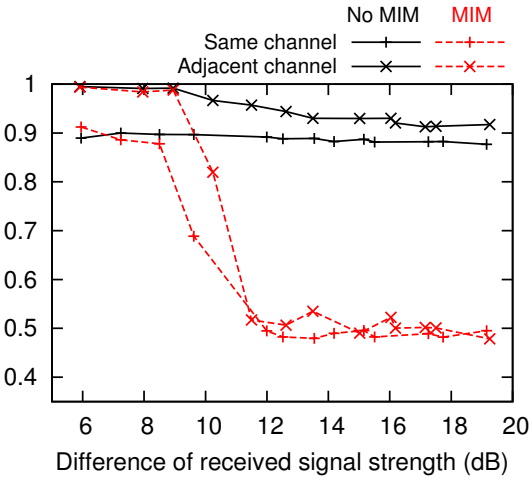


Figure 4.15: Receiver's channel width is 40 MHz.

previously in Figure 4.1. Thus, we can simply count and compare the number of occurrences where MIM was helpful (see Figure 4.1(a)) versus where MIM was detrimental (see Figure 4.1(b)) to decide whether and when to enable or disable MIM.

In order to count the occurrences of each case, we modified the ath9k driver to retain the partial frame that is knocked out when the MIM mechanism kicks in. Specifically, the modified driver no longer ignores frames that have the physical error flags `OFDM-RESTART` or `CCK-RESTART` set as these error flags indicate that the frame was knocked out by a stronger frame. If the knocked-out frame is not addressed to the receiver and is immediately followed by one addressed to the receiver, then we consider this an occurrence where MIM is helpful. On the other hand, if a frame addressed to the receiver is knocked out by a frame that is not addressed to the receiver, it is an occurrence where MIM was detrimental.

Figure 4.16 shows the state diagram of our proposed method to adaptively enable and disable MIM. We count the number of helpful (n_{good}) and detrimental (n_{bad}) occurrences in each epoch of 1 s. MIM will be disabled when $n_{good} < n_{bad}$ for a given epoch. A minor complication is that the adapter can only count and update n_{good} and n_{bad} when MIM is enabled. Thus, we need to periodically enable MIM to resume counting after MIM is disabled. We adopt a simple exponential backoff approach by enabling MIM after k epochs, and doubling k every time n_{good} turns out to remain smaller than n_{bad} when we

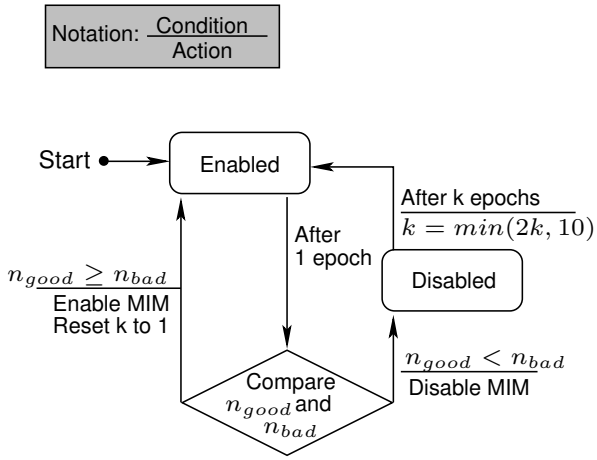


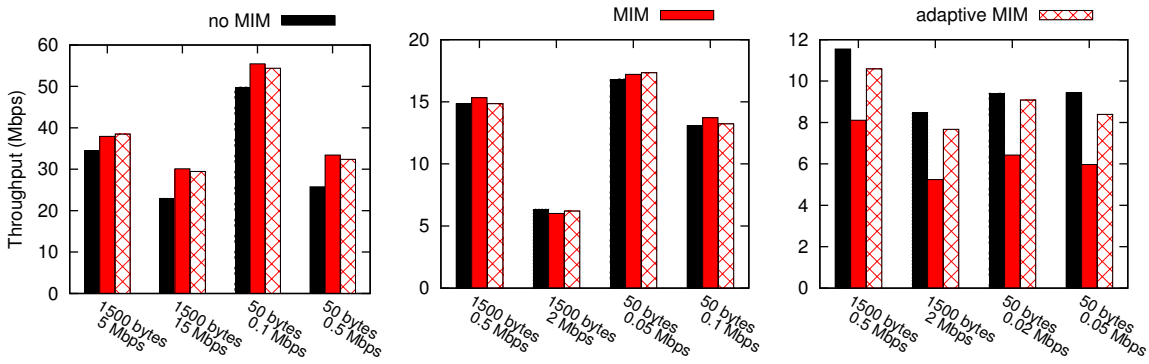
Figure 4.16: The state diagram of the proposed adaptive MIM method.

turn MIM on to do the counting. The initial value of k is set to 1 and the maximum value is 10. k will also be reset to 1 whenever $n_{good} \geq n_{bad}$. In this way, we try to strike a balance between effectiveness and responsiveness for the proposed method.

In Figure 4.17, we compare our adaptive MIM method to cases where MIM is always enabled or disabled for each of the scenarios described in Section 4.2. In addition, we also tested with the payload of interfering frame set to 50 bytes. Just like in Section 4.2, each test was run for 30 s. Our results show that under situations where MIM is helpful (see Figure 4.17(a)), our adaptive MIM method achieves similar throughput as the case where MIM is always enabled. At Position-2 where MIM has no effect, the throughput results are similar for all cases. In situations where MIM is detrimental, our adaptive MIM method achieves slightly lower throughput than the one with MIM disabled. This is because under such scenarios, our method has to occasionally enable MIM to update the counters n_{good} and n_{bad}, which incurs some overhead. We feel that this is an acceptable overhead.

4.5 Summary

In this chapter, we made an important observation that the MIM mechanism in modern 802.11 hardware could potentially degrade the performance of A-MPDU reception when



(a) Helpful (Position-1) (b) No effect (Position-2) (c) Detrimental (Position-3)

Figure 4.17: Effect of the proposed adaptive MIM method. The x-axis refers to the payload size and throughput of the interfering frames.

there is strong interference. With a comprehensive set of experiments, we characterized how turning MIM on and off would affect the reception of A-MPDU under various parameters and scenarios. We also investigated the impact of MIM in the presence of channel bonding and adjacent-channel interference. We showed that it is possible to avoid the potentially harmful effects of MIM by adaptively turning MIM on and off using a simple mechanism that monitors the interference pattern on received A-MPDUs.

Chapter 5

Mitigating ACK Interference with MinPACK

In this chapter, we describe MinPACK, our proposed solution to mitigate the interference due to the MAC Acknowledgment frames in 802.11 WLAN, thereby improving the overall efficiency.

As discussed in Chapter 1, interference is one of the most critical factors that limit the efficiency of 802.11 networks due to the shared nature of the wireless medium. Existing work on interference mitigation have focused almost exclusively on minimizing the interference from *MAC data frames* by regulating the transmission power of access points (APs) [11, 58, 64] and by scheduling the transmission time carefully [57, 71]. In our recent measurement study of corporate Wi-Fi AP deployments, we found that 802.11 MAC Acknowledgment frames can sometimes cause significant interference, and as illustrated in Figure 5.1, can effectively extend the interference range of an AP.

Because the main bulk of the access network traffic is downstream from the APs [56, 29], clients can generate a large number of ACK frames. Our analysis of the traces obtained from real AP density measurements suggests that the likelihood that a client will experience interference from MAC ACK frames can range from 25% to 50% in practical Wi-Fi deployments. We found that for a campus 802.11n AP deployment, ACK interfer-

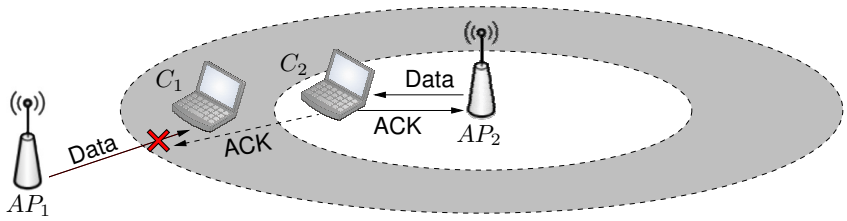


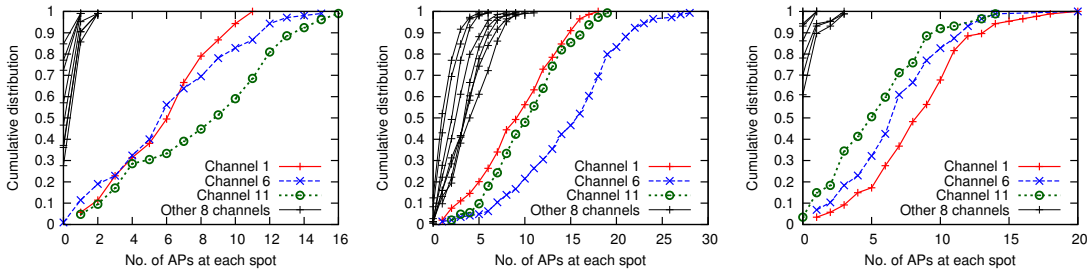
Figure 5.1: Interference between adjacent Wi-Fi hotspots. MAC ACK frames of clients effectively extend the interference range of a hotspot (i.e., clients in the gray ring like C_1 will not experience interference from AP_2's data frames but can experience interference from the MAC ACK frames of AP_2's clients like C_2).

ence could cause a reduction of several fold when two interfering 802.11n flows compete with each other. Furthermore, the ACK interference from an 802.11a/b/g flow can cause starvation for other 802.11n flows.

Power control of MAC data frames has been well studied in the literature [11, 58, 64]. However, MAC ACK frames are fundamentally different from MAC data frames, because they are generated automatically and do not provide any feedback or consider the channel state when transmitted. Thus, it is not straightforward to directly apply available power control algorithms on MAC ACK frames. In addition, due to its small frame size and low data rate, we found that we can reduce the power of ACK frame much more than that for data frame without adverse effect. To address the MAC ACK interference problem, we propose a simple ACK power control algorithms for clients, called MinPACK, and show through extensive experiments that MinPACK is able to achieve a median throughput improvement of 31% for 802.11a/b/g clients, and can potentially double the throughput for 802.11n clients.

5.1 Motivation

In this section, we first present a measurement study on the density of APs in densely populated metropolitan areas. Next, we present a network model to estimate the likelihood of ACK interference in modern Wi-Fi AP deployments. Finally, we show with experiments how ACK interference can cause significant performance degradation and that reducing the transmission power for MAC ACK frames can mitigate this problem. For the rest of



(a) Commercial (b) Residential (c) Campus

Figure 5.2: Number of APs observed during warwalking.

this thesis, we will use "data sender" and "ACK receiver" interchangeably, and also "ACK sender" and "data receiver" interchangeably.

5.1.1 Measurement Study on AP Density

Prior work on AP density measurements [11] relied on the estimated physical location of APs (obtained from wardriving) and assumed a fixed interference range (e.g., 50 m) to estimate AP density. This method is not accurate as the estimated AP locations from wardriving are shown to have a median error of 32 m [46]. To achieve a higher accuracy for AP density measurements, we conducted an equivalent experiment, but on foot and we call this "warwalking". We surveyed three classes of densely populated metropolitan areas: (i) a high-density residential area, (ii) a shopping mall and (iii) a university campus. Most APs in the residential area are unmanaged private hotspots while the APs on the university campus are dominated by an enterprise Wi-Fi network. The shopping mall has mix of both private hotspots and an enterprise deployment.

Throughout the warwalking exercise, we set a sniffer to repeatedly scan all available Wi-Fi channels (1 to 11), with a scan duration of 1 s for each channel. In each area, the path was about 1 km long and the walking speed was approximately 1 m/s. Since our walking speed is relatively slow compared to the range of Wi-Fi, we considered each 1 s spent in a channel to be a single "spot". We estimated the number of APs observed at each spot by counting the number of unique MAC addresses (or BSSIDs) that broadcast a beacon frame.

Precautions were taken to minimize potential measurement errors. First, as the sniffer might be able to receive beacon frames transmitted in adjacent channels, we examined the channel number embedded in the beacon frames to determine the channel an AP was on. Second, as some manufacturers support virtual APs, we consider a batch of MAC addresses to belong to the same physical AP if they do not differ more than k from each other, where is k is the largest number of consecutive MAC addresses observed in the trace. We found k to be 5 in our traces.

Figure 5.2 shows the cumulative distribution of the AP count for the three scenarios. As expected, the three orthogonal channels (1, 6 and 11) had the highest frequency in all the scenarios. The residential area had the densest deployment, with a median AP count of 15 at channel 6. In other words, an AP at channel 6 in residential area is expected to have 14 neighboring APs. Although the APs in campus area are mainly from an enterprise network, we found that the median AP count for channel 1 was 8, i.e., the AP density was still surprisingly high.

5.1.2 Modeling MAC ACK Interference

In this section, we model and estimate the likelihood that a data frame to a client device will experience ACK interference from other clients connected to neighboring APs. Because whether ACK interference occurs will depend on the positions of APs and clients, we first develop a simple model for the area from which the clients of neighboring cells can cause ACK interference (we call this the "interference region"). Next, we estimate the average number of clients that lie within the interference region and the probability of ACK interference in terms of the rate that clients receive data frames.

Modeling the Interference Region. For ACK interference to occur, adjacent APs cannot be too close, or too far from each other. Consider a simple case of two adjacent APs, AP_1 and AP_2 with associated clients C_1 and C_2 respectively. We assume that the APs and clients have the same transmission power and the channel is symmetric and they have the same transmission range R_T. We also assume they have the same carrier sense

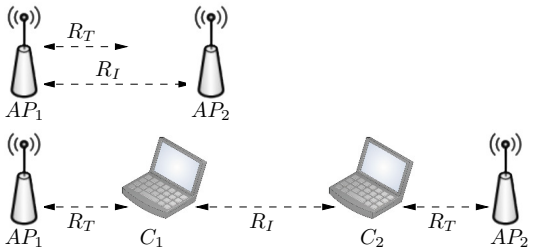


Figure 5.3: Illustration of the minimum and maximum distance between two adjacent APs for ACK interference to occur.

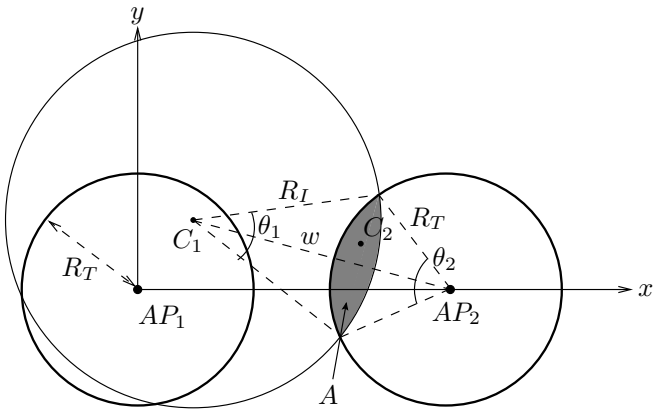


Figure 5.4: Network model used in the analysis.

and interference range R_I, s.t. $R_I \geq R_T$. Figure 5.3 illustrates the minimum and maximum distance between the two APs for ACK interference to occur. If the two APs are within R_I of each other, CSMA will prevent simultaneous transmission and hence there will be no ACK interference. On the other hand, if two APs are beyond $R_I + 2R_T$ from each other, ACK interference will not occur either. Hence, for ACK interference to occur, the two APs must be within a distance between R_I and $R_I + 2R_T$, and the clients are within the transmission range R_T of their associated APs.

We set up a coordinate system centered at AP_1 with AP_2 at $(u,0)$ and C_1 at (x,y), where $R_I \leq u \leq R_I + 2R_T$, as shown in Figure 5.4. The two small circles indicate the transmission range of the APs. The large circle indicates the area within which another client's transmissions will interfere with C_1. Thus, the shaded area represents the interference region from which a client C_2 that is associated with AP_2 can cause ACK interference to C_1.

Average number of clients in the interference region. If we assume that the location of the clients for an AP is uniformly distributed, then $p(u,x,y)$ can be modeled as

$$p(u,x,y) = \frac{A}{\pi {R_T}^2} \tag{5.1}$$

where $p(u,x,y)$ is the probability that C_2 lies in the interference region, for a given u, x and y and where A is the area of the shaded intersection. Since A can be computed by adding the two fan areas and then subtracting the two triangular areas, we can express A as

$$A = \frac{{R_I}^2}{2}(\theta_1 - \sin\theta_1) + \frac{{R_T}^2}{2}(\theta_2 - \sin\theta_2) \tag{5.2}$$

where θ_1 and θ_2 are the internal angles formed by the sectors at C_1 and AP_2. By setting w to be the distance between C_1 and AP_2, we can express A in terms of u, x and y by computing θ_1 and θ_2 using the Cosine rule for a triangle with sides R_T, R_I, and w, where $w = \sqrt{(u-x)^2 + y^2}$.

Suppose AP_2 has m clients, uniformly distributed within its transmission range, we can model the number of clients in A as a binomial distribution $B(m, p(u,x,y))$. Thus, the average number of clients in A will be $m \times p(u,x,y)$. The average number of clients in the interference region $\bar{m}$, over all values of u, x and y can then be expressed as:

$$\bar{m} = \iint_S \int_{u=R_I}^{R_I+2R_T} [m \times p(u,x,y)]\, du\, dx\, dy \tag{5.3}$$

where S is the area of the small circle centered at AP_1.

Likelihood of ACK interference. Suppose in Figure 5.4, it takes C_1 time t to receive a data frame from AP_1. If within this time t, a client C_2 in area A finishes receiving a data frame from AP_2, it will respond with an ACK which will interfere with C_1. If we assume that the rate at which C_2 receives data frames follows a Poisson distribution with rate λ, the rate of ACK transmissions from C_2 will also follow the same Poisson distribution. Let z denote the duration from the time C_1 starts receiving a data frame to the moment C_2

transmits the next ACK. z will follow an exponential distribution with rate λ. Since C_2's ACK will collide C_1's data frame if z is smaller than t, the probability of ACK interference due to C_2 (or a client in the interference region) can be expressed as:

$$P(z < t) = 1 - e^{-\lambda t} \tag{5.4}$$

Suppose there are on average n APs within the range of R_I and $R_I + 2R_T$ from AP_1, and each AP has m clients. Then, the probability that ACK interference is experienced at C_1 is:

$$P_{n,m} = 1 - e^{-nm\bar{m}\lambda t} \tag{5.5}$$

Because the transmissions of data frames take time, λ is limited by $\lambda < 1/t$, where t is the transmission time of a data frame. Assuming that the data frames sent by all APs are evenly distributed among their clients, λ is further bound by $m\lambda < 1/t$. Because of carrier sense between APs, λ is also constrained by the number of APs within carrier sense range, as denoted by N_I. To find N_I, we first compute the density of AP. Since there are n APs within the range of R_I and $R_I + 2R_T$ from our AP, the AP density d can be expressed as:

$$d = \frac{n}{\pi(R_I + 2R_T)^2 - \pi {R_I}^2} = \frac{n}{4\pi R_T(R_I + R_T)} \tag{5.6}$$

Thus, the average number of APs within interference range of any AP is simply:

$$n_I = d \times \pi {R_I}^2 = \frac{n{R_I}^2}{4R_T(R_I + R_T)} \tag{5.7}$$

Hence, the final upper bound on λ is $m\lambda(n_I + 1) \leq 1/t$.

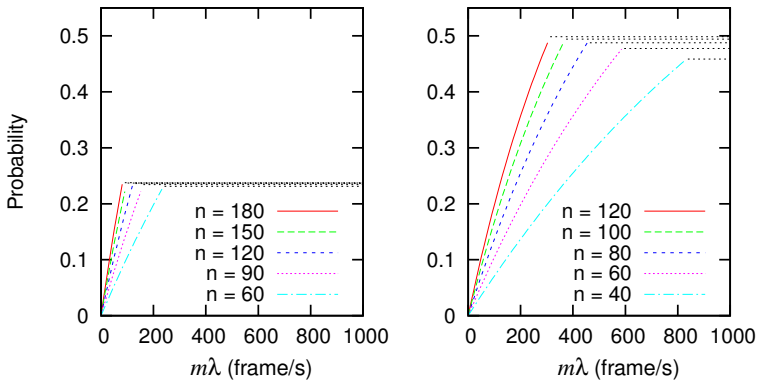
Assuming data frames are 1,500 bytes in size and sent at a typical data rate of 54 Mbps, the transmission time works out to be $t = 0.0002$ s. Under the `ns-2` model, the Wi-Fi sender implements carrier sense by detecting a high energy transmission on the channel. Hence, the interference range is typically set to twice the transmission range,

i.e., $R_I = 2R_T$. However, we observed that our Atheros Wi-Fi adapter considers the channel to be busy only when it detects a Wi-Fi preamble. Thus, in practice, we find that the carrier sense range is equal to the transmission range, i.e., $R_I = R_T$. For completeness, we show both cases in Figure 5.5, where we plot the probability of ACK interference against varying values of $m\lambda$ and n. We believe that most practical networks would lie somewhere between these two scenarios.

In our warwalking experiment, the median number of APs observed ranges from 5 to 15. Since we could only observe APs within our transmission range (R_T), we cannot directly observe n, the number of APs between R_I and $R_I + 2R_T$. As such, we assume that the APs are uniformly distributed and estimate the density of APs within R_T to derive the value of n. For $R_I = 2R_T$, the area enclosing the APs is 12 times that of R_T. Thus, n is estimated to be in the range from 60 to 180. For the other case where $R_I = R_T$, the area enclosing the APs is 8 times that of R_T, and thus n is estimated to be in the range from 40 to 120.

From our results in Figure 5.5, we observed that there is an upper bound on the probability that seems to be largely independent of λ and n, i.e., while the probability of ACK interference generally increases with the number of APs and also the sending rate, there is a cap on the probability of ACK interference. This is due to the bound we imposed on the sending rate to model the effects of CSMA. The cap on the ACK interference probability is dependent on the interference range. When the interference and carrier sense range is large, CSMA will prevent the APs from sending at a higher rate. As such, the average probability of ACK interference is bound at about 0.25 and 0.5 for the scenarios $R_I = 2R_T$ and for $R_I = R_T$ respectively.

We note that the maximum sending rate for the 802.11x data rate of 54 Mbps is about 4,700 frame/s. Figure 5.5 suggests that it is therefore quite plausible for a single busy AP to create sufficient traffic to reach the upper bound of $m\lambda$. Thus, we can expect an ACK interference probability of between 25% and 50% in 802.11x networks.



(a) $R_I = 2R_T$ (b) $R_I = R_T$

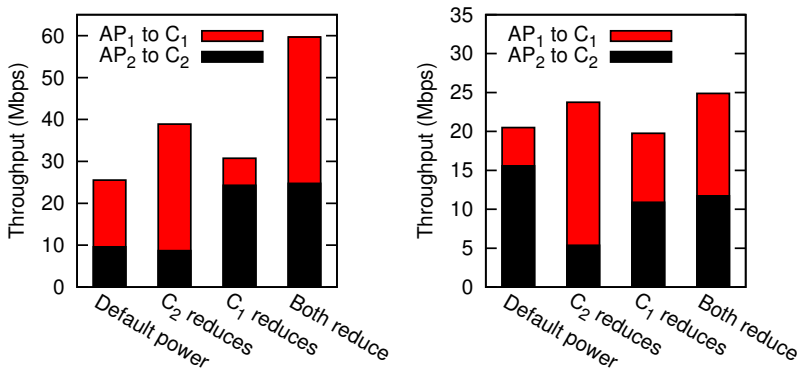
Figure 5.5: Computed probability of ACK interference in our model.

5.1.3 Impact of MAC ACK Interference

In order to understand the impact of MAC ACK interference in practical Wi-Fi networks, we conducted a set of experiments for the scenario shown in Figure 5.1 using our campus WLAN network, which comprises of Cisco 1140 series APs. We have two clients: Advantech system board with Compex WLE350NX 3x3 802.11n adapter, and ALIX system board with Compex WLM54AG 802.11a/b/g adapter. The traffic sources were Linux machines in our lab. The channels of the APs are dynamically assigned by the backend Cisco wireless LAN controller, and the two APs' channels were fixed at channel 149 for our experiments. RTS/CTS is not enforced by the APs.

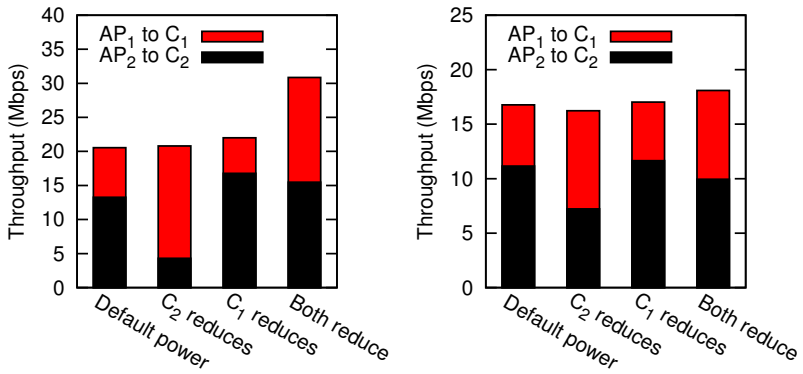
For each set of experiments, we compared the throughput of both clients at the default ACK power of 20 dBm against that when one or both reduce their ACK power to 10 dBm. We also placed sniffers next to each client to count the number of ACK frames transmitted. Figure 5.6 shows the results with both clients using 802.11n, Figure 5.7 shows the results with both clients using 802.11a, and Figure 5.8 shows the results with one client using 802.11a and the other using 802.11n.

ACK power control mitigates ACK interference. As expected, reducing the ACK power of one client improves the UDP throughput of the other client. For example, in Figure 5.6(a), C_1's throughput increased from 16 Mbps to 30 Mbps when C_2 reduced its ACK power. When both clients reduced their ACK power, the overall throughput improved



(a) UDP (b) TCP

Figure 5.6: Impact of ACK interference with two 802.11n links.

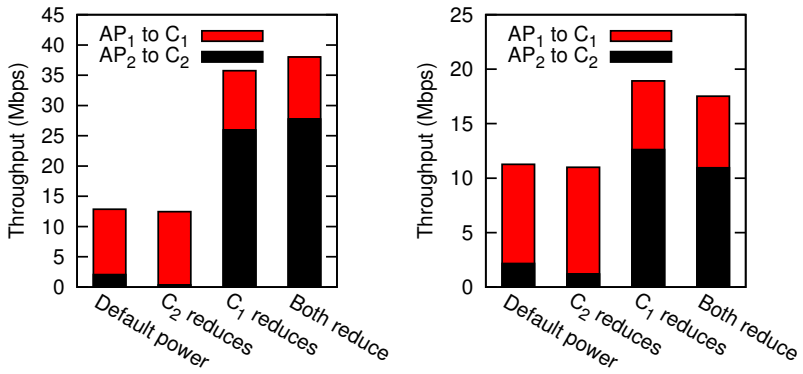


(a) UDP (b) TCP

Figure 5.7: Impact of ACK interference with two 802.11a links.

by 134%, compared to that when both were using the default power. We also observed similar improvements with both clients using 802.11a as shown in Figure 5.7(a).

ACK power control can improve local throughput. One counter-intuitive observation in our experiments is that a client can sometimes achieve a higher throughput by reducing its own ACK power. This can be seen in Figure 5.6(a) where C_1's throughput was increased when C_2 reduced its ACK power. Surprisingly, we found that C_1 could then achieve a further increase in throughput from 30 Mbps to 35 Mbps by reducing its own ACK power. The 802.11n standard allows a data sender to transmit multiple frames in a single Aggregate MPDU (A-MPDU) and to also selectively retransmit any lost frames in any order, as long as their sequence numbers lie within a transmission window of 64



(a) UDP (b) TCP

Figure 5.8: Impact of ACK interference with 11a link (AP_1 to C_1) and 11n link (AP_2 to C_2).

frames [7]. We suspect that the reason for this is that when only C_2 reduced its ACK power, C_1's ACK interference on C_2 was so significant that AP_2 had to use a low data rate that does not support A-MPDU. Thus, C_2 would reply with one ACK frame for every data frame it received from AP_2. However, once C_1 reduced its ACK power, C_2 experienced less ACK interference and AP_2 can then switch to a higher data rate that supports A-MPDU. With A-MPDU enabled, C_2 can send one Block ACK in response to several data frames, instead of sending one ACK for each frame. We found that after C_1 reduced its ACK power, C_2 sent 58% fewer ACK frames compared to when C_1 was using the default ACK power.

ACK power control improves TCP fairness. Figure 5.6(b) shows C_1 having a disproportionately small throughput as compared to C_2 when both clients were using the default ACK power. When both clients reduced their ACK power, they both achieved similar throughput and the overall throughput improved by almost 25%. (Note that we only reduced the power of MAC ACK frames but not the power of TCP ACK packets which are encapsulated as data frames.) One reason is because TCP traffic is very sensitive to packet loss, and the flow that incurred more interference (i.e., C_1) could not increase its congestion window. In other words, when two TCP flows interfere with each other, fairness is often compromised.

ACK power control prevents 802.11n from starvation. In our experiment where

C_1 was using 802.11a and C_2 802.11n, C_2 was nearly starved (see Figure 5.8(a)). We suspect it is because 802.11n is more vulnerable to interference than 802.11a, just like how 802.11g could be less robust against interference than 802.11b [40]. When C_1's ACK power was reduced, C_2's throughput greatly improved regardless of C_2's ACK power level. Similar results can be observed for TCP traffic, as shown in Figure 5.8(b). In other words, by reducing ACK power, we are able to prevent an 802.11n client from being overwhelmed by an 802.11a client.

5.2 802.11x MAC ACK Power Control

While reducing ACK power is helpful, ACK frames can be lost if we overdo it and the data sender will then wrongly interpret the loss as a loss of the data frame. This will cause the data frame to be retransmitted and in the worst case, possibly also cause the data rate to be reduced, resulting in a significant degradation in the throughput. The challenge therefore, is to reduce the ACK power without causing unnecessary ACK frame losses.

In this section, we describe the MinPACK algorithm. MinPACK is an acronym for *Minimum Power for ACK frames*. The basic idea of MinPACK is to gradually reduce the transmission power for the ACK frames until the point just before ACK frame losses will occur. This can be achieved if the ACK sender can accurately estimate the delivery success rate of its ACK frames.

We can try to estimate the success rate through one of two ways. A straightforward approach is to have cooperative feedback from the data sender (i.e., the ACK receiver) and a more indirect approach is to estimate the rate through passive observations. While the latter method is less accurate, we found that it is able to achieve estimation results that are comparable to the former *without requiring any modification to the 802.11 standard*. This thereby ensures compatibility with existing commercial 802.11 adapters and allows for immediate deployment in existing Wi-Fi networks.

5.2.1 Cooperative Feedback from ACK receiver

To accurately estimate the instantaneous ACK success rate (denoted by Φ), the ACK sender has to determine the number of ACK frames transmitted (denoted by n) and the number of ACK frames successfully delivered (denoted by k) in the past small time window of τ. Because ACK frames are transmitted automatically by the adapter hardware, we cannot directly count n in our implementation. Instead, since the hardware generates an ACK for every data frame successfully received, we can obtain n by counting the number of such data frames. On the other hand, the MadWiFi driver reports the status of every data frame transmitted. Thus, the ACK receiver can directly obtain k from the number of successful transmissions and send this information to the ACK sender using a small control message.

One slight complication is that both parties have to synchronize the time window τ, in which n and k are to be counted. Medium contention or the loss of control messages could cause τ on both parties to be out-of-sync. To address this problem, the range of frame sequence numbers is specified in the control messages. The ACK receiver simply counts and returns k within a specified range of frame sequence numbers.

5.2.2 Passive Estimation without Feedback

There are two drawbacks for the cooperative feedback approach. First, the control messages would incur an additional overhead on the network, and this overhead is especially high when the channel condition fluctuates rapidly, causing the frequency of control messages to increase. Second, the APs will have to be modified in order to provide clients with the feedback. If we adopt a passive approach in the estimation of Φ, clients will be able to perform ACK power control without requiring any modifications to the APs. This would make incremental deployment easier.

A data sender will repeatedly retransmit the same data frame until it receives the associated ACK frame, or until the retransmission limit is reached. Hence, by counting

the number of duplicate data frames received, the ACK sender can in principle infer the number of unsuccessful ACK frames (denoted by m). Φ can thus be estimated as $\frac{n-m}{n}$.

The limitation of passive inference is that while the ACK sender is able to accurately count n, there is a risk of under-estimating m, because of the retransmission limit. When this limit is reached, a data sender will proceed to transmit the next data frame. Thus, the ACK sender may wrongly infer that the ACK frame was successfully delivered when the limit was reached.

To address this problem, the data sender can encode a bit in each data frame to indicate whether it had received the ACK for the previous data frame. This will however require the data sender to be modified like the cooperative feedback approach and also hardware modifications which we are currently unable to implement in the available adapters. Nevertheless, we show in Section 5.3 that even without this optimization, we are able to achieve performance that is comparable to the cooperative feedback approach in practice.

5.2.3 Extension to Block ACK

The 802.11n standard allows a data sender to transmit multiple frames in a single A-MPDU, which means that instead of sending an acknowledgment for every frame, one ACK frame, called a Block ACK, is sent to acknowledge each A-MPDU. While the passive estimation method described in Section 5.2.2 is designed for conventional per-frame ACK, it is straightforward to extend it to handle Block ACK for 802.11n.

When an A-MPDU with at least one non-corrupted frame is successfully received, the hardware adapter will automatically reply with a Block ACK. As before, the ACK sender can count n by directly counting the number of such A-MPDUs received. To count m, the ACK sender has to maintain a short history of A-MPDUs received as the data sender might not immediately retransmit the unacknowledged frames. If any frame in a newly received A-MPDU matches one in the history, we can infer that the Block ACK for the corresponding A-MPDU in the history has been lost. This entire A-MPDU is then

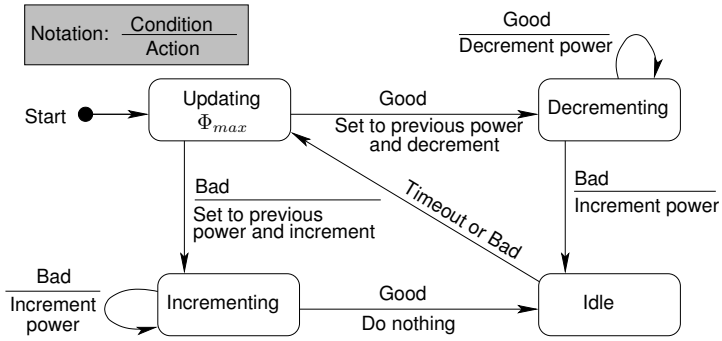


Figure 5.9: State diagram of ACK power control algorithm. Condition "Good" refers to $\Phi \geq \Phi_{max} - \delta$, whereas "Bad" for otherwise.

discarded from the history to prevent double counting as we have already determined the status of its Block ACK. As the transmission window of the data sender is 64 frames, we only need to maintain a history of A-MPDUs within this range. Our passive estimation method for Block ACK does not require any hardware modification and can be easily implemented using current 802.11n adapters.

5.2.4 MinPACK Power Control Algorithm

Ideally, the transmission power of ACK frames should be set at a level that is just slightly higher than that which will cause unnecessary ACK frame losses. To determine this level, we compare the instantaneous ACK frame delivery rate Φ to the ACK delivery rate when transmitting at maximum power Φ_{max}. Since the quality of 802.11x links varies over time, MinPACK needs to adapt to the dynamic environment by periodically adjusting the transmission power level. The key idea for MinPACK is very simple: we keep reducing the transmission power until Φ falls below Φ_{max}, and then set the power level at minimum power level required to bring Φ back to Φ_{max}. Periodically, we will probe the network to determine if the network condition has improved. Whenever Φ falls below Φ_{max}, we gradually increase the power level until Φ is close to Φ_{max}. Figure 5.9 shows the state diagram of our algorithm, which has 4 states.

The algorithm starts at the maximum transmission power, which is the default value set in the adapter. We estimate the instantaneous ACK delivery rate Φ whenever we

receive a new data frame. To keep Φ up to date, we use a sliding window of size τ = 200 ms in our implementation to compute Φ. We consider Φ to be "Good" and move to the next state if it is close to Φ_{max}, i.e., when $\Phi \geq \Phi_{max} - \delta$. δ introduces some hysteresis and we set it to 0.05 in our implementation. When $\Phi < \Phi_{max} - \delta$, we consider the condition to have turned "Bad" and perform the corresponding action and make the state transition as depicted in the state diagram. Because we use a sliding window of size τ to update Φ, we will wait for at least τ time between state transitions to ensure that we have a good estimate before we decide.

Handling Changes in the Link Quality. The link quality of 802.11x networks varies over time, e.g., due to weather condition [82]. To adapt to the change in link quality, we periodically try to update Φ_{max}. When a timeout occurs after MinPACK is in the Idle state longer than 10τ, MinPACK will attempt to update Φ_{max}. While a simple approach to determine Φ_{max} is to reset the transmission power to the default maximum value to make a measurement, doing so indiscriminately could potentially cause ACK interference to other clients. Thus, we introduce a heuristic to decide whether or not to update Φ_{max}. Every time Φ_{max} is measured, we also record the average RSSI value (R) of the data frames. To decide if we should update Φ_{max}, we compare the current average RSSI value (R_{curr}) with the recorded value (R_{prev}). If the latest RSSI is lower than previous one, we will update Φ_{max}. If the RSSI has improved, we will examine the current value of Φ. If Φ is close to 1, it means that nothing should be done since there is little room for improvement. Otherwise, we should update Φ_{max}. If the RSSI remains unchanged, we simply maintain the current Φ_{max}. In summary,

$$\begin{aligned} R_{curr} \leq R_{prev} - 2\,\text{dB} &\quad \rightarrow \text{update } \Phi_{max} \\ R_{curr} \geq R_{prev} + 2\,\text{dB} &\quad \rightarrow \text{consider if } \Phi \not\approx 1 \\ \text{Otherwise,} &\quad \rightarrow \text{do nothing.} \end{aligned}$$

We choose 2 dB as the threshold because our measurement showed that short-term RSSI reading tends to fluctuate within 2 dB.

5.3 Evaluation

5.3.1 Experiment Setup

We implemented MinPACK for OpenWRT (kernel 2.6.25) and Ubuntu (kernel 3.10.0) because we used two different hardware platforms for 802.11a/b/g and 802.11n respectively.

802.11a/b/g. The OpenWRT version was installed in an 802.11a/b/g urban wireless testbed deployed at a campus dormitory, with 20 ALIX boards (see the description in Chapter 3.3.1). Their antennas are stuck to the wall outside the buildings [82]. The adapters used are Compex adapters, featuring the Atheros AR5414 chipset. The default transmission power is 23 dBm for both data and ACK frames and the adapters were configured to run in 802.11a mode to minimize the interference from nearby campus wireless networks. The MadWiFi (version 0.9.4) driver was used with default MAC transmission limit of 10 and running in monitor mode. We adopted the popular RRAA method [81] as the default rate adaptation algorithm of data frames.

802.11n. The Ubuntu version of MinPACK was deployed on the Advantech board (1.66 GHz CPU and 1 GB RAM), equipped with an 802.11n adapter. Two such boards were used as clients to associate with and monitor the campus APs. The 802.11n adapter is also from Compex (with 20 dBm transmission power), and the `ath9k` driver was used.

We modified both the MadWiFi and `ath9k` drivers to implement fine-grained ACK power control by writing the desired value to a 6-bit ACK power control register in the hardware. Compared to the conventional method of using `iwconfig` to achieve ACK power control, our register-based approach offers the following advantages: (i) finer granularity of control (about 0.5 dBm), (ii) larger range (30 dBm for our 802.11a/b/g adapter), and (iii) option to not affect the transmission power of the data frames.

Both versions were implemented using Click modular router. In our experiments, Iperf was used as the traffic generator. We measured the throughput of MinPACK after the throughput stabilizes. It typically takes at most 15 s for MinPACK to converge from the default maximum power to the final power level. Each experiment lasted for 1 minute.

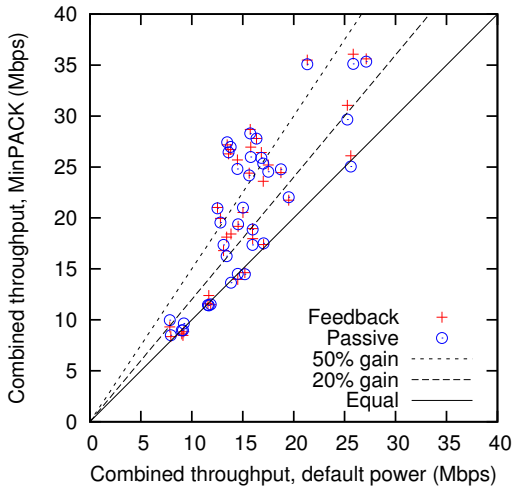


Figure 5.10: Throughput gain due to MinPACK for topologies in Figure 5.1.

5.3.2 Gain Achieved

To evaluate the effectiveness of MinPACK in mitigating ACK interference for the scenario in Figure 5.1, we selected 38 such topologies from our testbed at random and measured the throughput for concurrent UDP flows, for different scenarios. Both the active feedback and passive Φ estimation approaches were evaluated.

Figure 5.10 shows the results of the measured throughput for the two contending links. We see that MinPACK is able to improve the total throughput for most cases. With the passive estimation method, 12 of all the 38 cases (32%) saw an increase of more than 50% in total throughput, and the median throughput gain is 31%. The largest improvement was 104%. We also plot the distribution of ACK power reduction achieved by MinPACK in Figure 5.11. The median reduction in ACK interference was about 14 dB, and the maximum reduction was 28 dB. The power reduction achieved for the two different estimation methods were similar. For the rest of this thesis, the results are for MinPACK with the passive estimation method.

Figure 5.12 shows the impact of MinPACK on the fairness in the throughput achieved by the contending link pairs. For each link pair, we draw an arrow from the point of default maximum ACK power to the operating point that MinPACK converges to. Most arrows point towards the diagonal line, indicating an improvement in the fairness between the two

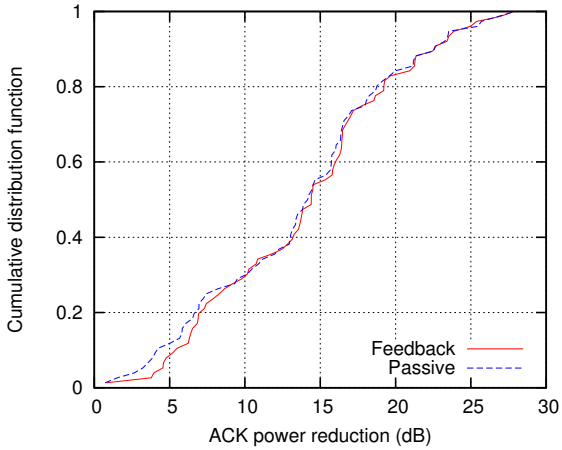


Figure 5.11: Results of power reduction of ACK frames.

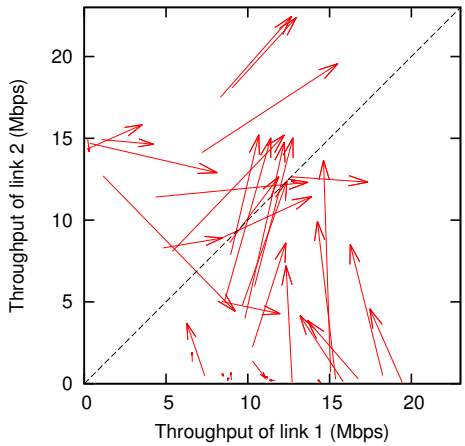
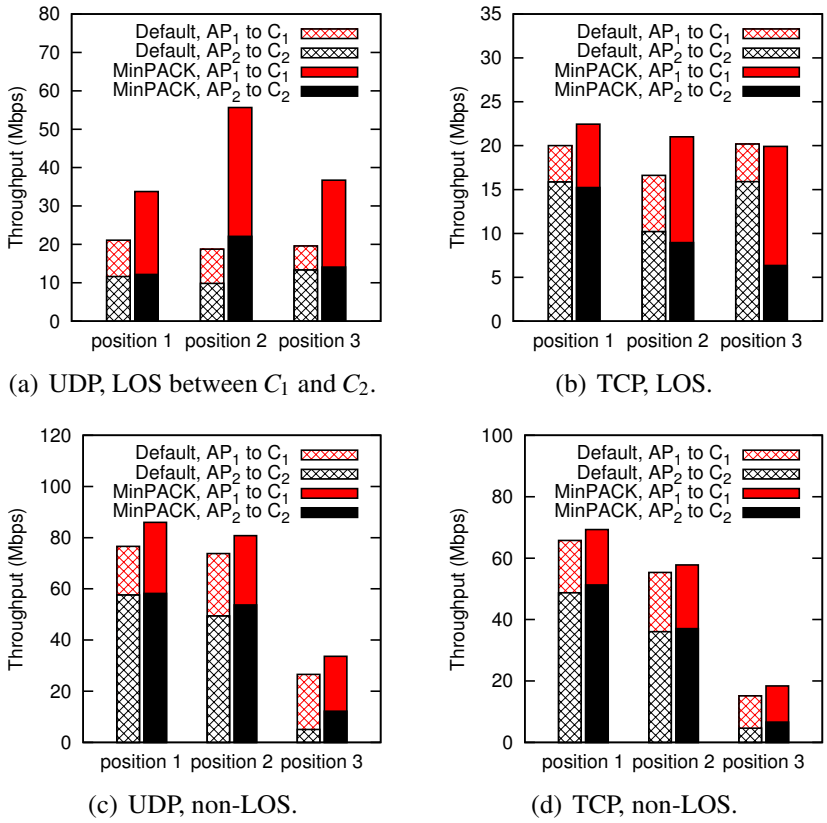


Figure 5.12: Improvement in fairness due to MinPACK for topologies in Figure 5.1.

flows. Some arrows also point towards the top right corner. This means that MinPACK is able to achieve improvements in both fairness and efficiency for these topologies.

To evaluate the performance of MinPACK with 802.11n adapters for the scenario of Figure 5.1, we used two Advantech boards as the clients and associated them with the campus WLAN APs. We first placed the clients in such a way that there was direct line-of-sight (LOS) between them. Three different scenarios were investigated. Figure 5.13(a) and Figure 5.13(b) show the throughput results of MinPACK for UDP and TCP traffic, respectively. As ACK interference is very strong due to direct LOS, MinPACK is able to significantly improve the overall UDP throughput as well as fairness (e.g., 196% throughput gain for position 2). The TCP throughput gain due to MinPACK is not as significant



(a) UDP, LOS between C_1 and C_2.

(b) TCP, LOS.

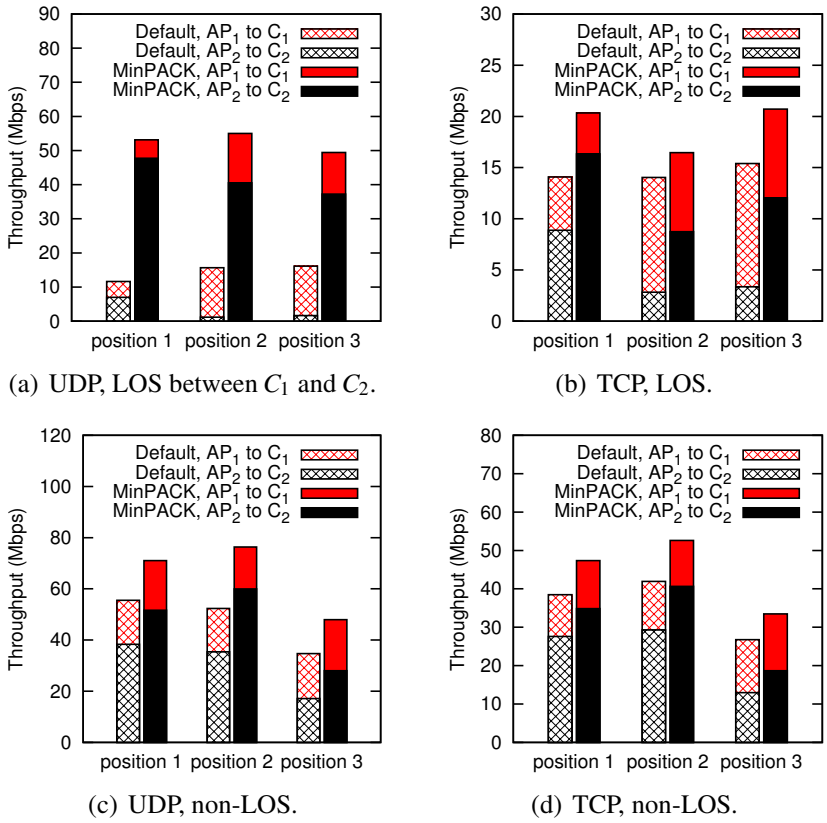
(c) UDP, non-LOS.

(d) TCP, non-LOS.

Figure 5.13: Achieved throughput for 802.11n vs. 802.11n in campus WLAN.

as that of UDP (e.g., 26% for position 2), but we still observe a great improvement of TCP fairness. We repeated the above experiment by moving the clients to the positions where there is no direct line-of-sight between the two clients. As shown in Figure 5.13(c) and Figure 5.13(d), the gain due to MinPACK is not big for non-LOS case (e.g., about 10% to 12% for UDP). That is largely because the ACK interference was already low and there is little room for improvement.

We also repeated the above experiments for the case for an 802.11a and an 802.11n client, as shown in Figure 5.14. Both clients used the same default ACK power of 20 dBm. With direct LOS, MinPACK is able to significantly improve the UDP throughput of the 802.11n client, which is less robust against ACK interference than 802.11a client (see Section 5.1.3). The total TCP throughput with MinPACK is also increased although the improvement is not large. When there is no direct LOS between the clients (Figure 5.14(c)



(a) UDP, LOS between C_1 and C_2.

(b) TCP, LOS.

(c) UDP, non-LOS.

(d) TCP, non-LOS.

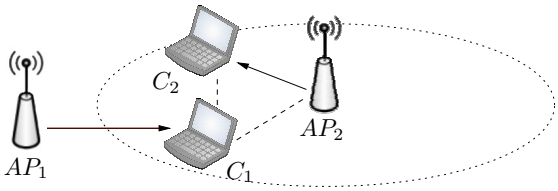
Figure 5.14: Achieved throughput for 802.11a (C_1) vs. 802.11n (C_2) in campus WLAN.

and Figure 5.14(d)), MinPACK can still improve the throughput of the 802.11n client by some extent, due to the vulnerability of 802.11n to interference.

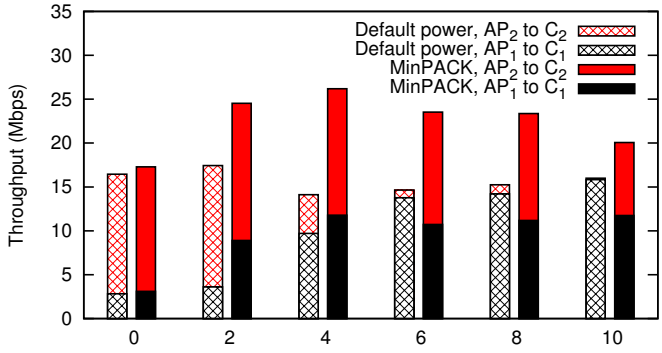
5.3.3 Power Control of Data Frames is Not Sufficient

It is not uncommon for a receiver to experience both ACK interference and data interference at the same time, as illustrated in the scenario in Figure 5.15(a). When AP_1 and AP_2 are transmitting at the same time, client C_1 is subject to the interference from both AP_2's data frames and client C_2's ACK frames. Conventional power control solutions [11, 58, 64] were designed to mitigate data frame interference by reducing the transmission power, but they did not consider the impact of ACK interference.

To investigate the coupling between MinPACK and the power control of data frames, we selected a sample topology of Figure 5.15(a) in our testbed, and run concurrent UDP



(a) Scenario of both ACK and data interference.



(b) Power reduction of AP_2's data frames (dBm)

Figure 5.15: Effect of power control of data frames.

traffic. During the experiments, we gradually reduced the transmission power of AP_2's data frames, at steps of 2 dBm. This is to emulate the behavior of the power control solutions of data frames. For each power level of node AP_2's data frame, we run the experiment with MinPACK and also with the default maximum power of ACK frames.

Figure 5.15(b) shows the measured throughput, and we make several observations. First, when AP_2's data frames were sent at the default maximum power, their interference on C_1 was so strong that the throughput of C_1 did not improve much even after MinPACK mitigated the ACK interference from C_2. Second, reducing the power of AP_2's data frames alone might not be sufficient to increase C_1's throughput because C_1 was still subject to the ACK interference from C_2. For example, after the power of AP_2's data frames was reduced by 2 dBm, the increase of C_1's throughput was very small when the default ACK power was used. Using MinPACK, the throughput of C_1 was greatly improved. Third, when the default ACK power was used and the power of AP_2's data frames dropped further (e.g., from 4 dBm onwards), we observed a continuous increase of C_1's throughput but a gradual reduction of C_2's throughput. The reason is likely because, as AP_2 reduced its

power, C_1 could receive more data frames from AP_1. As a result, more ACK frames were transmitted from C_1, causing more interference to C_2. By mitigating the ACK interference from C_1, MinPACK is able to prevent the throughput reduction of C_2, thereby improving both the combined throughput and fairness performance.

5.3.4 Client Mobility

We also investigated the performance of MinPACK when a client is mobile. The experiment setup is similar to the scenario of Figure 5.1. Client C_1 was initially located close to AP_1. It was then moved away from AP_1 towards C_2 at a speed of approximately 1 m/s for about 60 s. Figure 5.16 shows the change in throughput at default maximum ACK power and with MinPACK, respectively. At maximum power, client C_1 initially had a much higher throughput than C_2. However, as C_1 was moved away from AP_1 (e.g., after 40 s), the throughput of client C_2 increased and client C_1 was almost starved. The shift of the two clients' throughput is partly due to the continuous signal strength drop from AP_1 to C_1. Initially the signal received at C_1 was strong enough to sustain the interference from C_2's ACK but not later when it became weak. On the other hand, MinPACK achieved concurrent transmission in the first half of the trace. When the signal received at C_1 was weak (e.g., the second half), MinPACK was able to prevent the starvation of C_1 by mitigating the ACK interference from C_2. Note that, at the late stage of the trace for MinPACK, client C_2 incurred some degree of throughput drop. The reason is because the signal between AP_1 and C_1 became weak and there was smaller room for C_1 to reduce its ACK power.

5.4 Summary

We study the interference problem of MAC ACK frames in 802.11 networks. Existing work on interference mitigation only consider data frames, and little attention has been paid to the interference due to MAC ACK frames. To demonstrate the ACK interference

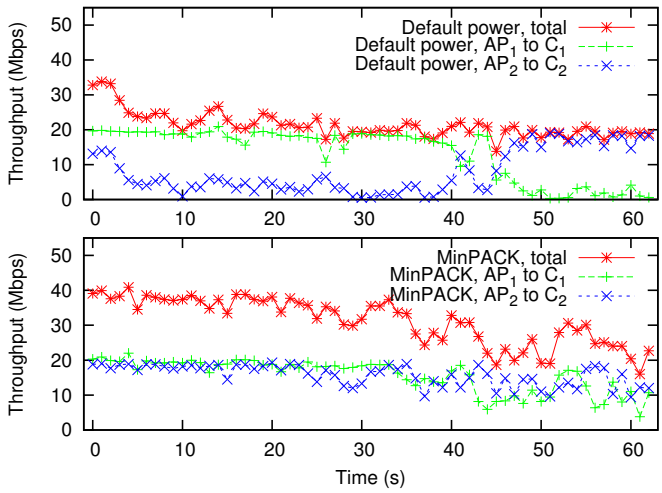


Figure 5.16: Performance with mobile client.

problem is serious, we collect extensive warwalking traces and find that the density of 802.11 networks running at the same channel is astonishingly high. We propose a simple and effective power control algorithm, called MinPACK, to mitigate the interference due to ACK frames, without affecting the original throughput. Through extensive testbed and real WLAN experiments, we show that MinPACK is able to improve both the throughput and fairness performance. Furthermore, MinPACK is complementary to available power control algorithms on data frames, and is also adaptive to changing environments.

Chapter 6

Conclusion and Future Work

In this thesis, we describe and evaluate algorithms to mitigate the adverse effects of physical layer capture as well as the interference due to MAC ACK frames. In particular, we propose FairMesh to mitigate MAC unfairness arising from capture effect in mesh networks, and MinPACK to mitigate MAC ACK interference and improve throughput in WLANs. We also investigate the impact of MIM mechanism on the reception of aggregate frames in 802.11n networks.

6.1 Impact of Physical Layer Capture Effect

FairMesh is a practical solution to compreshensively mitigate the MAC unfairness caused by physical layer capture in 802.11 mesh networks. Unlike existing works on MAC unfairness, FairMesh decouples the action of unfairness assessment from the remedial action, and utilizes the proposed water-discharging algorithm to adjust *CWmin* to mitigate unfairness.

Our experiments on a 20-node mesh testbed and `ns-2` simulations show that FairMesh is able to achieve approximate max-min fairness, not only for the basic topologies listed in Figure 3.1 but also for arbitrary topologies with up to tens of nodes. FairMesh remains efficient under high data rate and high loss rate, and interacts well with multi-hop TCP

traffic. Furthermore, FairMesh can be easily modified to support other notion of fairness such as proportional fairness. Since the root causes of MAC unfairness in 802.11 (i.e., capture effect and asymmetric topology) are also present in many other wireless technologies, we believe that the basic design of FairMesh would be broadly applicable to other networks beyond 802.11 mesh networks.

In addition to causing MAC unfairness, we also found that the capture effect could be detrimental to the reception of aggregate frames (i.e., A-MPDU) because of the MIM mechanism. When the interfering frame is stronger, the MIM mechanism could cause a throughput reduction of more than 30%. With a comprehensive set of experiments, we characterized the impact of enabling/disabling the MIM mechanism and found that: i) the air time of the interfering frames directly determines the frame delivery ratio when MIM is disabled; ii) the MIM mechanism will kick in when the interfering frames are 10 dB stronger; and iii) the MIM mechanism can still take effect when the interfering frames are at an adjacent channel but it requires a larger difference of signal strength.

As the latest 802.11ac standard adopts more aggressive frame aggregation (up to 1 MB per A-MPDU vs. 64 KB for 802.11n), we believe that our work on the MIM mechanism would have broad implications on next generation 802.11 networks in the near future. In addition, our detailed measurement study would likely be helpful to researchers who are seeking to improve the existing algorithms for rate adaptation and/or A-MPDU size selection in the presence of strong interference. Furthermore, the results of our measurement studies could also be used to enhance the 802.11n models in existing simulation tools to improve the interference models for MIM implementations.

6.2 Interference due to MAC ACK Frames

Our measurement study and analysis showed that the interference due to MAC ACK frames is common and serious in practical WLANs. In particular, the MAC ACK frames from clients could effectively extend the interference range of the AP that the clients

are associated with. Existing works on interference mitigation are focused only on Data frames but not on ACK frames.

MinPACK is a simple yet effective power control protocol to mitigate the interference due to MAC ACK frames. Based on accurate estimation of ACK success rate, MinPACK gradually reduces the power of ACK frames until the level just before ACK frames start experiencing more losses. In other words, MinPACK tries to minimize ACK interference without affecting the original throughput. We have also enhanced MinPACK to work with Block ACK in 802.11n networks.

The experiments on our 20-node testbed show that MinPACK achieves a median throughput gain of 31% over an arbitrary set of link pairs and it does no harm to the total throughput of any link pair. At the same time, MinPACK is also able to improve the fairness of most link pairs. While existing works only attempt to contain the interference due to Data frames, our experiment shows that it is insufficient to mitigate Data interference and MinPACK can complement available Data power control protocol to achieve much better throughput performance. Furthermore, MinPACK is shown to be adaptive to moderate client mobility.

Since ACK frame is an essential component to ensure the delivery of Data frame over wireless links, ACK interference might exist for other types of wireless technologies such as IEEE 802.15.4 radio. In this sense, we believe that the idea of MinPACK could also be applied to mitigate the corresponding ACK interference.

6.3 Open Issues and Future Work

There are several open issues remaining for the solutions presented in this thesis. For FairMesh, the evaluations were conducted on a stationary mesh testbed, and it would be interesting to investigate how FairMesh performs in a mobile environment. In addition, we also plan to study the impact of upper layers (e.g., routing and transport protocol) on FairMesh. For the study on the MIM mechanism, a valuable future work is to enhance the

available rate adaptation algorithms to be MIM-aware. For MinPACK, our investigation was focused only on downstream traffic where ACK frames come from clients. It remains to be explored how MinPACK performs when there is upstream traffic. In addition, we plan to integrate MinPACK with available Data power control algorithms and study their performance in practice. It is also interesting to extend the ACK interference model to more general scenarios, e.g., senders with different transmission ranges.

The work presented in this thesis also lays the foundations for two potential research directions for 802.11 networks. One research direction is to review the physical layer of 802.11 standard to allow early detection of desirable frames. In the current standard, the physical layer receives the whole 802.11 frame before passing it up to the MAC layer, which will then determine whether it is a desirable frame based on the destination MAC address in the MAC header. If the frame is an interfering frame (i.e., not desirable), the physical layer has wasted some time receiving the whole frame body. The longer the interfering frame air time is, the more waste the receiver will incur. Therefore, a better design is to have the physical layer to check the MAC destination address immediately after finishing receiving the MAC header and ignore the frame if it is not desirable. As compared with the adaptive MIM method proposed in Chapter 4, the new physical layer provides much better fine-grained control on the reception of frames in case of collision.

Another potential research direction is to develop practical aerial 802.11 networks with unmanned aerial vehicles (UAV). Wireless UAV networks have a broad range of applications, e.g., situation monitoring for firefighting and live broadcasting of sport events. Due to recent advances in UAV technologies, it becomes feasible to largely deploy wireless UAV networks in practice. We have developed a WiFi-enabled quadcopter and demonstrated that it is feasible to use the detection of mobile phone WiFi signal for search and rescue operations [75]. As 802.11 links involving UAV are highly dynamic [12], it is expected that the available 802.11 algorithms like rate adaptation may not work well for aerial 802.11 links. In addition, how the solutions proposed in this thesis will perform in aerial networks is another area that remains to be explored.

Bibliography

[1] ALIX system board (ALIX2c2) by pcengines. Website. http://www.pcengines.ch/.

[2] Cisco enterprise wireless mesh solution overview. http://tinyurl.com/olx63o4.

[3] Cisco visual networking index forecast (2012-2017). http://www.cisco.com/go/vni.

[4] Global Wi-Fi hotspot market 2013 report. http://tinyurl.com/otgs8vs.

[5] miniPCI wireless adapters (WLM54AG-23) by Compex. Website. http://www.compex.com.sg/.

[6] NUS Mesh Networks. http://mesh.ndslab.net.

[7] IEEE 802.11 standard: Wireless LAN medium access control (MAC) and physical layer (PHY) specification, June 2007.

[8] Fehmi Ben Abdesslem, Luigi Iannone, and Marcelo Dias de Amorim. On the feasibility of power control in current IEEE 802.11 devices. In *Proceedings of PerCom '06*, March 2006.

[9] Arup Acharya, Archan Misra, and Sorav Bansal. MACA-P: a MAC for concurrent transmissions in multi-hop wireless networks. In *Proc. of PerCom '03*, Mar. 2003.

[10] Daniel Aguayo, John Bicket, Sanjit Biswas, Glenn Judd, and Robert Morris. Link-level measurements from an 802.11b mesh network. In *Proceedings of ACM SIGCOMM Conference 2004*, Aug. 2004.

[11] Aditya Akella, Glenn Judd, Srinivasan Seshan, and Peter Steenkiste. Self-management in chaotic wireless deployments. In *Proceedings of MobiCom '05*, MobiCom '05, August 2005.

[12] Mahdi Asadpour, Domenico Giustiniano, and Karin Anna Hummel. From ground to aerial communication: Dissecting wlan 802.11n for the drones. In *Proc. of WiNTECH '13*, Sep. 2013.

[13] Amotz Bar-Noy, Alain Mayer, Baruch Schieber, and Madhu Sudan. Guaranteeing fair service to persistent dependent tasks. In *Proceedings of SODA '95*, Jan. 1995.

[14] Brahim Bensaou, Yu Wang, and Chi Chung Ko. Fair medium access in 802.11 based wireless ad-hoc networks. In *Proceedings of MobiHoc '00*, Aug. 2000.

[15] Dimitri Bertsekas and Robert Gallager. *Data Networks*. Prentice Hall, 1992.

[16] Vaduvur Bharghavan, Alan Demers, Scott Shenker, and Lixia Zhang. MACAW: a media access protocol for wireless LAN's. In *Proceedings of SIGCOMM '94*, Aug. 1994.

[17] Giuseppe Bianchi, Ilenia Tinnirello, and Luca Scalia. Understanding 802.11e contention-based prioritization mechanisms and their coexistence with legacy 802.11 stations. *IEEE Network*, 19(4):28–34, Jul.-Aug. 2005.

[18] John Bicket, Daniel Aguayo, Sanjit Biswas, and Robert Morris. Architecture and evaluation of an unplanned 802.11b mesh network. In *Proceedings of ACM MobiCom'05*, Cologne, Germany, Aug. 2005.

[19] Ioannis Broustis, Jakob Eriksson, Srikanth V. Krishnamurthy, and Michalis Faloutsos. Implications of power control in wireless networks: a quantitative study. In *Proceedings of PAM '07*, April 2007.

[20] Joseph Camp. *Experimental and Analytical Evaluation of Embedded Link Performance with Small-Scale Channel Fluctuations*. PhD thesis, Rice University, 2009.

[21] Joseph Camp, Vincenzo Mancuso, Omer Gurewitz, and Edward W. Knightly. A measurement study of multiplicative overhead effects in wireless networks. In *Proceedings of INFOCOM '08*, Apr. 2008.

[22] Daniel Camps-Mur, Manil Dev Gomony, Xavier Pérez-Costa, and Sebastià Sallent-Ribes. Leveraging 802.11n frame aggregation to enhance QoS and power consumption in Wi-Fi networks. *Computer Networks*, 56(12):2896–2911, May 2012.

[23] Qi Chen, Felix Schmidt-Eisenlohr, Daniel Jiang, Marc Torrent-Moreno, Luca Delgrossi, and Hannes Hartenstein. Overhaul of IEEE 802.11 modeling and simulation in ns-2. In *Proceedings of MSWiM '07*, Oct. 2007.

[24] Click. The click modular router. `http://read.cs.ucla.edu/click/`.

[25] Saumitra M. Das, Himabindu Pucha, Konstantina Papagiannaki, and Y. Charlie Hu. Studying wireless routing link metric dynamics. In *Proceedings of ACM IMC'07*, pages 327–332, New York, NY, USA, Oct. 2007. ACM.

[26] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of the 9th ACM International Conference on Mobile Computing and Networking (MobiCom '03)*, San Diego, California, Sep. 2003.

[27] Lara Deek, Eduard Garcia-Villegas, Elizabeth Belding, Sung-Ju Lee, and Kevin Almeroth. The impact of channel bonding on 802.11n network management. In *Proc. of CoNEXT '11*, Dec. 2011.

[28] Richard Draves, Jitendra Padhye, and Brian Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Proceedings of ACM MobiCom '04*, pages 114–128. ACM Press, Sep. 2004.

[29] Hossein Falaki, Dimitrios Lymberopoulos, Ratul Mahajan, Srikanth Kandula, and Deborah Estrin. A first look at traffic on smartphones. In *Proc. of IMC '10*, Nov. 2010.

[30] Zuyuan Fang, Brahim Bensaou, and Yu Wang. Performance evaluation of a fair backoff algorithm for IEEE 802.11 DFWMAC. In *Proceedings of MobiHoc '02*, Jun. 2002.

[31] Bernhard Firner, Chenren Xu, Richard Howard, and Yanyong Zhang. Multiple receiver strategies for minimizing packet loss in dense sensor networks. In *Proc. of MobiHoc '10*, Sep. 2010.

[32] Sachin Ganu, Kishore Ramachandran, Marco Gruteser, Ivan Seskar, and Jing Deng. Methods for restoring MAC layer fairness in IEEE 802.11 networks with physical layer capture. In *Proceedings of REALMAN '06*, May 2006.

[33] Michele Garetto, Jingpu Shi, and Edward W. Knightly. Modeling media access in embedded two-flow topologies of multi-hop wireless networks. In *Proceedings of MobiCom '05*, Aug. 2005.

[34] Boris Ginzburg and Alex Kesselman. Performance analysis of A-MPDU and A-MSDU aggregation in IEEE 802.11n. In *Proc. of Sarnoff '07*, Apr. 2007.

[35] Sascha Gubner and Christoph Lindemann. Evaluating the impact of frame aggregation on video-streaming over IEEE 802.11n multihop networks. In *Proc. of WoWMoM '12*, Jun. 2012.

[36] Daniel Halperin, Thomas Anderson, and David Wetherall. Taking the sting out of carrier sense: interference cancellation for wireless LANs. In *Proc. of MobiCom '08*, Sep. 2008.

[37] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. Predictable 802.11 packet delivery from wireless channel measurements. In *Proceedings of SIGCOMM'10*, Aug. 2010.

[38] Bo Han, Lusheng Ji, Seungjoon Lee, Robert R Miller, and Bobby Bhattacharjee. Channel access throttling for improving WLAN QoS. In *Proceedings of SECON '09*, Jun. 2009.

[39] Bo Han, Aaron Schulman, Francesco Gringoli, Neil Spring, Bobby Bhattacharjee, Lorenzo Nava, Lusheng Ji, Seungjoon Lee, and Robert Miller. Maranello: practical partial packet recovery for 802.11. In *Proceedings of NSDI '10*, Apr. 2010.

[40] K. D. Huang, David Malone, and Ken R Duffy. The 802.11 g 11 Mb/s rate is more robust than 6 Mb/s. *Wireless Communications, IEEE Transactions on*, 10(4):1015–1020, 2011.

[41] Xiao Long Huang and Brahim Bensaou. On Max-Min fairness and scheduling in wireless ad-hoc networks: analytical framework and implementation. In *Proceedings of MobiHoc '01*, Oct. 2001.

[42] Ying Jian and Shigang Chen. Can CSMA/CA networks be made fair? In *Proceedings of MobiCom'08*, Sep. 2008.

[43] Glenn Judd and Peter Steenkiste. Characterizing 802.11 wireless link behavior. *Wireless Networks*, 16(1):167–182, Jan. 2010.

[44] Phil Karn. MACA: a new channel access method for packet radio. In *Proceedings of the 9th ARRL Computer Networking Conference*, september 1990.

[45] Frank Kelly, Aman Maulloo, and David Tan. Rate control in communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49(3):237–252, Mar. 1998.

[46] Minkyong Kim, Jeffrey J Fielding, and David Kotz. Risks of using AP locations discovered through war driving. In *Pervasive Computing*, pages 67–82. Springer, 2006.

[47] Tae-Suk Kim, Hyuk Lim, and Jennifer C. Hou. Improving spatial reuse through tuning transmit power, carrier sense threshold, and data rate in multihop wireless networks. In *Proc. of MobiCom '06*, Sep. 2006.

[48] Leonard Kleinrock and Fouad Tobagi. Packet switching in radio channels: Part i–carrier sense multiple-access modes and their throughput-delay characteristics. *IEEE Transactions on Communications*, 23(12):1400–1416, December 1975.

[49] Andrzej Kochut, Arunchandar Vasan, A. Udaya Shankar, and Ashok Agrawala. Sniffing out the correct physical layer capture model in 802.11b. In *Proceedings of ICNP '04*, Oct. 2004.

[50] Karol Kowalik, Marek Bykowski, Brian Keegan, and Mark Davis. Practical issues of power control in IEEE 802.11 wireless devices. In *Proceedings of ICT '08*, June 2008.

[51] Lito Kriara, Mahesh K Marina, and Arsham Farshad. Characterization of 802.11n wireless LAN performance via testbed measurements and statistical analysis. In *Proc. of SECON '13*, Jun. 2013.

[52] Jeongkeun Lee, Wonho Kim, Sung-Ju Lee, Daehyung Jo, Jiho Ryu, Taekyoung Kwon, and Yanghee Choi. An experimental study on the capture effect in 802.11a networks. In *Proceedings of WiNTECH '07*, Sep. 2007.

[53] Krijn Leentvaar and Jan Flint. The capture effect in FM receivers. *IEEE Transactions on Communications*, 24(5):531–539, May 1976.

[54] Xi Liu, Anmol Sheth, Michael Kaminsky, Konstantina Papagiannaki, Srinivasan Seshan, and Peter Steenkiste. Pushing the envelope of indoor wireless spatial reuse using directional access points and clients. In *Proc. of MobiCom '10*, Sep. 2010.

[55] Jiakang Lu and Kamin Whitehouse. Flash flooding: Exploiting the capture effect for rapid flooding in wireless sensor networks. In *Proceedings of IEEE INFOCOM'09*, Apr. 2009.

[56] Gregor Maier, Anja Feldmann, Vern Paxson, and Mark Allman. On dominant characteristics of residential broadband internet traffic. In *Proc. of IMC '09*, Nov. 2009.

[57] Justin Manweiler, Naveen Santhapuri, Souvik Sen, Romit Roy Choudhury, Srihari Nelakuditi, and Kamesh Munagala. Order matters: transmission reordering in wireless networks. In *Proceedings of MobiCom '09*, Sep. 2009.

[58] Vivek P. Mhatre, Konstantina Papagiannaki, and Francois Baccelli. Interference mitigation through power control in high density 802.11 WLANs. In *Proc. of INFOCOM '07*, May 2007.

[59] Bratislav Milic and Miroslaw Malek. NPART - node placement algorithm for realistic topologies in wireless multihop network simulation. In *Proceedings of Simutools '09*, Mar. 2009.

[60] Amit Mondal and Aleksandar Kuzmanovic. Removing exponential backoff from TCP. *SIGCOMM Computer Communication Review*, 38(5):17–28, 2008.

[61] Thyagarajan Nandagopal, Tae-Eun Kim, Xia Gao, and Vaduvur Bharghavan. Achieving MAC layer fairness in wireless packet networks. In *Proceedings of MobiCom '00*, Aug. 2000.

[62] Eng Hwee Ong, Jarkko Kneckt, Olli Alanen, Zheng Chang, Toni Huovinen, and T Nihtila. IEEE 802.11ac: Enhancements for very high throughput WLANs. In *Proc. of PIMRC '11*, Sep. 2011.

[63] Utpal Paul, Riccardo Crepaldi, Jeongkeun Lee, Sung-Ju Lee, and Raul Etkin. Characterizing WiFi link performance in open outdoor networks. In *Proc. of SECON '11*, Jun. 2011.

[64] Kishore Ramachandran, Ravi Kokku, Honghai Zhang, and Marco Gruteser. Symphony: synchronous two-phase rate and power control in 802.11 WLANs. In *Proceedings of MobiSys '08*, June 2008.

[65] Bhaskaran Raman, Kameswari Chebrolu, Dattatraya Gokhale, and Sayandeep Sen. On the feasibility of the link abstraction in wireless mesh networks. *IEEE/ACM Transactions on Networking.*, 17(2):528–541, Apr. 2009.

[66] Shravan Rayanchu, Arunesh Mishra, Dheeraj Agrawal, Sharad Saha, and Suman Banerjee. Diagnosing wireless packet losses in 802.11: Separating collision from weak signal. In *Proceedings of INFOCOM'08*, Apr. 2008.

[67] Roofnet. The mit roofnet. Website, 2009. `http://pdos.csail.mit.edu/roofnet/`.

[68] Naveen Santhapuri, Romit Roy Choudhury, Justin Manweiler, Srihari Nelakuduti, Souvik Sen, and Kamesh Munagala. Message in message MIM: A case for reordering transmissions in wireless networks. In *Proceedings of HotNets'08*, Oct. 2008.

[69] Jingpu Shi, Omer Gurewitz, Vincenzo Mancuso, Joseph Camp, and Edward W. Knightly. Measurement and modeling of the origins of starvation in congestion controlled mesh networks. In *Proceedings of INFOCOM '08*, Apr. 2008.

[70] Vivek Shrivastava, Dheeraj Agrawal, Arunesh Mishra, Suman Banerjee, and Tamer Nadeem. Understanding the limitations of transmit power control for indoor wlans. In *Proceedings of IMC '07*, October 2007.

[71] Vivek Shrivastava, Nabeel Ahmed, Shravan Rayanchu, Suman Banerjee, Srinivasan Keshav, Konstantina Papagiannaki, and Arunesh Mishra. CENTAUR: realizing the full potential of centralized WLANs through a hybrid data path. In *Proc. of MobiCom '09*, Sep. 2009.

[72] Dionysios Skordoulis, Qiang Ni, Hsiao-Hwa Chen, Adrian P Stephens, Changwen Liu, and Abbas Jamalipour. IEEE 802.11n MAC frame aggregation mechanisms for next-generation high-throughput WLANs. *Wireless Communications, IEEE*, 15(1):40–47, 2008.

[73] Kun Tan, He Liu, Ji Fang, Wei Wang, Jiansong Zhang, Mi Chen, and Geoffrey M Voelker. SAM: enabling practical spatial multiple access in wireless LAN. In *Proc. of MobiCom '09*, Sep. 2009.

[74] Ilenia Tinnirello, Domenico Giustiniano, Luca Scalia, and Giuseppe Bianchi. On the side-effects of proprietary solutions for fading and interference mitigation in IEEE 802.11b/g outdoor links. *Computer Networks*, 53(2):141–152, Oct. 2009.

[75] Wei Wang, Raj Joshi, Aditya Kulkarni, Wai Kay Leong, and Ben Leong. Feasibility study of mobile phone WiFi detection in aerial search and rescue operations. In *Proc. of APSys '13*, Jul. 2013.

[76] Wei Wang, Ben Leong, and Wei Tsang Ooi. Mitigating unfairness due to physical layer capture in practical 802.11 mesh networks. *IEEE Transactions on Mobile Computing*, To appear.

[77] Wei Wang, Wai Kay Leong, and Ben Leong. Potential pitfalls of the message in message mechanism in modern 802.11 networks. In *Proc. of WiNTECH '14*, Sep. 2014.

[78] Wei Wang, Qiang Wang, Wai Kay Leong, Ben Leong, and Yi Li. Uncovering a hidden wireless menace: Interference from 802.11x MAC acknowledgment frames. In *Proc. of SECON '14*, Jun. 2014.

[79] Christopher Ware, John Judge, Joe Chicharo, and Eryk Dutkiewicz. Unfairness and capture behaviour in 802.11 adhoc networks. In *Proceedings of ICC '00*, Jun. 2000.

[80] Kamin. Whitehouse, Alec. Woo, Fred. Jiang, Joseph. Polastre, and David Culler. Exploiting the capture effect for collision detection and recovery. In *Proceedings of EmNets'05*, May 2005.

[81] Starsky H. Y. Wong, Hao Yang, Songwu Lu, and Vaduvur Bharghavan. Robust rate adaptation for 802.11 wireless networks. In *Proceedings of MobiCom '06*, Sep. 2006.

[82] Guoqing Yu, Wei Wang, James Yong, Ben Leong, and Wei Tsang Ooi. Adaptive antenna adjustment for 3D urban wireless mesh networks. In *Proc. of SECON '13*, Jun. 2013.

[83] Anfu Zhou, Min Liu, Zhongcheng Li, and Eryk Dutkiewicz. Modeling and optimization of medium access in CSMA wireless networks with topology asymmetry. *IEEE Transactions on Mobile Computing*, Aug. 2011.

[84] Xiaoyi Zhu, Angela Doufexi, and Taskin Kocak. Throughput and coverage performance for IEEE 802.11ad millimeter-wave WPANs. In *Proc. of VTC Spring '11*, May 2011.

[85] Michele Zorzi and Flaminio Borgonovo. Performance of capture-division packet access with slow shadowing and power control. *IEEE Transactions on Vehicular Technology*, 46(3):687–696, Aug. 1997.