Interference Mitigation and Detection in WiFi Networks Under Congestion

by

Kan Cai

B. Eng., Northeastern University, 1998

M. Sc., The University of British Columbia, 2004

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

(Computer Science)

The University of British Columbia
(Vancouver)

June 2015

Abstract

In IEEE 802.11, nodes regulate access to the airspace they share in a decentralized fashion using CSMA/CA. The goal of this approach is to share the common airspace fairly and efficiently without requiring centralized channel administration or direct coordination among peer nodes. However, it is well known that strong interference, as consequence of this de-centralized coordination scheme, can lead to extremely unfair network bandwidth allocation between competing devices.

Interference detection and mitigation has posed great challenges. The cause of interference is complicated, involving many networking factors such as topology and traffic, and the interference relationship changes all the time. This thesis addresses these challenges by proposing a throttling based interference mitigation system (Shaper) and an online passive interference detection system (VOID). The main contribution of this thesis is to point out the correlated relationship between interference and congestion.

First, this thesis provides a more thorough analysis on the impact of node topology, traffic type and signal strength on wireless performance. We came up with 9 UDP models and 10 TCP models just for two competing flow scenarios. The outcome of wireless interference can get harder to predict, however, as we introduce more factors into the interference model such as more competing nodes, sending rate, signal propagation model, etc.

On the other hand, this thesis identifies the immediate cause to the unfair bandwidth distribution under interference: 802.11 network congestion. We observed and explained that all competing devices are able to perform well regardless of

topology or traffic class, as long as there is sufficiently more bandwidth than the aggregate throughput demands. Therefore, we propose to trade the aggregate throughput to mitigate the impact of interference and prove its effectiveness through simulation and emulation.

Finally, the key to successful addressing the interference is an accurate and fast interference detection mechanism and this thesis proposes such a system called VOID. It deploys the correlation between congestion and interference to infer the interference relationship from the ip-layer throughput variations. It is fast, accurate and more importantly, very easy to deploy in existing WiFi networks.

Preface

I conducted my research work presented in this thesis in collaboration with my Ph.D. supervisor Michael J. Feeley. The thesis content is mostly based on the following four published papers, in which I am the primary contributor on both the design and evaluation:

- Kan Cai and Michael J. Feeley and Brendan Cully and Sharath J. George. Understanding Performance for Two 802.11 Competing Flows. In Proceedings of the 4th IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS). Pages 1-11. 2007.

- Kan Cai and Michael Blackstock and Reza Lotun and Michael J. Feeley and Charles Krasic and Junfang Wang. Wireless unfairness: alleviate MAC congestion first! in Proceedings of the 2nd ACM Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WINTECH). Pages 43-50. 2007.

- Kan Cai and Junfang Wang and Reza Lotun and Michael J. Feeley and Michael Blackstock and Charles Krasic. A wired router can eliminate 802.11 unfairness, but it's hard. In Proceedings of the 9th Workshop on Mobile Computing Systems and Applications (HotMobile). Pages 49-54. 2008.

- Kan Cai and Michael Blackstock and Michael J. Feeley and Charles Krasic. Non-intrusive, dynamic interference detection for 802.11 networks. In Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement (IMC). Pages 377-283. 2009.

Much of the original text of the above publications has transitioned into this thesis, and therefore some of the sentences and paragraphs were co-authored by the collaborators listed above. Chapter 4 also presents our recent evaluation results on interference detection in live networks, which has not yet been published.

Table of Contents

List of Tables

List of Figures

Glossary

AIMD Additive Increase Multiplicative Decrease

VOID Wireless Online Interference Detector

IEEE Institute of Electrical and Electronics Engineers

CSMA/CA Carrier Sense Multiple Access with Collision Avoidance

AP Access Point

MIMO Multiple Input and Multiple Output

MLR Multiple Linear Regression

NAV Network Allocation Vector

WLAN Wireless Local Area Networking

TCP Transmission Control Protocol

IP Internet Protocol

UDP User Datagram Protocol

TR Transmission Range

SR Sensing Range

OR Out of Range

DIFS	Distributed Inter-Frame Space
EIFS	Extended Inter-Frame Space
PLCP	Physical Layer Convergence Protocol
MAC	Media Access Control
ACK	Acknowledge
OFDM	Orthogonal Frequency Division Multiplexing
FIFO	First In First Out
SFQ	Stochastic Fairness Queueing
RED	Random Early Detection
HTB	Hierarchical Token Bucket
RTT	Round Trip Time
SNMP	Simple Network Management Protocol
CC	Correlation Coefficient
RF	Radio Frequency
LSC	Life Sciences Center
WCS	Wireless Control System
FPR	False Positive Ratio
FNR	False Negative Ratio
VTBR	Valid Time Bucket Ratio
CRC	Cyclic Redundancy Check

CTS Clear-To-Send

PISD Proportional Increase Synchronized multiplicative Decrease

IAC Interference Alignment and Cancellation

MUBF Multi-user Beamforming

SINR Signal to Interference and Noise Ratio

Acknowledgments

First of all, I would like to say a big thank you to my supervisor Mike Feeley. He is not only a brilliant researcher, but also a great friend. He always motivates me and challenges my ideas with high standards, while at the same time provides me great freedom to explore the research area that intrigues me most. Without his continuous encouragement and patience, I would not be able to finish this thesis 5 years after starting my job at Google.

I also would like to thank my supervisor committee members: Norm Hutchinson, Alan Wagner and Charles Krasic. They are always there offering their ideas and advice at different stages of my work. I have never hesitated to bounce my idea of them during my research. It is my privilege to work with such great researchers.

I'd like to thank the UBC NSS folks for their support and feedback. In particular, I would like to thank Mike Blackstock being my research buddy and personal friend. I appreciate his new research perspectives, and his willingness to listen and discuss any crazy ideas of mine.

Also, big thanks to my parents, Ping Wu and Minli Cai, to my sister, Lei Wu, for their belief and faith in me even when myself is skeptical.

Finally, and mostly importantly, I would like to thank my wife, Junfang Wang, for her unconditional love, endless encouragement and endurance, and for completing me with a family. I love you!

I would like to dedicate this thesis to my son, Raymond Cai, who is just one year old now yet has already opened a new world to me. I look forward to many

years ahead together with you.

Chapter 1

Introduction

1.1 Issues: Wireless Interference and Unfairness

Wireless communication has seen rapid advances with millions of IEEE 802.11 devices widely deployed [18]. The reason for this popularity is mainly three fold. First, the frequency band WiFi devices operate at is unregulated and thus the production and deployment cost is lower than the other wireless technologies. Second, 802.11 conveniently allows users to be mobile while being connected to the Internet anywhere at anytime. Furthermore, 802.11 mesh networks can extend the Internet coverage in places where wired connections are expensive to provide or hard to set up. Finally, it provides satisfactory bandwidth to meet the throughput demands of most applications. As 802.11 popularity grows, the challenge will be to continue to deliver this second benefit as device density and throughput demand increases.

Unfortunately, the performance of 802.11 is unpredictable. In a dynamic network where users come and go and signal interference and packet collisions between competing devices persist, the amount of bandwidth a device actually receives depends on the constantly changing environmental factors such as node topology and channel conditions. In networks where hidden terminals and exposed terminals exist, some devices can be starved by the others for long periods

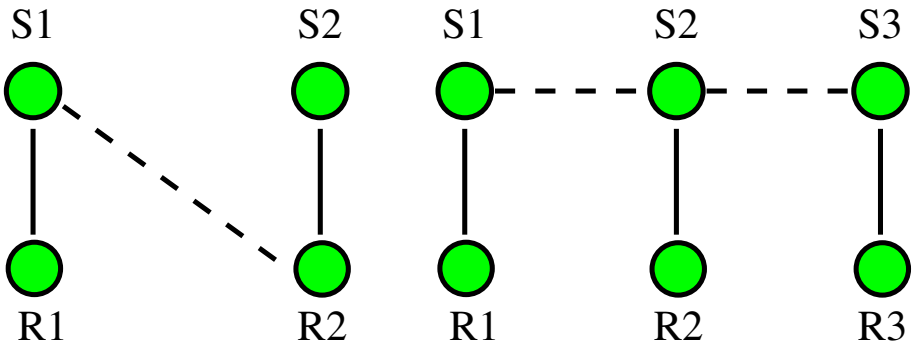
of time [30, 31, 40, 50]. As mobile computing becomes increasingly important in our daily lives, it is more critical to provide a reasonable degree of fairness for bandwidth distribution between competing 802.11 devices.

The wireless interference problem and the unfairness problem are fundamental to 802.11's decentralized, signal-based airspace arbitration mechanism using a CSMA/CA and random back-off protocol. Nodes with packets to send engage in an uncoordinated competition for channel access by delaying transmission until senders see clear air and by backing-off and re-transmitting when collisions occur. This approach works well with only one Access Point (AP) in the network — the AP acts as the only arbitrator for packet scheduling and can ensure fair networking access time for all its associated devices.

However, the 802.11 protocol is not designed to support multiple autonomous systems that operate together: in a network where multiple APS co-exist and have incomplete and inconsistent channel conditions, the signal sensing alone can not ensure a fair competition for airspace [40]. We illustrate this problem in Figure 1.1 with the two infamous node topologies: the hidden-terminal and exposed-terminal topologies.

Both of these problematic scenarios can result in severe unfair bandwidth allocation [30, 31]. In the hidden-terminal scenario, signals sent by one sender can be corrupted by another, but not vice versa. We can see from Figure 1.1a that the senders of the two flows, S1 and S2, cannot sense each other but S1's signal is strong enough to corrupt S2's signal at its receiver R2. Therefore, when S1 and S2 transmit at the same time, S2's transmission will fail while S1's will succeed.

In the exposed-terminal scenario, a sender is exposed to many other senders. Therefore, even if it does not experience any packet loss itself, it will lose its fair share because it is forced to relinquish the airspace more often than others. We can see from Figure 1.1b that the three senders do not corrupt each other's signals. However, S2 is positioned between S1 and S3 and can sense both of their signals, and thus it is not able to transmit when either S1 or S3 is transmitting. When flows 1 and 3 transmit at full speed, the sender S2 will hardly detect any idle airspace



(a) Hidden Terminals **(b)** Exposed Terminals

Figure 1.1: Two Problematic Topologies

for its own transmissions.

In addition to node topology, various environmental factors, sending rate and signal strength, for example, can all affect the outcome of bandwidth allocation for wireless devices. But as long as the aggregate throughput demand does not exceed the 802.11 network capacity, the wireless unfairness problem will not occur. The reason is that the winning devices only use a fraction of the available bandwidth and the remaining bandwidth is still sufficient to meet the disadvantaged devices' needs. The consequence of wireless devices competing in unsaturated WiFi networks is an increase in latency for the losing devices. Even if packets are corrupted, the TCP and 802.11 retransmission mechanisms are able to recover the dropped packets when the airspace is idle. However, when the winning devices become more active, airspace becomes congested and retransmissions are no longer effective at salvaging packages. The disadvantaged devices can be completely deprived of any bandwidth by the nearby competing devices for a significantly long time.

Unfortunately, 802.11 network congestion is already quite common in wireless networks. Henderson et. al. [55] have shown that wireless users use bandwidth-greedy applications just as they would on wired connections. As the capacity of the network backbone that connects an AP to the Internet increases to Gigabit

ethernet, congestion is even more likely to occur on the last-hop wireless links. Collected traces have shown that, even in well planned wireless networks such as hotels [83] and university buildings [41], wireless users often suffer from performance degradation caused by packet collisions. To be presented in Chapter 4, our interference analysis in UBC shows that interference happens often in live enterprise wireless networks, and lasts 10 minutes on average when it occurs. As the likelihood of 802.11 network congestion continues to grow, the wireless congestion and unfairness problems will get worse.

Many approaches have been proposed to mitigate interference in 802.11 networks. For example, the RTS/CTS packet exchange of 802.11 was designed to resolve the hidden terminal problem by sharing the transmission schedule among potential interfering neighbors. However, it is not used in practice as it requires to transmit two additional control packets without any payload, which can lead to bandwidth inefficiency in the absence of hidden terminals. Furthermore, RTS/CTS packets cannot be used to address the exposed terminal problem. Therefore, the interference mitigation problem is mostly addressed by systems running on top of 802.11, which must identify and change the interference relationship in the network.

Two most common solutions for example are power adaptation and channel hopping. Both alleviate interference by changing the underlying node topology to break the interference relationship between the dominating devices and the victim ones. In this thesis we present another approach in which aggregate throughput is constrained to eliminate congestion. As we have discussed above, as long as 802.11 network congestion does not occur, the impact of signal interference can be remedied by TCP and 802.11 packet salvage schemes. In the next section, we will discuss the challenges these proposals have to address to remain effective in practice.

1.2 Challenges

To effectively mitigate interference in live 802.11 networks, a system needs to address the following three challenges. First, it is hard to identify the cause of the interference from a large number of possible candidates: problematic topology, signal strength fluctuation, traffic burstiness, even non-802.11 devices such as microwaves, cell phones, etc. Many of the proposed solutions, MIMO in 802.11n [12], channel hopping [20, 33, 53], power adaptation [20, 33], for example, can help in certain scenarios, but have their limitations in environments where node density and mobility is high.

Second, one prerequisite of any interference mitigation systems is to correctly identify the underlying interference relationship among wireless devices. Only if the set of interfering devices are detected can these systems select the right targets for channel hopping, channel switching or traffic throttling. However, prior work in pair-wise interference measurements have shown that it takes many hours to derive an interference map even in a static environment consisting of only tens of wireless devices [74]. On the other hand, the interference relationship changes frequently in live networks. The wireless network is inherently dynamic due to node mobility and traffic variation, and it is difficult for an interference mitigation system to adapt to this changing environment.

Finally, to be easily deployed in existing 802.11 networks, these interference mitigation systems need to be effective with little cooperation from the wireless end-user devices. These end-user devices are running different firmware and drivers, and they are usually out of the control of the networking administrators. Therefore, the effectiveness of an interference mitigation system will be limited in reality if it requires firmware updates or protocol redesign to all wireless clients.

1.3 Motivations

This thesis is motivated by our earlier discussion on the correlation between interference and congestion in Section 1.1, and aims to come up with an effective

and practical interference mitigation system by addressing the above three challenges in live networks. We know that whether a wireless device can achieve its fair share of the network bandwidth depends not only on environmental factors such as node topology, channel conditions, traffic class, etc., but also on the level of network congestion. The impact of interference on throughput and fairness is limited as long as congestion does not occur — the package salvage schemes of the Transmission Control Protocol (TCP) and 802.11 layers can efficiently reschedule packets for the disadvantaged devices when airspace is idle. Therefore, if we can prevent congestion from happening in 802.11 networks, we could actually mitigate the impact of interference in the network.

In Chapter 3, we conducted an experiment that demonstrates the correlation between wireless unfairness and airspace congestion. We set up the hidden-terminal topology as illustrated in Figure 1.1a with two 801.11G APs and two 802.11G laptops, each laptop is associated with one AP. We streamed both long-term TCP flows to the laptops with Mxtraf (a traffic generator, as part of the QStream project [7]) as fast as possible. Both flows, however, went through a central router where we can throttle the aggregate throughput of both flows. We started with 10M bps aggregate throughput and then increase by 2M bps every minute, and track both the aggregate throughput and the bandwidth share of each flow. We observed that even though flow 2 is obviously the losing flow in this hidden-terminal topology, it can still achieve its fair bandwidth share until the aggregate throughput reaches 20M bps – both flows split the bandwidth 50-50. However, as the aggregate throughput continues to grow to 24M bps, the winning flow (i.e., flow 1) was able to finally starve the losing flow (i.e., flow 2) completely.

In brief, the key of our observation is that there exists strong correlation between 802.11 network congestion and wireless unfairness: as long as there is sufficient airspace bandwidth for the losing flows to recover from their packet losses caused by interference, the competing devices can still meet their throughput demands to achieve wireless fairness. When airspace is not congested, interference

still exists but its impact is low. Severe interference problems such as throughput starvation will occur if and only if the aggregated throughput exceeds the available bandwidth capability.

Therefore, we are motivated to leverage this correlation to trade the overall throughput for interference reduction and better fairness. Our evaluation shows that fairness can be achieved once we throttle the aggregated throughput to 15-20% below the available bandwidth capacity. The advantage of this approach is that it makes no assumptions on the underlying node topology or channel conditions, thus it can be effective in various scenarios. Moreover, traffic throttling takes place at the IP layer and requires no changes to the end-user devices or APS, making it easy to be deployed in existing 802.11 networks.

Finally, we exploit the property of interference under congestion to address the related challenge of distinguishing interference relationship. As described earlier, this information is required to deploy interference mitigation strategies, throttling in our case, in wireless networks. Fortunately, when the airspace is fully saturated, the throughput that a disadvantaged node can achieve is mostly determined by its surrounding competing devices: when its neighbors' throughput goes up, its own will go down, and vice versa. An interference analysis system can thus track throughput variations in the network and identify the set of interfering devices if it has observed enough throughput-change correlations among them. This approach again does not require any cooperation from end-user devices and uses only the IP-layer throughput information collected from central routers.

1.4 Thesis

In this thesis, we describe our interference detection and mitigation system that exploits the correlation between interference and congestion in centralized 802.11 networks. It consists of two subsystems. One is called Shaper, designed to trade throughput for fairness by throttling aggregated throughput at central routers. The other is called called VOID, which infers interference relationships from correlated throughput variations monitored at the central router.

Our system is effective because it is capable of addressing all the three challenges discussed in Section 1.2. It throttles throughput at the IP layer, preventing congestion from occurring in the first place and thus limiting the impact of interference in the network. It tracks the throughput variations at the wired network to correlate interference relationships among wireless devices. We show in Chapter 4 that this approach is not only accurate but also quick to converge, taking only seconds to minutes. Finally, our approach makes no assumptions about the underlying node topology or channel conditions and does not require cooperation from end-user devices.

In particular, this thesis makes the following contributions in each of the respective chapters:

Chapter 2: Interference Impact Analysis: Whether a wireless device can achieve its fair share of the network bandwidth depends on the node topology, channel condition, traffic class, and other environmental factors. To better understand how interference can impact the performance of wireless devices, we simulated two competing wireless flows in hundreds of scenarios by varying three environmental factors: node topology, traffic class and signal strength. These scenarios are categorized into 9 UDP models and 10 TCP models based on the differences in throughput and fairness. These results also demonstrated that the interference relationship is very complicated in 802.11 networks, where a slight change in one of the networking settings can completely change the outcome of throughput for a wireless device. Therefore, many of the previously proposed approaches designed to address interference by adjusting one network parameter: power adaptation or channel hopping, for example, cannot stay effective in continuously changing WiFi networks.

Chapter 3: Achieve Fairness by Congestion Reduction: Congestion in 802.11 networks is the immediate cause of unfair bandwidth allocation, and therefore reducing the congestion level is the key to achieving fairness in WiFi networks. Our experiments clearly demonstrate the correlation between congestion and unfairness, and in Chapter 3, we introduce a system called Shaper that en-

sures fair bandwidth distribution to competing devices by throttling their aggregate throughput at a central router that all traffic goes through. Unlike the other interference mitigation systems, Shaper is a cross-layer approach that only uses IP-layer information, and therefore it can stay effective in almost all scenarios regardless of the cause of interference. We not only prove the effectiveness of Shaper in our testbed environment, but also provide an in-depth discussion of system design, implementation, and challenges to deploy Shaper in large and central wireless networks.

Chapter 4: Passive Interference Detection Using Online Traffic: The key to make Shaper (or any interference mitigation system) effective is to detect interference when it occurs. Moreover, such an interference detection system has to be adaptive, fast, accurate, and require no modifications to the end-user devices or APs. VOID is such a system that meets all these requirements. It looks for throughput variation patterns in the wired networking traces: one's throughput goes up while the other's goes down, to correlate interference relationships using a Multiple Linear Regression (MLR) model. It only needs information readily available from the IP-layer at a central router, and therefore can be easily deployed in any centralized WiFi networks. It is fast, able to converge to an interference map within seconds, and thus can quickly adapt to environment changes. It is also accurate; our testbed and live network experiments show that it is able to accurately identify the truly interfering devices from many candidates included for analysis.

1.5 Thesis Organization

Chapter 2 provides a thorough background and analysis on how WiFi devices could interfere with each other. We vary the node topology, traffic type and link state to illustrate how they affect the bandwidth allocation between competing flows and why. Chapter 3 concentrates on the correlation between 802.11 network congestion and wireless unfairness, and demonstrates that we can trade aggregate bandwidth utilization for fairness. We also discuss the challenges to deploy such a system in existing WiFi networks. One of the biggest challenges, interference

detection, is addressed in Chapter 4. We proposed VOID to infer interference from throughput variations in the wired trace. Chapter 5 discusses the related work on interference detection and mitigation and finally Chapter 6 concludes the thesis and discusses the future research directions.

Chapter 2

Interference Impact Analysis

As this thesis intends to detect and mitigate interference in 802.11 networks, it is critical to first understand how interference can affect devices' performance in various scenarios. Most of the previous interference analysis work focuses mainly on the impact of problematic topologies exemplified by the hidden-terminal and exposed-terminal scenarios, and assumes the other environmental factors remain unchanged.

In reality, however, an interference relationship is more complicated than simple topology categorization. As we later demonstrate in this chapter, even if the topology stays unchanged, the wireless devices can perform significantly differently if the traffic type or channel condition changes. To deploy the proper interference mitigation approach for a poorly performing wireless client in one scenario, we need to understand the direct underlying environmental factor that is adversely affecting its performance: is the poor performance because of poor signal strength or high sending rate? Is the problem happening at the packet sending side or the receiving side? To help answer these questions, we provide a more detailed analysis on the impact of interference on 802.11 flows and provide concrete suggestions to improve the performance of devices in different scenarios.

In this chapter, we consider both TCP and UDP flows and a comprehensive set of node topologies with two competing flows. These two-flow scenarios are the

basic building blocks of any network topology, and previous works [50, 73] have shown that they can be used to evaluate the collision probability for a transmitter and to predict the interference effects in more complex scenarios. We vary these two-flow topologies to consider all combinations of the following four node-to-node interactions: (1) nodes unable to read or sense each other, (2) nodes able to sense each other but not able to read each other's packets and nodes able to communicate with (3) weak and with (4) strong signal. We evaluate all possible cases through simulation and show that the cases can be reduced to 9 UDP and 10 TCP models with similar efficiency/fairness characteristics. We also validate our simulation results with experiments conducted in a laboratory testbed. These more detailed models improve on previous work such as hidden-/exposed-terminal categorization and are thus better suited as a basis for adaptive techniques to improve performance in 802.11 multi-hop Wireless Local Area Networking (WLAN).

The content of this chapter is mostly based on our previous paper [35]. We start by giving an introduction in Section 2.1. Section 2.2 introduces the methodology that we use to model all different interference scenarios. In Section 2.3, we describe the 9 UDP and 10 TCP models for two 802.11G competing flows in great detail. Then, Section 2.4 provides both the simulation and testbed results to prove the accuracy of our models. Finally Section 4.8 summarizes this chapter.

2.1 Introduction

As we discussed in Chapter 1, in IEEE 802.11, nodes regulate access to the airspace they share in a decentralized fashion using Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) and random backoff protocols. Nodes with packets to send engage in an uncoordinated competition for channel access by delaying transmission until sender and receiver see clear air and by backing-off and re-transmitting when collisions occur. The goal of this approach is to share the common airspace fairly and efficiently without requiring centralized channel administration or direct coordination among peer nodes.

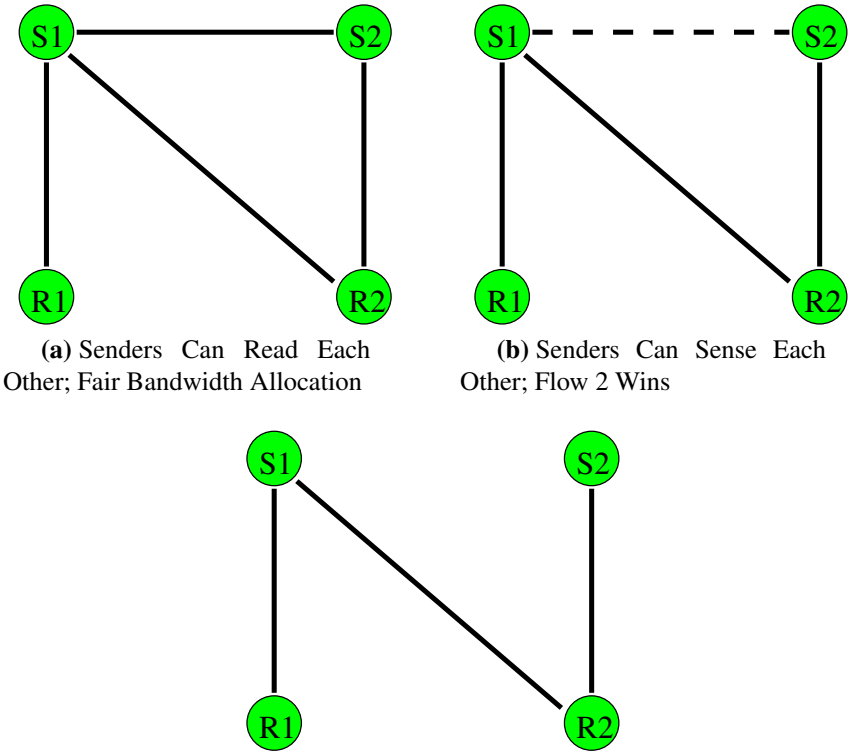
Unfortunately, in congested environments things often do not go according

to plan. It has been shown that the protocol often exhibits unpredicted performance degradation [31, 62] and unfair channel allocation [30, 31, 40] due to the node topology and other environmental factors. This performance of 802.11 is surprisingly not well understood and the complexity of the environment and the decentralized nature of the protocol make understanding elusive. Nevertheless, as 802.11 popularity grows, congestion increases and emerging applications such as media streaming place new demands on network performance predictability, there is a growing need for a deeper understanding of how 802.11 deals with congested traffic in practice [55].

The early understanding of 802.11 was that sending nodes are confronted with two types of potentially competing nodes: *hidden-* and *exposed-terminals* [31]. The issue for the protocol is that the sender decides when to send, but it is the channel conditions at the receiver that determine successful delivery. Nodes hidden from the sender can cause corruption, and nodes exposed to the sender but hidden from the receiver may not. Chen et al. [40] extended this basic model to observe that sender-receiver node pairs can have *incomplete* or *inconsistent* views of network topology. They argue that incomplete information leads to network inefficiency while inconsistent information leads to unfairness. They do not show, however, what network conditions lead to one or the other of these problems.

The best attempt we know of to describe the conditions that lead to inefficiency and unfairness is by Garetto et. al. [50]. They model the behavior of a set of four nodes consisting of two competing UDP flows. They model node interaction as a binary condition on each node pair indicating whether the pair is within transmission range of each other. Their approach leads to 16 topologies, which they classify into one of the four categories based on similar performance characteristics.

This chapter provides a new model of two-flow competition that extends this earlier work in three ways. First, we model traffic that can be sensed but not read. We show that typically at least 42% of the traffic a node senses is too weak to be read. The difference between readable and unreadable competing traffic is the



(a) Senders Can Read Each Other; Fair Bandwidth Allocation

(b) Senders Can Sense Each Other; Flow 2 Wins

(c) Senders Cannot Sense Each Other; Flow 1 Wins

Figure 2.1: Two Competing Flows

amount of time the sender waits before attempting to send again. The importance of this difference can be seen in the example in Figure 2.1, in which two 802.11 flows are either fair or unbalanced toward one or the other depending only on whether the two senders can read, sense or not sense each other's packets.

Second, we model both UDP and TCP traffic. The key difference between UDP and TCP is that TCP has a counter flow of transport-level ACK messages. We show that the presence of this counter flow is important to understanding the behavior of wireless congestion.

Finally, we consider flows where sender and receiver are close enough that the flow is resilient to noise generated by the competing flow.

In all, we characterize the two-competing flow scenario using 19 802.11g models (9 UDP and 10 TCP) that predict network performance based only on topology. We validate our models using simulation and experimentation in a lab testbed.

2.2 Modelling Methodology

Our work takes an approach based on simulation and experimentation. We first devised a set of node-topology parameters and node-performance characteristics. We then simulated every combination of topologies characterized by these parameters and grouped them by common performance characteristics. Finally, we used testbed experiments to validate topologies whose performance differed from previous work or which constituted interesting inflection points in the performance-parameter space. This section describes these parameters, characteristics and our assumptions.

2.2.1 Model Parameters

We consider three parameters in our model. The first is link state. A two-flow scenario consists of four nodes with six links between them. Two of these links are between the sender and receiver of each of the two flows, which are assumed to be in transmission range. The other four links — between senders, between receivers, and between sender and receiver of different flows — can have one of three possible link states: out of range, in sense-range only, and in transmission range. The other two parameters are traffic type and flow robustness. Taken together these three parameters and their possible values yield a total of 648 distinct network/traffic-type topologies.[1] The remainder of this section describes these three parameters in more detail.

[1] 3 link states; 4 inter-flow links; 2 traffic type; 2 interference levels. The number of scenarios is $3^4 * 2 * 2^2 = 648$.

Tri-State Link There are three states between any pair of nodes depending on the transmission power, carrier sensing threshold and background noise: (1) Transmission Range (TR), in which a node can clearly receive a packet from the other node; (2) Sensing Range (SR), in which a node can only sense the signal from the other node, but is not able to capture its packet correctly; (3) Out of Range (OR), in which a node cannot sense any signal from the other node at all.

Receiving a packet or sensing a packet has different impacts on the length of delay before a node sends its next packet. When a node is in transmission range of another node, it is able to set its Network Allocation Vector (NAV) correctly and then use Distributed Inter-Frame Space (DIFS) to contend for the airspace with the others. When it senses a packet whose payload it can not decode, however, it follows a different approach.

If the packet is sent by a 802.11g (or 802.11a) node that uses the standard Orthogonal Frequency Division Multiplexing (OFDM) modulation scheme, then the sensing node is unlikely to decode the Physical Layer Convergence Protocol (PLCP) header if it cannot decode the payload. This is because part of the PLCP header, the `SERVICE` field, is also encoded by the higher-rate modulation that the payload uses. When a node fails to decode the PLCP header, it still uses DIFS to schedule its next packet after it cannot sense any airspace activity. Thus, because the NAV was not properly set up, if the sensed traffic is actually a MAC data packet, this sensing node might not back off long enough to avoid interference to the returning MAC ACK packet.

Traffic Type We investigate the performance of two traffic types in all the topologies: (1) UDP traffic (2) TCP traffic. The key difference between UDP and TCP is that TCP flows consist of two sub flows, i.e., the TCP-`DATA` subflow and the TCP-ACK subflow.

Flow Robustness The distance between a flow's sender and receiver also plays an important role in a noisy environment. If the nodes are close enough, the signal

strength at the receiver is strong enough that the flow is resilient to most noise, while a distant sender provides a weak signal that is vulnerable to noise. We examine two points along this signal-strength continuum using the interference-level parameter which takes on two possible values: (1) Interference-Susceptible and (2) Interference-Immune.

2.2.2 Performance Characteristics

We classified the simulation results according to two qualitative performance metrics: *fairness* and *communication efficiency*. The first metric has three values: fair or unfair with one or the other of the flows dominating. The other metric classifies interference between the flows by indicating whether there is interference and, if so, which packets conflict: data packets send by flow senders, ACK packets sent by flow receivers or one type from each flow.

2.2.3 Assumptions

Finally, we make three important simplifying assumptions. First, we assume that link conditions are symmetric; that is node A has the same view of B's traffic that B has of A's traffic.

Second, we restrict our analysis to *two* flows under the belief that pair-wise interference is common enough to warrant isolated study and under the hope that these results will provide a building block for analysis of more complex topologies such as mesh networks [50].

Third, we assume that nodes out of sensing range do not interfere with each other's traffic. This assumption is built into common network simulators and approximates expected behavior. However, at higher sending rates, the assumption does not hold, though it is likely rate-adaption schemes incorporated on most 802.11 adaptors would lower sending rate if faced with significant interference, even from an otherwise invisible node. Experiments conducted in our testbed indicate that at a low sending rate of 6 Mbps (802.11g), any signal above -70 dBm would not be vulnerable to noise from a node that could not be sensed.

2.3 802.11g Models

We simulated each of the 648 802.11g scenarios characterized by our model; details of the simulation are presented in Section 2.4.3. As explained in paragraph 2.2.1, we assume in these scenarios a sensing node is not able to decode the PLCP header of the sensed packet and thus will not use EIFS to schedule its next transmission. By grouping performance-similar topologies together, we derive 9 UDP and 10 TCP models.

2.3.1 UDP Models

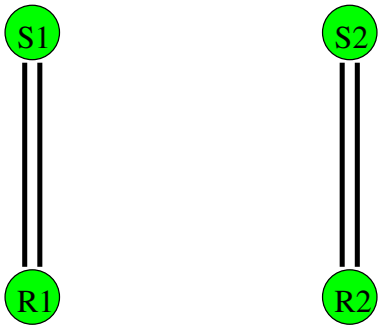
We begin with the models for two competing UDP flows. Figure 2.2 provides a graphical representation of each model and its legend is shown in Figure 2.3. Note that *symmetric sense* means that either two senders can *at least* sense the other receiver or neither does. There is no difference for a sender when it captures a MAC ACK packet or just senses the ACK — in either case, it will use DIFS to contend for the airspace after the MAC-ACK transmission finishes.

Base Cases.

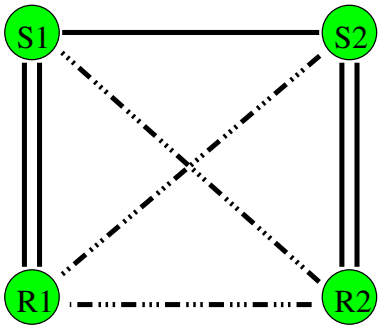
We begin with four initial models that exclude the possibilities that two senders are in sensing range and that a flow is robust to interference. These models are essentially the same as the four models of Garetto et. al.

UM_1: Independent For completeness we begin with the trivial model in which two flows that are sufficiently distant from each other that neither flow affects the other.

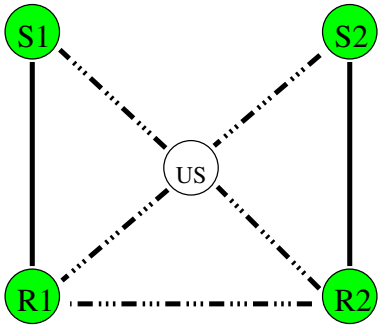
UM_2: Senders TR In this model the senders of each flow can read each other's packets. In this case, there is no interference between two flows and thus bandwidth is evenly divided regardless of the nature of other node relationships (receiver-receiver and sender-receiver) or signal strength (weak- or strong-signal links).



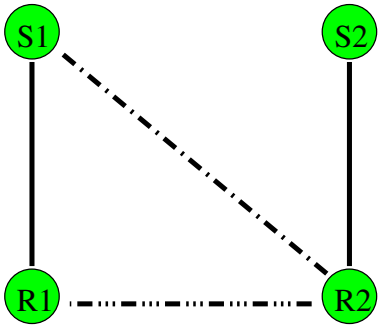
(a) Independent (Fair, No Interference)



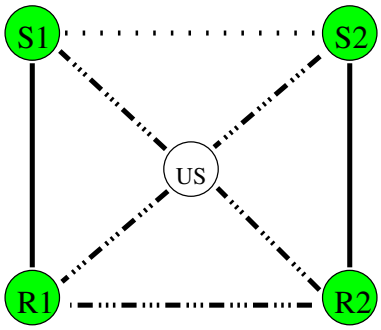
(b) Senders TR (Fair, No Interference)



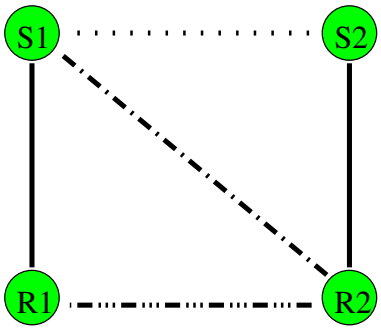
(c) Sym, Senders OR (Fair, Data-Data Collides)



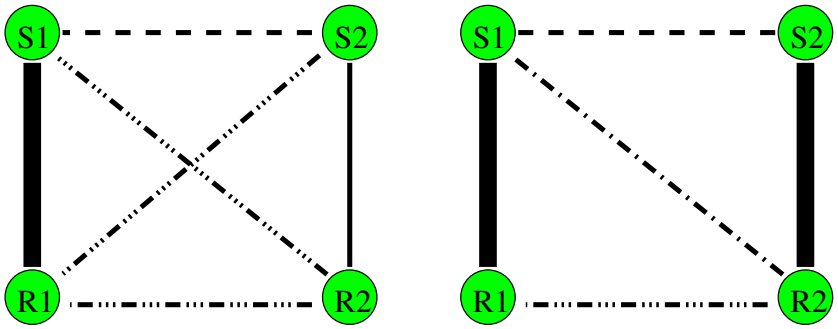
(d) Asym, Senders OR (Unfair, Flow 1 Wins)



(e) Sym, Senders SR (Fair, Data/MAC-ACK Collides)

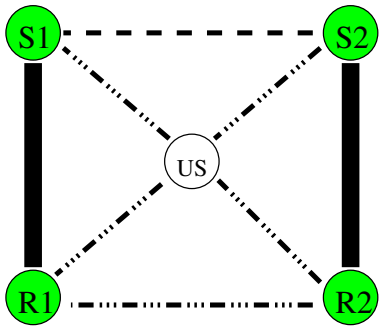


(f) Asym, Senders SR (Unfair, Flow 2 Wins)



(g) 1 Robust, Senders Not TR (Unfair, Flow 1 Wins)

(h) 2 robust, Asym Senders not TR (Unfair, Flow 2 Wins)



(i) 2 Robust, Sym Senders Not TR (Fair)

Figure 2.2: Models for UDP Fows

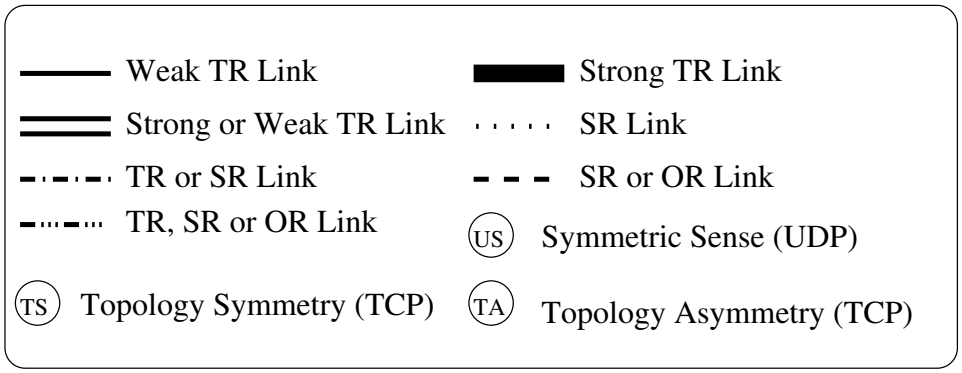


Figure 2.3: Legend for Models

UM_3: Symmetric Sense, Senders OR Two models are needed to capture the behaviour when sending nodes are out of range. The first model covers the case where the ability of a sender to sense the receiver of the other flow is symmetric. In this case, bandwidth is evenly distributed to the two flows, but aggregate bandwidth is reduced due to congestion-caused packet loss.

There are two types of packet loss. First, in the case where receivers can sense the senders of the other flow, then data packets from two senders can collide. Second, in the case where receivers cannot sense the senders of the other flow but two receivers are in sense range, then a DATA packet sent by one flow's sender can collide with a MAC-ACK packet sent by the other flow's receiver.

UM_4: Asymmetric Sense, Senders OR The next model covers the case where sending nodes are out of range and the sense relationship between flows is asymmetric. In this case, bandwidth allocation unfairly favors the flow whose sender is able to sense the other flow. The reason this flow gets more bandwidth is that when both senders send DATA packets at the same time, the packets sent to the receiver of the losing flow will be garbled but the other receiver sees only the packets from its sender.

Sensing Unreadable Traffic.

We now consider two new models in which senders can sense each other's packets but can not read them: one symmetric and one asymmetric.

UM_5: Symmetric Sense, Senders SR When the topology is symmetric, similar to UM_2, the two flows receive a fair bandwidth allocation. Unlike the earlier model, however, the fact that senders can only sense means that DATA packets can garble MAC-ACK packets from the other flow.

UM_6: Asymmetric Sense, Senders SR The next model is similar to UM_4, where a sender's ability to sense the other flow is asymmetric, but now senders are in

sense-only range of each other. As in UM_4, this model leads to unfair bandwidth allocation, but now with the other flow — the flow who's sender does not sense the other flow — winning.

The difference between this model and UM_4 is that senders can sense each other and so they do not send at the same time. They can not read each other's packets, however, and so they do not back off long enough to avoid overlapping with the MAC-ACK packet sent by the other flow's receiver. The asymmetry shown in Figure 2.2f indicates that the MAC-ACK packets of flow 1 are more likely to be garbled since sender 2 cannot sense receiver 1 at all, thus giving flow 2 the advantage.

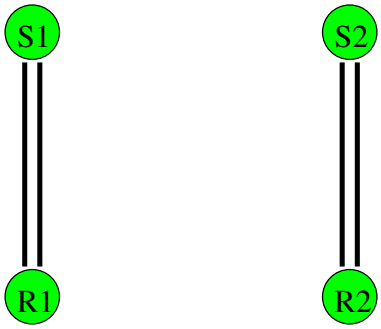
Robust Flows.

We now consider the three models in which one or both flows are robust to any noise.

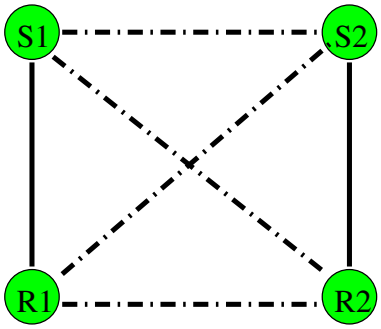
UM_7: One Robust Flow If only one flow is robust, it dominates the other flow whenever senders are not in transmission range.

UM_8: Two Robust Flows, Asymmetric If both flows are robust, senders are not in transmission range and the sense relationship between sender and receiver of different flows is asymmetric, then the flow whose sender senses the other receiver is dominated. This flow is disadvantaged because the sender avoids sending DATA packets when the other flow's receiver is sending MAC-ACK packets, even though the robustness of the two flows means that collisions will not occur.

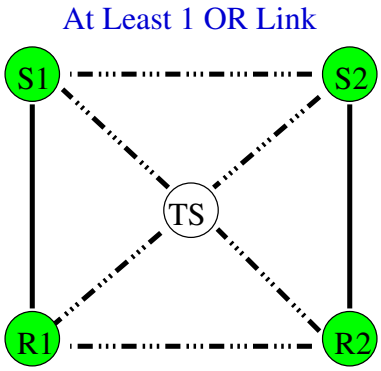
UM_9: Two Robust Flows, Symmetric When the topology is symmetric, bandwidth is shared equally between the two robust flows.



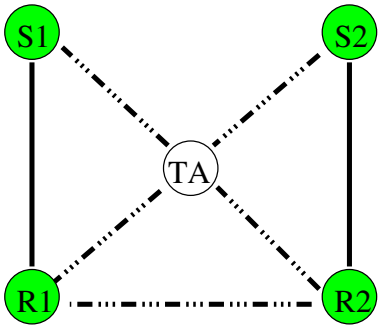
(a) Independent (Fair, No Interference)



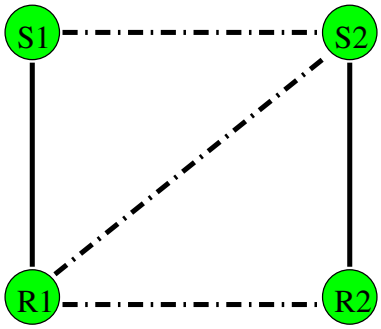
(b) All SR/TR Links (Fair, Little Interference)



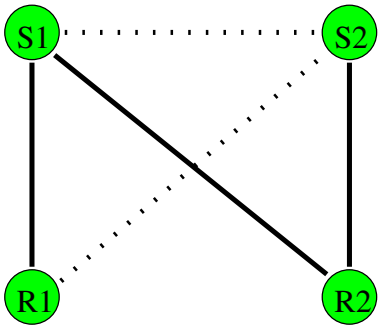
(c) Not All SR/TR Links, Sym (Fair, Interference)



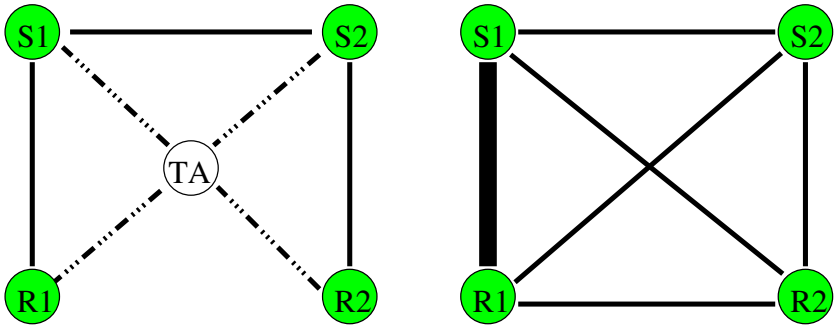
(d) Asym, Senders OR (Unfair, TCP Data-Data Collides)



(e) Asym, Sender Receiver OR Only (Unfair, TCP Data-ACK Collides)

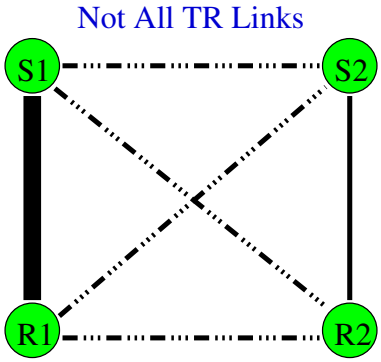


(f) Asym, Receivers OR Only, Senders SR (Unfair, EIFS Impacts)

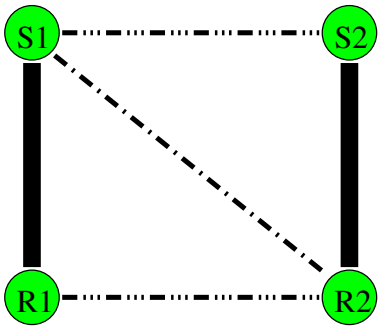


(g) Asym, Receivers OR, Senders TR (Unfair, TCP ACK-ACK Collides)

(h) Robust, All TR Links (Fair, No Interference)



(i) 1 Robust, Not All TR Links (Unfair, The Robust Flow Wins)



(j) 2 Robust, One Sender-Receiver Link Only (Unfair)

Figure 2.4: Models for TCP Flows

2.3.2 TCP Models

Modeling TCP is more complex due to the fact that each TCP flow consists of two sub-flows: DATA packets sent from sender to receiver and TCP-ACK packets sent from receiver to sender. TCP-ACK packets differ from MAC-ACK packets in the way that they are initiated. MAC-ACK packets follow reception of a DATA packet after a bound interval, but TCP-ACK packets are simply data packets to the MAC layer and are thus sent only when the channel is sensed to be clear. This difference changes the notion of cross-flow symmetry from that of UDP. *Unlike* UDP, TCP

flows are considered to be asymmetric when senders of both flows can sense the other flow's receiver, but one is in transmission range of the receiver and the other is in sense-only range.

We will first introduce the three fair models in this section. We then present four unfair models based on their different causes, followed by three other models that consider robust flows.

Fair Models.

There are three TCP models that result in fair network bandwidth allocation.

TM_1: Independent Again, for completeness, we begin with the trivial model in which two flows that are sufficiently distant from each other that neither flow affects the other.

TM_2: All SR/TR *Links* Model TM_2 requires that all pairs of nodes are at least in sensing range. In this model, two flows also experience little interference. This is because, considering any two TCP subflows in this model, they fit in either model UM_2 or model UM_5, which both result in a fair network allocation.

TM_3: Incomplete and Symmetric When not all pairs of nodes are connected, two TCP flows can still achieve fairness as long as the topology is symmetric. However, if at least two nodes are out of sensing range, there is a good chance that their packets will collide.

It is worth pointing out that TCP treats transmission failure as signal of network congestion and consequently reduces its sending rate. This behaviour is often considered as not desirable as wireless links are usually lossy, but it indeed alleviates the problems of congestion and signal interference. For example, given a symmetric topology in which the link between the sender of flow 1 and sender of flow 2 is not present, interference causes the collective UDP throughput to drop by 38%, while TCP flows suffer only 6% degradation.

Unfair Models.

All asymmetric topologies except those all-sensing cases result in unfair network allocation. There are four cases.

TM_4: Asymmetric, Senders OR When the senders are out of range, the main problem is collision between the two TCP-DATA subflows. The flow whose sender has less topology information loses. For example, if sender 1 and receiver 2 are out of range, then flow 1 loses just as in the hidden-terminal case for UDP flows.

TM_5: Asymmetric,Sender-Receiver OR *Only* When the only link missing is the link between the TCP sender of flow 1 and TCP receiver of flow 2, most packet collisions are TCP-DATA/TCP-ACK collisions. Flow 2 wins in this model because sending a TCP-ACK packet usually takes less time than sending a TCP-DATA packet. Therefore, when such collisions occur in the airspace, the probability of successfully retransmitting an ACK packet is higher than that of a DATA packet.

TM_6: Asymmetric, Receivers OR *Only, Senders* SR When the only link missing is the link between the two receivers, the only possible cause of asymmetry is when sender 1 and receiver 2 are in transmission range, but sender 2 and receiver 1 are in sensing-only range.

In this case, both flows suffer from TCP-ACK/TCP-ACK collisions. According to the 802.11 protocol, when such collisions occur, both TCP senders use EIFS to schedule the next transmission unless they can capture a packet clearly in the airspace. Therefore, in this model, the flow whose TCP sender can clearly capture the packet sent by the TCP receiver of the other flow wins.

TM_7: Asymmetric,Receivers OR*, Senders* TR If senders are within transmission range, EIFS no longer affects the fairness because the two TCP senders can reset EIFS to DIFS for each other.

This model is similar to model TM_4 except that its fairness is determined by

the worst TCP receiver not sender; the flow whose TCP receiver has less topology information than the other TCP receiver loses. For example, if sender 1 and receiver 2 are out of range, then flow 2 loses.

Robust Flows.

We now consider three models in which one or both flows are robust to any interference.

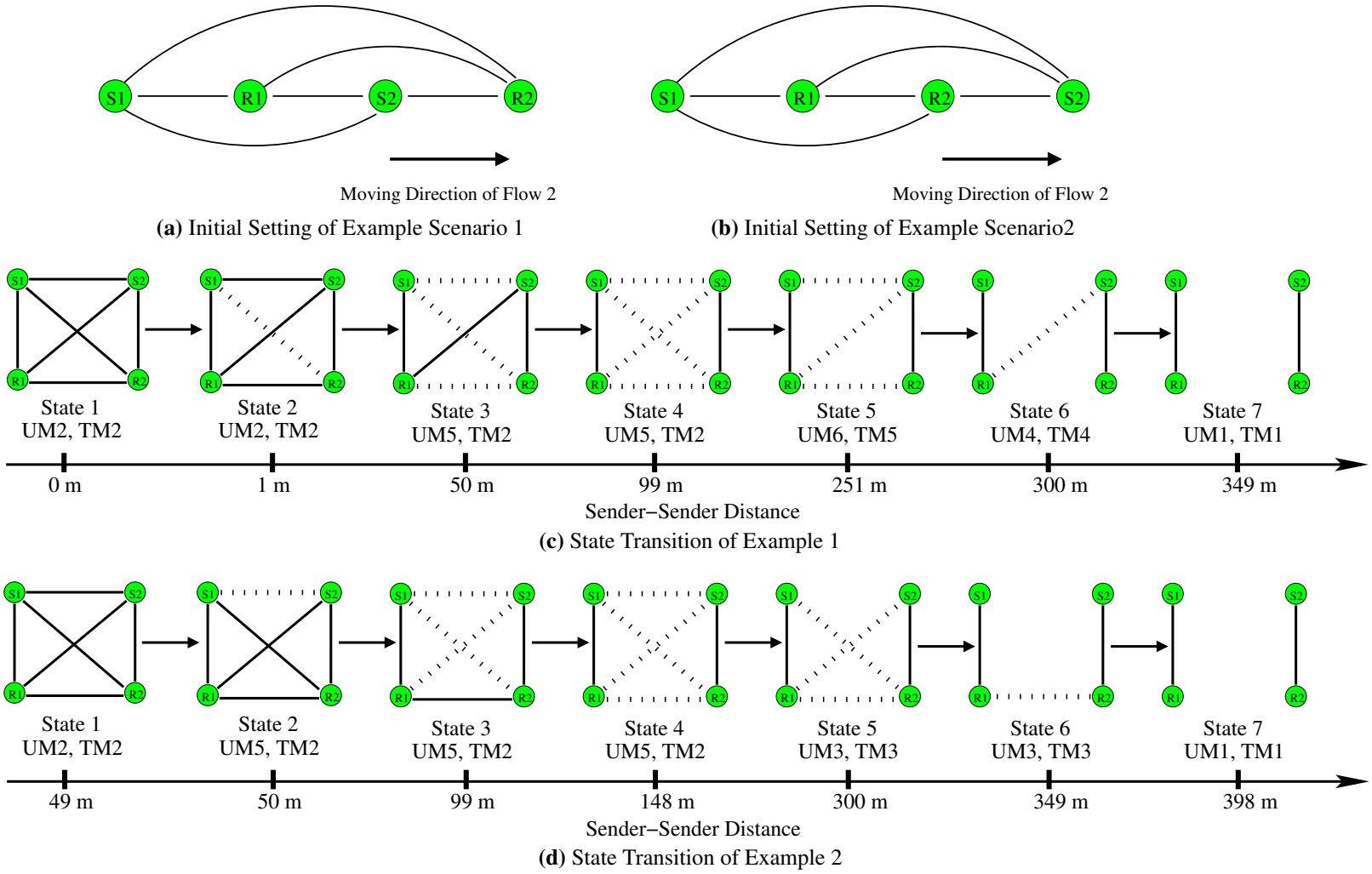
TM_8: One Robust Flow, All TR *Links* If flow 1 is the only robust flow but all four nodes are within transmission range, the network bandwidth can be still fairly allocated since every node has the complete knowledge of network topology and traffic.

TM_9: One Robust Flow, Not All TR *Links* If not all nodes are within transmission range of each other then packet collisions will occur. Flow 1 is not affected because of its strong signal strength while flow 2 has to back off and retransmit. Thus, flow 1 always wins.

TM_{10}: Two Robust Flows, One Sender-Receiver Link Only When both flows are immune to interference, there is only one unfair topology, i.e., when the TCP sender of flow 1 and the TCP receiver of flow 2 are connected. In this case shown in Figure 2.3j, flow 1 loses because sender 1 senses receiver 2's signal and thus cannot send as fast as sender 2.

2.4 Evaluation

We simulate all 648 scenarios using Glomosim [102]. To further ensure that the simulator is trustworthy, we also set up a controlled testbed that consists of four wireless nodes to study the interesting scenarios.



(a) Initial Setting of Example Scenario 1

(b) Initial Setting of Example Scenario2

(c) State Transition of Example 1

(d) State Transition of Example 2

Figure 2.5: Model Transition in Two Scenarios

For ease of exposition, we do not provide detailed results of the simulation. Instead, we present the results from the following two example scenarios that are capable of capturing behaviours of all the weak-link models but TM_6 and TM_7.

2.4.1 Example Scenarios

The four nodes are initially placed as illustrated in Figure 2.5a and Figure 2.5b, ensuring that all pairs of nodes are within transmission range. The only difference between these two examples is the placement of nodes S_2 and R_2. The nodes of flow 2 then gradually move away from flow 1, which weakens the signal strength between these two flows. This causes three links, i.e., (S_1, S_2), (S_1, R_2) and (R_1, R_2), to experience all three link states as the distance increases. We can see from Figure 2.5c and 2.5d that each example involves 7 state transitions until they completely move out each other's radio range.

2.4.2 Relationship to UDP/TCP Models

According to the model definitions in Section 2.3, we can convert the state transitions we simulate into model transitions as shown in Figure 2.5c and 2.5d.

Note that, due to their symmetric topologies, all states of scenario 2 belong to models that result in fair network allocations. We can therefore predict that the two flows can achieve the same throughput even though they might suffer from inefficiency problems caused by signal interference in some of the states.

Scenario 1, on the other hand, suffers not only from signal interference but also from unfairness problems in its 5^{th} and 6^{th} states. The topologies of its 2^{nd} and 3^{rd} states are also asymmetric. However, the flows will evenly split the network bandwidth according to the models they fall into.

We can thus predict that, if both flows are UDP flows in scenario 1, flow 1 wins in the state 5 and then loses in state 6 according to the analyses given in Section 2.3.1. If both flows are TCP flows, flow 2 wins in both states as explained in Section 2.3.2.

2.4.3 Simulation

The Simulator.

Glomosim [102] is a scalable simulator. But its physical settings are based solely on 802.11b and are not able to reflect the changes that have been incorporated into the 802.11g protocol. We thus implement a new physical layer based on the 802.11g specification [10] that includes changes such as transmission rates, duration of physical preambles, minimum CWnd size, signal extension duration, etc. We have also verified its accuracy by comparing our simulation results of one flow, either TCP or UDP, against that collected from our testbed using 802.11g cards; they are closely matched. We also ensure that Glomosim accurately models 802.11g's sensing behaviour as described in Section 2.2.1.

The transmission rate is set to the highest rate 54 Mbps in our simulations. The MAC-layer ACKs are sent at the 24Mbps rate, modelled after the Ralink drivers in our testbed. The minimum receiving signal strength is set to -58 dBm and sensing signal strength -76 dBm, corresponding to 50 meters and 300 meters in the two-way propagation model. The minimum signal-to-noise ratio is set to 18 dBm (receiving signal strength - sensing signal strength) so that nodes out of sensing range are also out of interference range. It is worth noting that, since our model only assumes three link states, our choices of distances of transmission range and sensing range are rather arbitrary. Each simulation lasts 900 seconds and repeats 10 times with different seeds.

Simulation Results.

The simulation shows that UDP and TCP performance differs significantly as seen by comparing Figure 2.6 and Figure 2.7. This results confirm the importance of modeling TCP separately from UDP.

Similarly, an example of the importance of distinguishing sensed traffic that can be read from that cannot is seen in scenario 1 by comparing states 2, 5 and 6. Performance of these states varies significantly, but the only key difference in their

topologies is the state of the link between the two sending nodes. The previous models that assume that all sensed traffic is readable would have mispredicted the performance of state 5.

UDP Flows.

We present the mean values of two UDP flows' throughputs in Figure 2.6; the standard deviations are low and thus not shown.

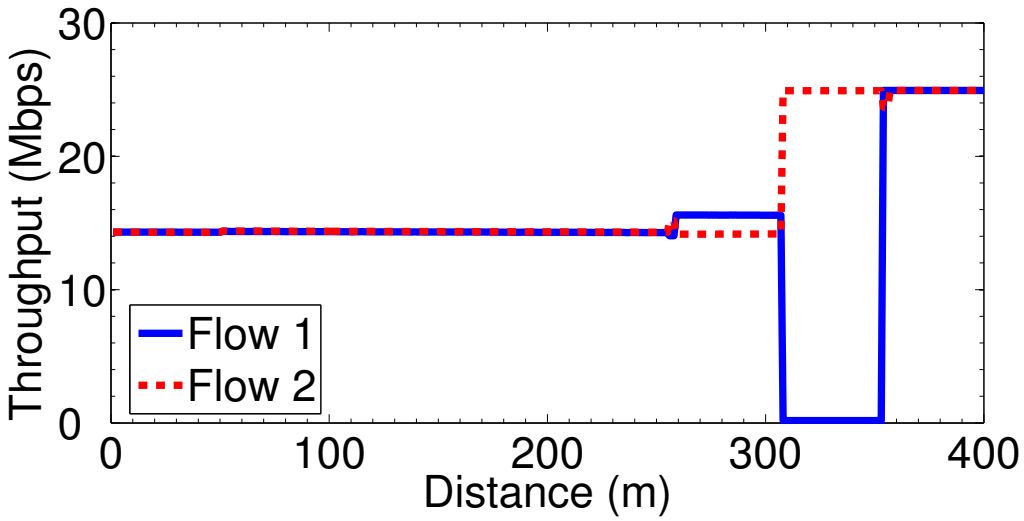
Scenario 1 We can see from Figure 2.6a that, in the first scenario, two flows first split the network bandwidth evenly and then experience unfairness as the distance between them increases: flow 1 wins at first and then loses. This is exactly what our models have explained and predicted.

Scenario 2 We can see from Figure 2.6b that two flows fairly share the network bandwidth throughout all the states in scenario 2 even if interference occurs. This is expected because all the topologies shown in Figure 2.5d are symmetric and thus fall into models that promise fairness.

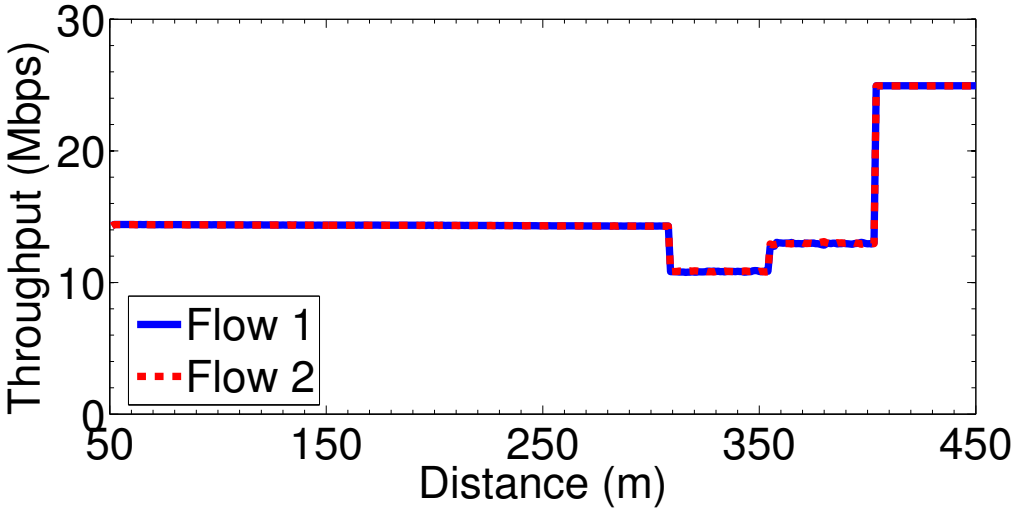
TCP Flows.

The TCP performance results are presented in Figure 2.7. Flow 2 wins in scenario 1 when interference starts to occur while both flows achieve fairness throughout scenario 2.

Scenario 1 When the distance between the senders is less than 250 meters, all the nodes are within sensing range and therefore all topologies belong to model TM_2, in which the two flows share the network bandwidth fairly and also cause little interference. Beyond 250 meters, sender 1 and receiver 2 move out of sensing range. The topology belongs to model TM_5 and thus, unlike the corresponding UDP scenario, flow 1 loses this time because the major collisions between sender 1 and receiver 2 are due to TCP-DATAs and TCP-ACKs. When the distance increases

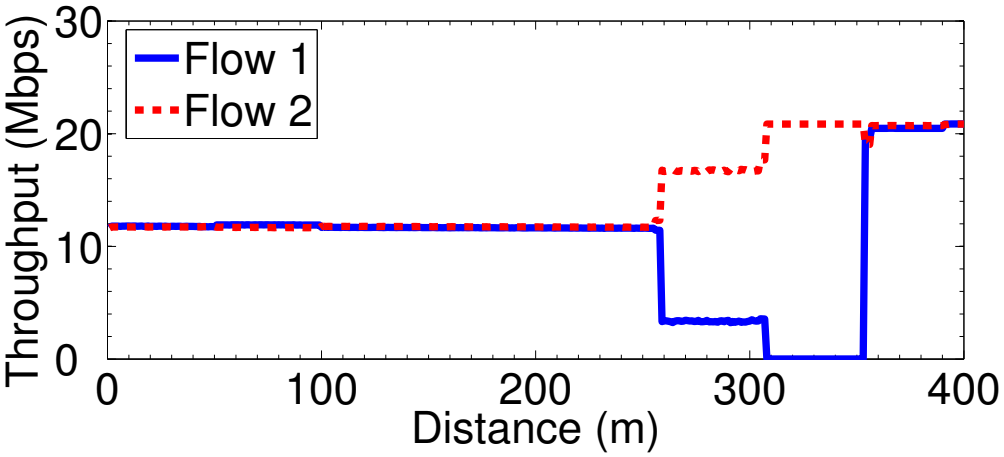


(a) 802.11g, Example 1

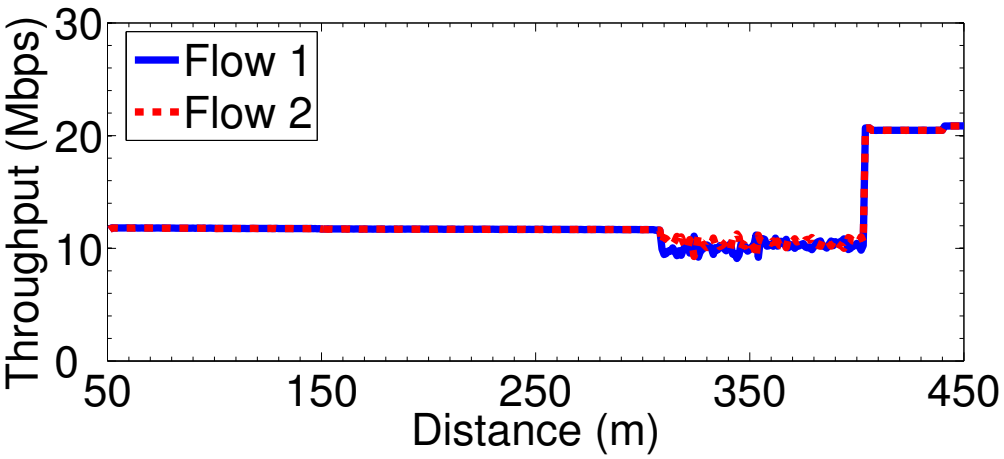


(b) 802.11g, Example 2

Figure 2.6: UDP Performance in Simulations



(a) 802.11g, Example 1



(b) 802.11g, Example 2

Figure 2.7: TCP Performance in Simulations

to beyond 300 meters, the two TCP senders move out of sensing range and thus the performance of flow 1 drops due to the TCP DATA-DATA collisions at receiver 1.

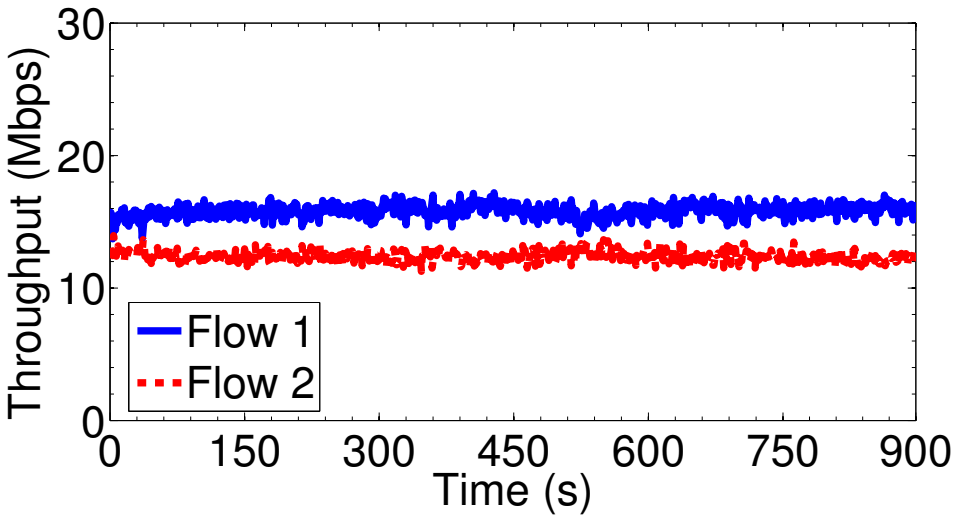
Scenario 2 Two flows fairly share the network throughout all topologies in scenario 2. As in the UDP scenarios, after the two senders move beyond 300 meters apart, the impact of interference starts to show. However, the performance penalty, compared to that of UDP flows, is much less, because TCP's sending rate is regulated by its ACKs. Lowering the sending rate can significantly alleviate congestion and thus reduce packet collisions. On the flip side, the performance of a TCP flow fluctuates due to its congestion avoidance scheme; the standard deviation can be as high as 800Kbps and starvations lasting for seconds are not rare.

2.4.4 Testbed Experiments

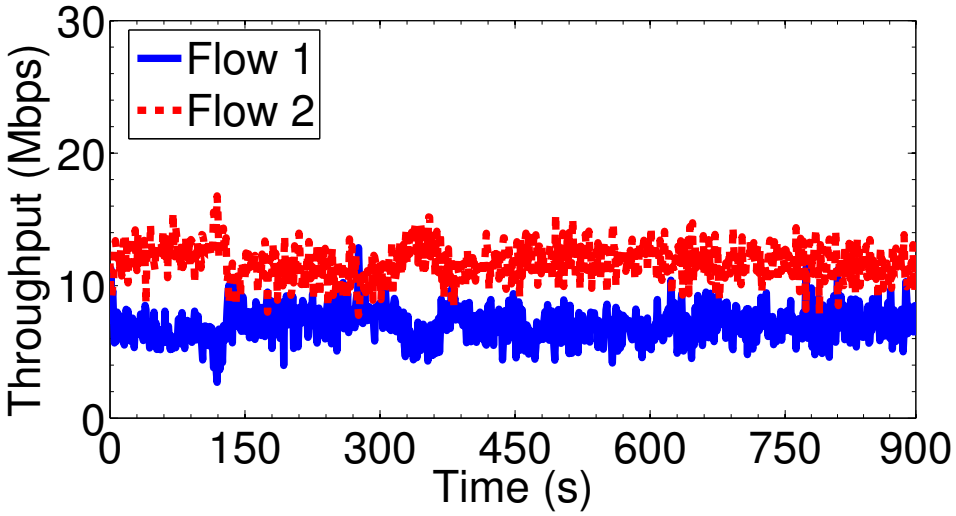
We set up a controlled testbed to verify the simulation results. The testbed consists of 4 desktop computers. Each desktop is equipped with one LinkSys WMP54G card that uses the Ralink 2560 chipset. Our testbed is running FreeBSD 6.1 Release. Our experiment is conducted in a $8m$ X $8m$ room. We use altered antennas to attenuate signal strength, and change the link state by adjusting transmission-power settings.

We confirm the link state settings (i.e., TR, SR or OR) before each experiment by letting two nodes broadcast messages with the same SSID. The two nodes are out of sensing range if they are able to transmit at full speed. Otherwise, if they transmit about half as fast but are not able to receive any messages, the link is in sensing state.

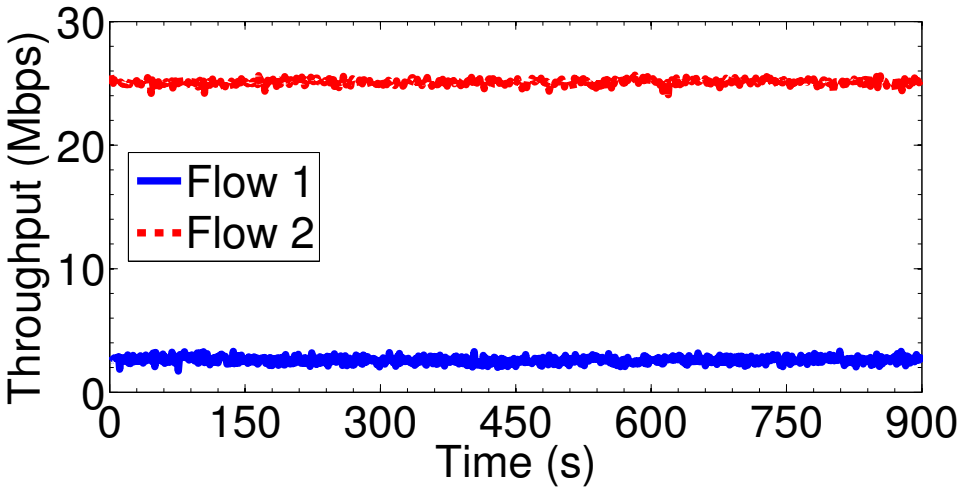
We set up snapshots of interesting topologies from the simulation using our testbed. All experiments were conducted late at night to minimize the impact of extraneous interference. The transmitting rate was fixed at 54Mbps. Each run lasted 15 minutes and the autorate function was disabled so that topology does not change during the runs.



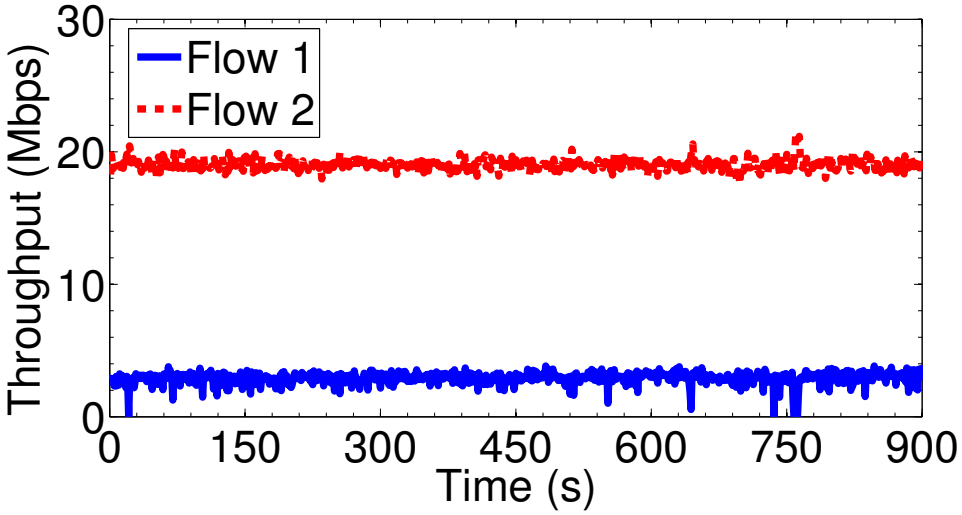
(a) UDP, Snapshot 1



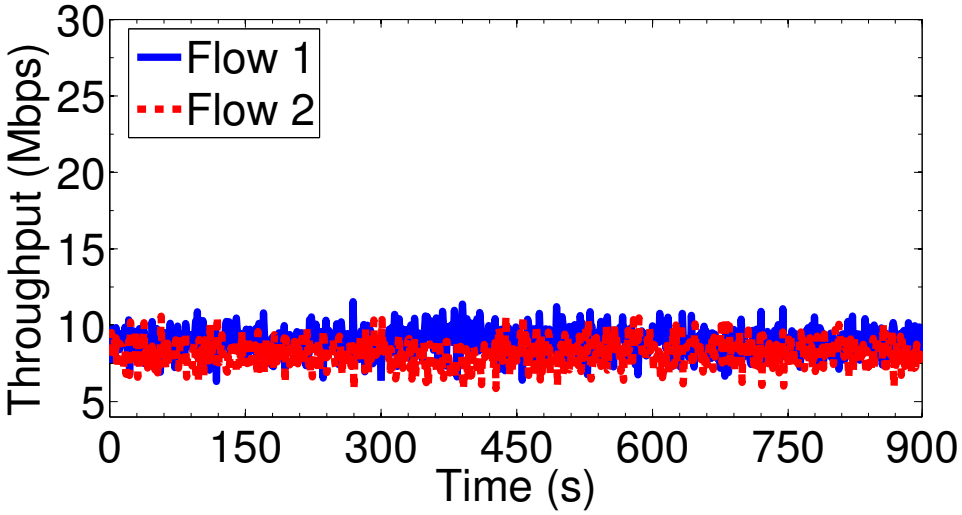
(b) TCP, Snapshot 1



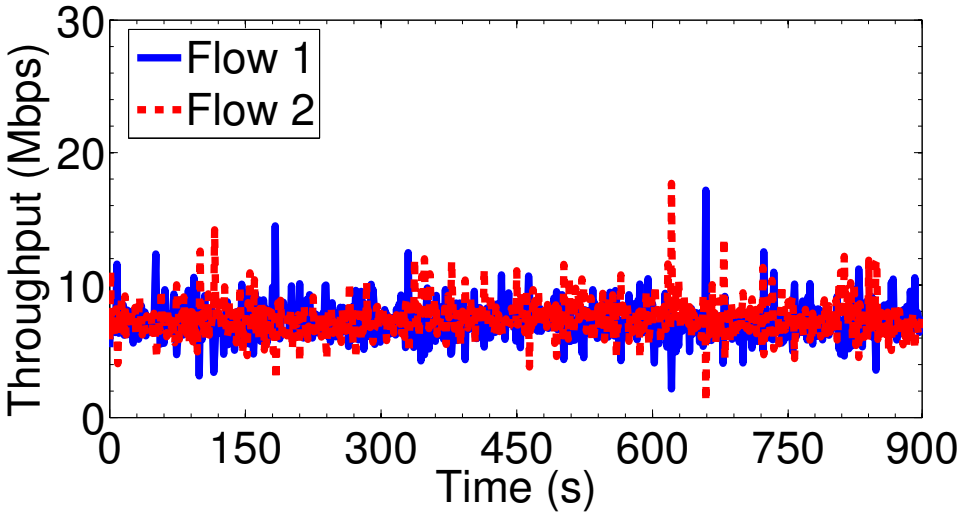
(c) UDP, Snapshot 2



(d) TCP, Snapshot 2



(e) UDP, Snapshot 3



(f) TCP, Snapshot 3

Figure 2.8: Performance of Snapshots

In this section, we report the results from three snapshots that correspond the the main inflection points in the simulation graphs from the previous section. We then show a state transition scenario using our testbed to illustrate the impact of link-state changes on flow performance in real networks. We repeat 3 runs in each setting and the mean values are reported. The biggest standard deviation is observed in fair TCP scenario, 1.4Mbps out of 7.4Mbps.

Snapshots.

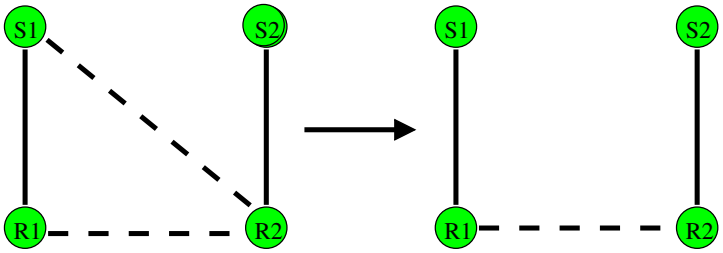
We set up the following three snapshots and show the testbed results in Figure 2.8: snapshot 1 — state 5 in Figure 2.5c; snapshot 2 — state 6 in Figure 2.5c; and snapshot 3 — state 5 in Figure 2.5d.

Comparing the testbed results against the simulation results shown in Figure 2.6 and Figure 2.7, we can see that they match closely, though the performance of TCP flows fluctuate more in the testbed. The testbed results also closely follow the predictions of our models.

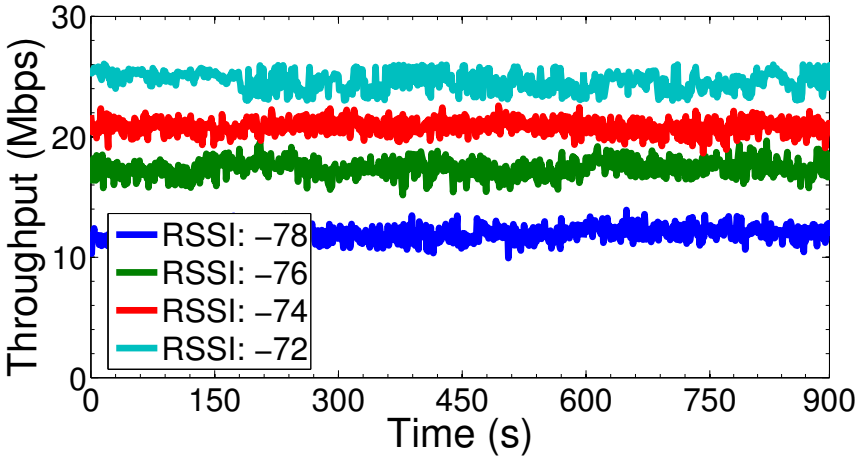
State Transition.

Figure 2.9a illustrates a transition experiment. The link between S_1 and R_2 is initially in the sensing state. We then gradually reduce their transmission power, and eventually put them out of sensing range. Note that this scenario actually corresponds to the transition of State 6 from Figure 2.5c to Figure 2.5d.

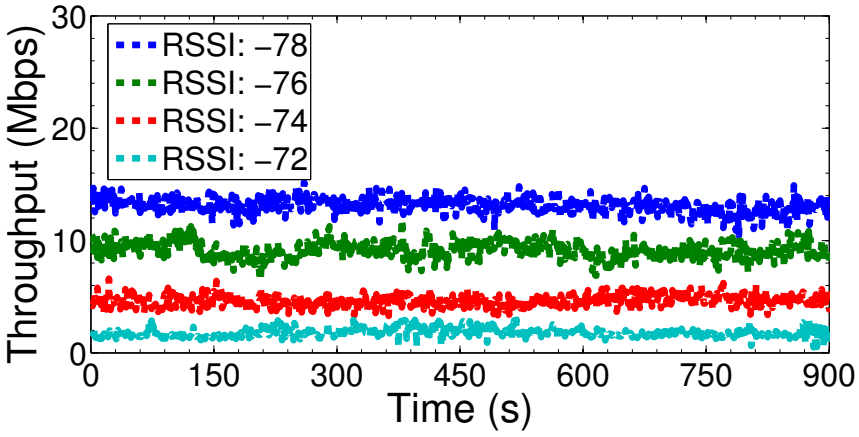
We observe the impact of this transition and report the throughput changes of both flows in Figure 2.9b and Figure 2.9c respectively. It is not surprisingly that this transition favors flow 2. The initial topology belong to model UM_5, in which flow 1 dominates because it garbles most of flow 2's packet at receiver R_2. As S_1's signal strength attenuates, the noise at R_2 is reduced. Thus, the performance of flow 2 gradually improves and that of flow 1 declines. The final state is a symmetric topology belonging to model UM_3, in which both flows equally split the network bandwidth. It is worth noting that our testbed can monitor these gradual link-state changes that simulators such as Glomosim fail to catch. Not



(a) State Transition (RSSI: -72dBm → -78dBm)



(b) Flow 1



(c) Flow 2

Figure 2.9: State Transition Experiment

shown in there, TCP follows the same trends and our testbed witnesses that as well.

2.4.5 Sensing Range

Finally, we conducted a different set of experiments to understand how frequently nodes sense traffic that is too weak for them to read. The difficulty in collecting this information experimentally is that 802.11 CSMA/CA is implemented in firmware and we cannot directly determine when a wireless adaptor is sensing traffic.

We thus used an indirect approach to measure sense-only traffic. We configured a machine with two Dlink DWL-G520 wireless adaptors using the Atheros chipset. One card is used to sense traffic by sending 1-byte messages at roughly 2-second intervals. We carefully measure the latency of each packet send to determine whether the adaptor sensed traffic and thus backed off at least once before sending the probe packet.

The other card operates in the *monitor* mode to passively capture all traffic readable by the card, regardless of its destination address. We carefully log the start and end time of every packet received by the card and correlate these times with the log generated by the first card. If we see that a probe packet was delayed at a time when the second card was not receiving a packet, then we conclude that this backoff is due to traffic that can be sensed but not read (in sense-only range).

We collected two 48-hour traces in two university labs in two buildings. We conservatively set the backoff-delay threshold to 350 μs — the longest possible 802.11g first-try back-off time without any optimization — 50 μs + 15 * 20 μs; the average delay in our traces is around 120 μs. We consider delays longer than this threshold to indicate that a packet was sensed or captured. The trace files show an average of 75% of the delays are due to sensing instead of packet receiving. Even in the worst hour in our trace, at least 42% of the backoffs are due to signal sensing. Among the entire data collected delays as large as 9.6 ms were observed, during which 18 packets were received.

2.5 Summary

In this chapter, we analyzed the scenarios of two competing flows and provided 19 concrete models that predict the performance and fairness based on node topologies, traffic type and channel conditions. In particular, these models considered three factors absent from previous work: (1) sensing state, (2) TCP flows and (3) weak or strong signals. We also validated our models via both simulations and real experiments in a testbed and the results show that they are accurate.

More importantly, our analysis demonstrates the dynamic nature of interference in 802.11 networks, where a slight change in the environment can completely change a device's performance. There are already 628 scenarios and 19 models when considering merely three network settings for two competing flows. There are still many other factors left out of our analysis since the analysis space will grow exponentially as more variants, such as rate adaptation and more wireless flows, are included for consideration. This interference complexity indicates that alleviating interference by adjusting just one environment setting will not work in all circumstances.

In the next chapter, we investigate the wireless interference problem from layers above the link layer, i.e., the TCP and IP layers. We show that network congestion is the immediate cause of severe interference and unfair bandwidth allocation. We then develop a strategy that mitigates WiFi interference by limiting IP-layer congestion.

Chapter 3

Achieve Fairness by Congestion Reduction

In Chapter 2, we showed that the wireless interference in live networks is complex and is hard to resolve in the physical or link layer alone. In this chapter, we propose a cross-layer approach called Shaper that uses TCP and a central router to achieve a fair bandwidth allocation among competing 802.11 devices.

The key idea of Shaper is to trade bandwidth for fairness by throttling throughput at a central router for the involved devices when the unfair bandwidth distribution problem is detected. As we discussed in Chapter 1, the impact of interference is limited as long as there is bandwidth available for TCP and 802.11 to retransmit and salvage the dropped packets. Only when all bandwidth has been used, do disadvantaged devices experience severe performance degradation, even suffering from bandwidth starvation in the worst scenarios.

Therefore, a promising solution is to use a shared, central router on the wired network to which all potentially interfering access points are connected, and to restrict the aggregate bandwidth delivered to these devices to prevent 802.11 network congestion from happening in the first place. This method mitigates interference by allowing 802.11 to schedule virtually every packet delivered into the wireless network, in effect using the wired router instead of 802.11 to dis-

tribute bandwidth among competing flows. Another important advantage of this approach is that it can be easily deployed in existing network management systems, requiring no change to the end-user devices or access points.

This chapter discusses the above idea in detail and is mostly based on our previous two papers [34, 36]. It starts with an introduction of our motivation and contributions in Section 3.1. Then we propose our throttling solution in Section 3.2. Section 3.3 presents the evaluation results in our testbed, proving Shaper is effective in dealing with wireless unfairness in a variety of scenarios. The main challenge of Shaper, interference detection, is discussed in Section 3.4, and finally we will summarize this chapter in Section 3.5.

3.1 Introduction

In the last chapter, we showed that wireless unfairness can easily occur in 802.11 networks. The culprits of this unfairness problem are twofold. First, the 802.11 protocol is not designed to support the cooperation of multiple autonomous systems. Whether a wireless device achieves its fair share of the network depends on the topology, channel conditions and other environmental factors. Let us consider the two well-known scenarios described previously, the hidden and exposed-terminal topologies, as shown in Figure 1.1. In the hidden-terminal scenario, signals sent by one sender can be corrupted by another, but not vice versa. In the exposed-terminal scenario, a sender is exposed to many other senders. Even if it does not experience any packet loss itself, it loses its fair share because it is forced to relinquish the airspace more often than the other wireless senders.

The second culprit and the immediate cause of this unfairness problem is *congestion in 802.11 networks*. Severe unfairness occurs *only when* more packets are pushed into the network than 802.11 can handle. In hidden-terminal scenarios, when the aggregate traffic load exceeds the 802.11 capacity and dominating flows request more bandwidth than their fair share, network utilization is so high that there is not enough bandwidth in the airspace for losing flows to recover from packet loss using TCP or MAC-layer retransmissions. Similarly, in exposed-

terminal scenarios, the exposed terminal is not given fair airspace access to send its packets. Frequent packet losses and long delays cause the TCP sender in a losing flow to push fewer packets into the network, at which point the flow is doomed to lose its fair network share to others.

Wireless networks are getting more congested than ever. Hundreds of millions of APs and 802.11-enabled laptops have been sold worldwide [18, 22]. Wireless traffic has also grown faster than the substantial increases in bandwidth. Previous research [55] has shown that wireless users use bandwidth-greedy applications just as they would on wired connections. As the capacity of the network backbone that connects APs to the Internet increases to Gigabit ethernet, congestion is even more likely to occur on the last-hop wireless links. Collected traces have shown that, even in well planned wireless networks such as hotels [83] and university buildings [41], wireless users often suffer from performance degradation caused by packet collisions. As the congestion problems increase, unfairness is more likely to happen.

3.1.1 Motivation: A Cross-Layer Solution to Unfairness

The correlation between 802.11 network congestion and unfair bandwidth distribution leads us to a surprisingly simple cross-layer solution to addressing the unfairness problem: throttle traffic above the 802.11 MAC layer, at an upstream router. We argue that as long as fewer packets are pushed into the airspace than 802.11's capacity can handle, standard TCP and fair queuing will together allow 802.11 to achieve fairness.

This cross-layer approach is effective because throttling the aggregated throughput below the network capacity slows winning flows and thus grants losing flows the extra airspace they need to deliver packets and recover from packet loss. Successfully delivered packets in turn open up the TCP sending window for a losing flow. Consequently, a losing flow is now able to push more packets to its receiver. Packets from all flows are queued at the upstream router and are treated equally by the fair queue management mechanism. This feed-back loop continues until a

max-min fairness is achieved between flows.

The key idea of our approach is to prevent 802.11 from making bandwidth allocation decisions. We limit the number of packets pushed to the shared airspace so that the access points are able to deliver/receive all of them. The bandwidth allocation decision across multiple AP domains is thus designated to the upper layers at the router.

3.1.2 Contributions

The contributions of this chapter are as follows. First, we highlight the often neglected fact that MAC-layer congestion has to be dealt with before addressing unfairness in 802.11 wireless networks. We provide a prototype solution, called Shaper, that ensures fairness by throttling traffic at upstream routers. An interesting aspect of this solution is that it can address a challenging wireless problem by manipulating the traffic over the wired backbone.

Second, we show how to address the unfairness problem across multiple access point domains in a large and cooperative wireless network, where a centralized router can control the in and out traffic limits for wireless devices. Wireless networks deployed in hotels, airports, business enterprises and university campuses all fit this model. Instead of designing new MAC/physical protocols [43, 84, 90], introducing complex wireless fair queuing algorithms [68, 72, 80] or adaptation schemes [43, 95], Shaper makes adjustments at the network layer using the off-the-shelf traffic regulating and fair queuing shemes and requiring no modifications to 802.11. It is thus easier to deploy.

Finally, we present an in-depth discussion of the system design and challenge. We have already implemented a prototype and used our testbed to verify that our approach is indeed an effective solution to the wireless unfairness problem regardless of the topology.

3.2 Proposed Solution

We argue in this thesis that standard 802.11, TCP and a fair queue are enough to provide fairness even under unfavorable network conditions as long as we can prevent wireless congestion from happening. This section describes this idea in more detail.

3.2.1 Assumptions

We make three assumptions in Shaper. Shaper is designed for use in centrally administrated networks such as large-scale campus wireless networks and other cooperative networks, and thus we assume Shaper has control over the central router to which all of the APs connect. We also assume that the Shaper has access to some global information, including AP association lists, physical AP positions, etc.

We consider only the traffic delivered to or received from the Internet, including both down-link and up-link traffic. Shaper resides in a router and needs to see all of the traffic. If the two end hosts of the same flow are in the same subnet, a router does not see the traffic and thus shaping will not occur. Finally, we only consider TCP traffic, the dominant traffic on the Internet. Traffic throttling has no impact on UDP senders.

3.2.2 Overview

Given the above assumptions, when Shaper detects that the distribution of network bandwidth is unfair, it will regulate the in and out traffic limits for those APs using the shared edge router they are connected to. This throttling scheme must meet two requirements. First, it must guarantee that the aggregate throughput does not exceed the network capacity. Second, it should slow the dominating flows so that the relinquished bandwidth can be assigned to the starved flows.

As an initial test of our approach, we repeated the hidden terminal experiment shown in Figure 1.1. This time, we set up a simple traffic shaping rule at the

router connecting S1 to R1, which throttles S1's throughput to a maximum of 10 Mbps. With this simple change, the two flows evenly split the network bandwidth, averaging 10 Mbps each, instead of the initial 19 Mbps to 1 Mbps split.

This result shows that, using our approach, shaping traffic at layer 3 can be an effective solution to wireless unfairness between TCP flows. The obvious tradeoff is that network utilization and round-trip latency may suffer.

3.2.3 The Details: A Two-Component Scheme

The shaping procedure consists of two components to ensure that the losing devices can deliver as many packets as the others: *throttling* and *fair queuing*.

Throttling

The first component is to throttle the throughput below the aggregate network capacity in a neighbourhood in order to reduce the chance of simultaneous media access. Once Shaper detects that unfairness is occurring in a neighbourhood consisting of two or more APs, it needs only to throttle the aggregate throughput of the access points in that area, not specific flows or end-user devices. This is effective since each access point is the central point in its domain; each packet sent in the WLAN is either sent to or from an AP.

Queuing

Throttling traffic for specific access points is not enough to address unfairness since it alone does not define how bandwidth is distributed between flows. If the throttled network load remains high and packet loss remains possible due to interference or network level congestion, then competition between flows will continue to lead to unfairness in favor of the winning flows.

One approach to solve this is to throttle only the dominating flows. But, this would require Shaper to identify which dominating flows are causing problems to the losing ones. Moreover, tracking over hundreds of thousands of flows in a large-scale network raises scalability concerns. By contrast the fair queuing

approach requires no per-flow analysis or flow management.

Fair Queue. Fair queue management algorithms such as RED [47] and SFQ [70] can provide statistical fairness on a per-flow basis with low overhead. For example, if the aggregated throughput is throttled down to B Mbps and there are N flows routing through the router, then each flow will be granted a throughput of $\frac{B}{N}$ Mbps if needed. Our prototype uses SFQ.

Non-Overflow FIFO *Queue.* Configuring a FIFO queue to never overflow can also guarantee per-flow fairness. If there are no dropped packets, the TCP window for a sender will continue to grow until it is capped by the sending/receiving buffer size or the bandwidth-delay product. At this point each TCP flow becomes ACK paced and has a full window size of packets in flight queued at the router. Fairness is thus achieved. However, a non-overflow queue is not practical since it requires a queue large enough to buffer all in-flight packets. Furthermore, if the channel conditions cause some flows to drop more packets than the others, then FIFO is not able to achieve perfect fairness either.

3.3 Evaluation

We have set up a controlled 10-node testbed to verify the effectiveness of our prototypes, and we present the results in this section.

3.3.1 Testbed and Traffic Set-up

Our testbed consists of six desktops, three LinkSys APs and one router in an $8m$ x $8m$ room. Each desktop is equipped with either a Ralink or Atheros-chipset wireless card. We use altered antennas to attenuate signal strength, and change the network topology by adjusting transmission-power settings. The desktops run Linux kernel 2.6.21.1 with high-resolution timers enabled; the system clock frequency is set to 1000 Hz, and the APs run OpenWRT. The transmitting rate

is fixed at 54 Mbps. The autorate function is disabled so that topology does not change during the runs.

We use Mxtraf (part of the QStream project [7]) to generate long-term TCP flows between pairs of desktops. In each flow, exactly one desktop, the Mxtraf client, is connected to an 802.11 AP and the Mxtraf server continuously pushes TCP packets to the client as fast as TCP allows. All traffic is configured to be routed through the sole router in the testbed. The throughput of each flow is measured at the IP level reported by Mxtraf.

We use up to three competing flows in these experiments. However, it is worth noting that the effectiveness of Shaper is independent of the number of flows. It depends only on the correlation between 802.11 network congestion and unfairness. Therefore, the traffic Shaper should remain an effective approach in addressing unfairness for many-flow scenarios.

3.3.2 The Topologies

We use the testbed to set up the two unfair topologies shown in Figure 1.1 for evaluation. Shaper helps the losing flow to achieve fair throughput in both cases by alleviating the congestion level in the airspace. Since the impact of throttling and fair queueing on fairness in both scenarios are the same, we will focus only on the hidden terminal topology to present our evaluation results. However, demonstration of the impact of Shaper on the exposed terminal scenario is posted online.[1]

We confirm the topology and link state settings before each experiment by letting two nodes broadcast messages with the same SSID. Since we do not have direct access to the device firmware, we infer link state using the following heuristics. The two nodes are out of sensing range if they are able to transmit at full speed. Otherwise, if they transmit about half as fast but are not able to receive any messages, the link is in sensing state. When two senders send at full speed, we validate that one sender causes interference to the other receiver by checking the throughput information and the number of retransmissions of the other flow.

[1] http://www.cs.ubc.ca/~kcai/threeAPs_shaping.html

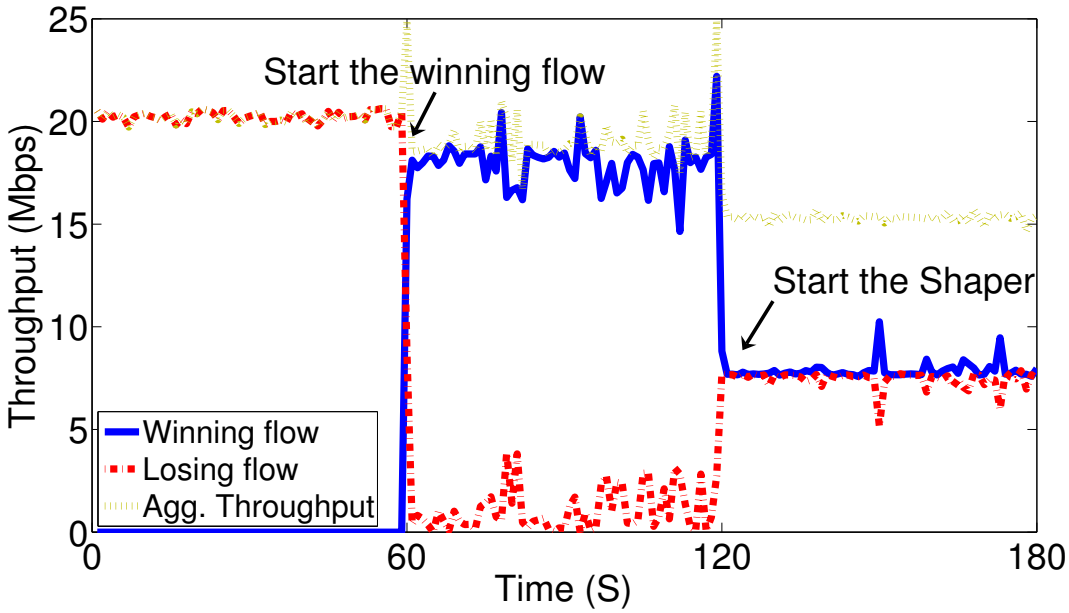
All experiments are repeated three times in our controlled testbed and are also conducted late at night to minimize the impact of extraneous interference. The difference between runs is found to be low and so only the median values are reported. The results shown in the figures have been smoothed over one-second periods.

3.3.3 Effectiveness of Shaper

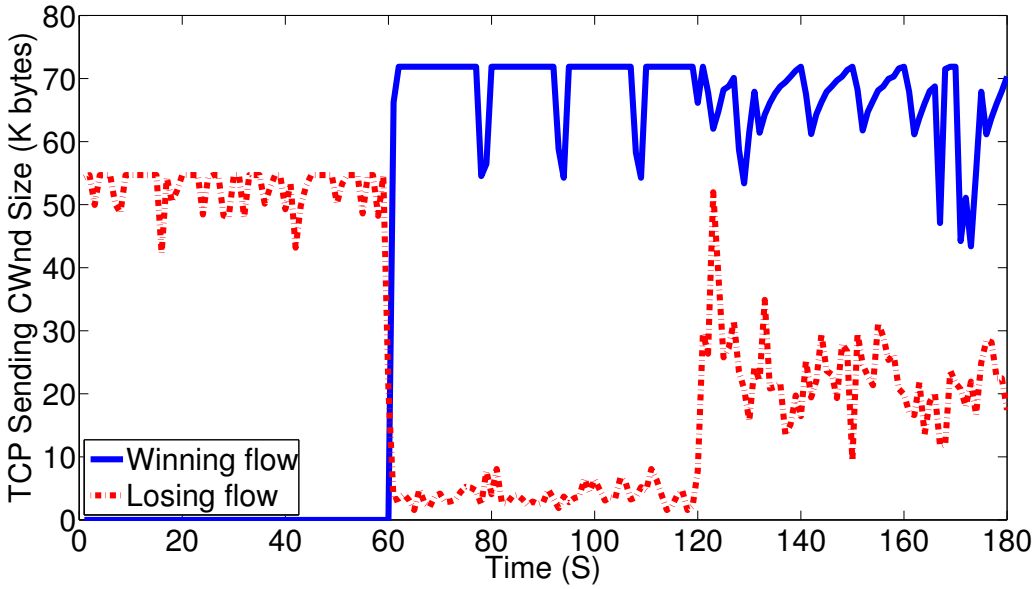
The effectiveness of our shaping approach is illustrated in Figure 3.1. This experiment consists of three one-minute phases. The first (losing) flow starts as the only flow in phase one. A second (winning) flow starts in the second phase (at 60 seconds) when the two flows begin to compete with each other for the airspace. At this point, the network bandwidth distribution is decided by the 802.11 and TCP protocols. In phase 3, Shaper starts to throttle the aggregated bandwidth down to 16 Mbps. For these experiments, Shaper uses Hierarchical Token Bucket (HTB) [4] for traffic throttling and deploys SFQ for queue management. HTB burst is set to 0; no other change is made.

We can see from Figure 3.1a that the first flow achieves 20 Mbps throughput alone. However, immediately after the second (winning) flow starts, the first (losing) flow's throughput drops to around 1-2 Mbps. Recall that this unfairness is due to the asymmetric topology where the winning flow can corrupt the losing flow and not vice versa. At certain times in phase 2, the losing flow is completely starved, i.e., its TCP sender is forced to delay its transmission using the TCP-layer exponential back-off timer. The winning flow, on the other hand, takes all the bandwidth: it achieves a throughput of 18.5 Mbps on average. This bandwidth disparity shows that 802.11 and TCP are not able to guarantee fairness when nearby wireless flows from different AP domains compete.

Figure 3.1a also shows that Shaper is effective in addressing unfairness. As soon as we enable Shaper, the two flows split the bandwidth nearly 50-50. The throughput of the losing flow is increased from 1 Mbps to nearly 8 Mbps. Note that the aggregate throughput is limited by the Shaper to 16 Mbps. This trade-off



(a) The Impact of Shaper on Throughput



(b) The Impact of Shaper on TCP Sending Cwnd

Figure 3.1: The Effectiveness of Shaper

between network utilization and fairness will be discussed later in Section 3.3.7.

Figure 3.1b illustrates how this works. The sending windows of both flows are capped by the bandwidth-delay product, at 54 Kbytes and 71 Kbytes respectively. The difference in window size is due to the TCP window auto-tuning feature in the kernel [46, 85]. When the losing flow is the only flow using the airspace, its sender is able to keep a window size well above 40 Kbytes, and thus it has enough packets to saturate the pipe to its receiver. Note that the losing flow has quite a lossy wireless channel. Even if the 802.11 and TCP retransmission schemes can quickly recover from packet loss, its TCP window is reduced every time a TCP packet is lost.

In phase 2, the losing flow's window size is cut down to 1-3 Kbytes, about the size of only one or two packets. This is because the winning flow's packets have saturated the shared airspace, causing the losing flow's packets to be dropped all the time. This constant packet loss in turn causes the losing flow to halve its sending window until it reaches the minimum window size (i.e. one packet), when the TCP exponential back-off kicks in. This window reduction, backoff scheme is designed to avoid network congestion at routers in wired networks, however, it will not help the losing flow in the face of wireless packet loss. In fact it causes the losing flow's performance to drop even more. The winning flow's sender is transmitting at full speed with a full-size window — we can see a clear ack-paced pattern for it.

Shaper limits the number of packets that can be pushed into the shared airspace below the 802.11 capacity. It also uses fair queuing to further cut the winning flow's throughput by ensuring the losing flow is de-queued fairly at the router. With the extra unused airspace, the losing flow is more likely to recover from packet losses using MAC and TCP-layer retransmissions. We can see from Figure 3.1b that increasingly successful packet deliveries allows the losing flow's retransmission window to increase 15 fold. In this run, it is worth noting that Shaper also makes the winning flow experience more packet losses due to increasing noise from the losing flow; its sender, however, can easily recover these

packet drops to keep the sending window high.

3.3.4 Impacts of Throttling

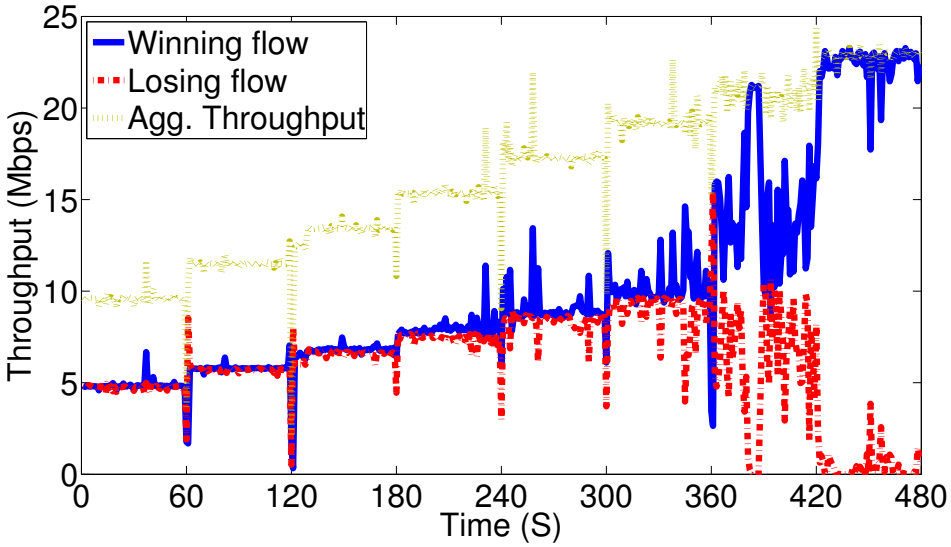
We have claimed in this chapter that the aggregate throughput has to be throttled below the network capacity (24 Mbps for TCP flows in our testbed) in order to make Shaper effective. In this section, we validate this assertion. Shaper begins with an aggregated throughput of 10 Mbps for these two flows, and then gradually increases the cap by 2 Mbps per minute to 24 Mbps.

We can see from Figure 3.2a that Shaper is able to distribute the aggregated bandwidth evenly between these two flows as long as the aggregated bandwidth is under 20 Mbps, almost 85% of the best throughput that 802.11g can offer to one TCP flow. If Shaper allows the aggregate throughput to grow beyond that point, the winning flow has more packets to send and thus is more likely to corrupt the losing flow's packets. As soon as TCP and MAC retransmissions fail to recover those packet losses over the wireless link, the losing flow will suffer from a significant performance drop. Figure 3.2a shows that the losing flow gets roughly one third of of the 20-Mbps aggregated throughput, and is almost shut out after that.

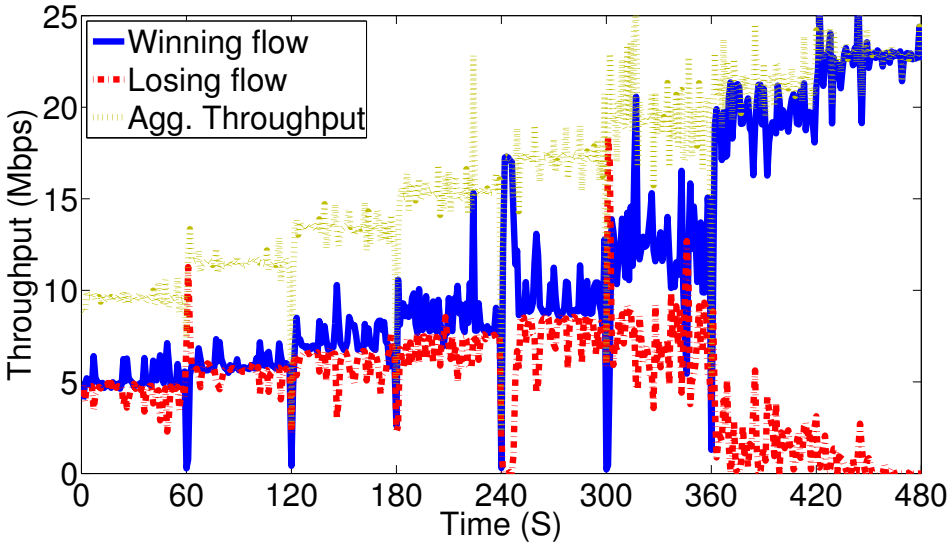
3.3.5 Impacts of Fair Queuing

We've also argued that using a fair queue is necessary in order to provide better fairness between flows when packet loss remains possible. In this section we examine how the throttler performs without fair queuing.

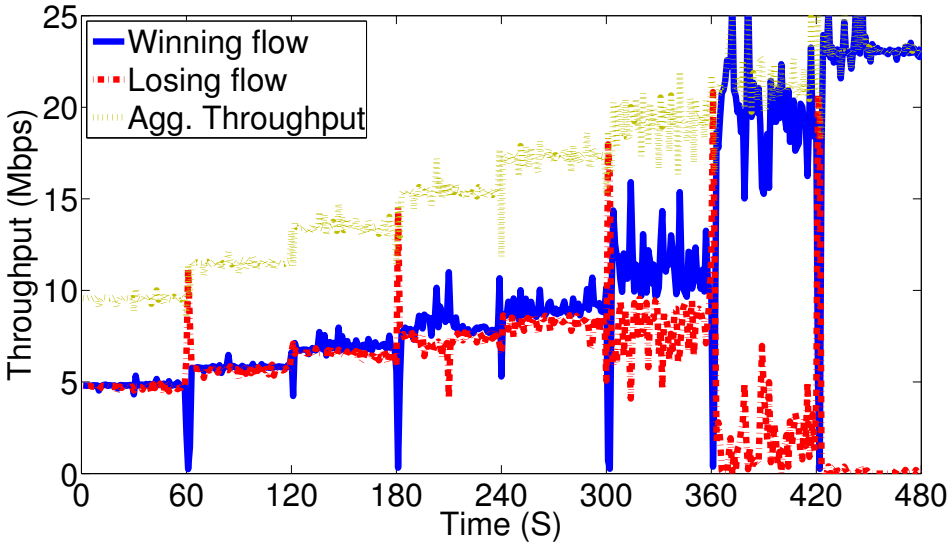
We repeat the hidden-terminal experiment with both both SFQ and FIFO queuing. The experiment is repeated in two environments: in one, two flows have similar channel conditions where the probabilities that the two flows drop packets due to background noise or propagation errors are almost the same. We calculate the probability by counting the number of TCP-layer retransmissions for both flows when they are the only flow in the network. In the other, the losing flow experiences a more lossy channel than the winning flow. The losing flow alone can sustain 20-Mbps throughput in both cases though. No queue overflow occurs.



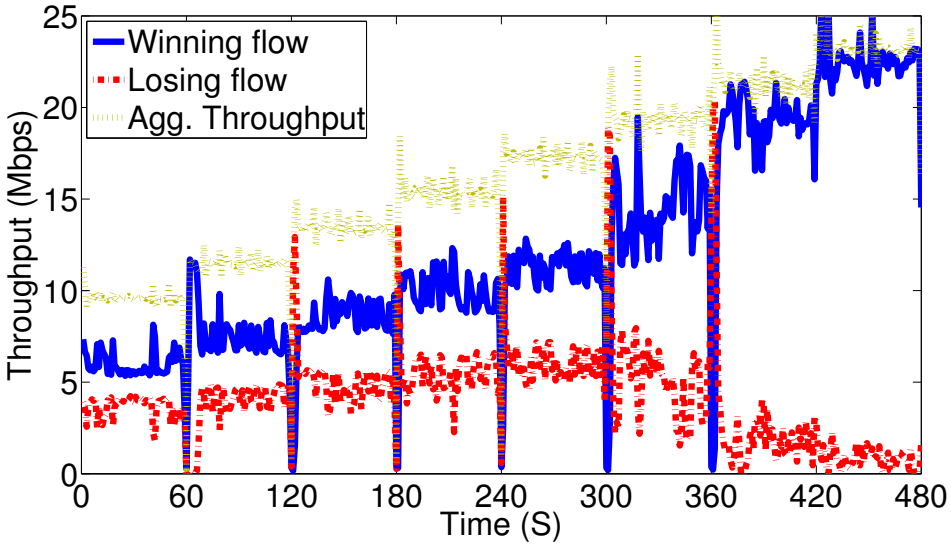
(a) When Using SFQ Queueing Scheme, Two Competing Flows That Have Similar Channel Conditions Can Achieve Perfect Fairness Until Airspace Is Congested.



(b) When Using FIFO Queueing Scheme, Two Competing Flows That Have Similar Channel Conditions Can Achieve Fairness Until Airspace Is Congested.



(c) When Using SFQ Queueing Scheme, The Losing Flow That Has Worse Channel Conditions Can Still Achieve Fairness When Competing With The Winning Flow Until Airspace Is Congested.



(d) When Using FIFO Queueing Scheme, The Losing Flow That Has Worse Channel Conditions Can No Longer Achieve Fairness When Competing With The Winning Flow. But It Still Benefits From Throttling by Reaching A Higher Throughput Than It Would Have Before Congestion Occurs.

Figure 3.2: The Impact of Throttling, Fair Queuing and Channel Condition on Unfairness in Hidden Terminal Topolgies

The results are shown in Figure 3.2.

We observe two obvious differences between SFQ and FIFO. First, with FIFO, flow throughput fluctuates significantly more than with the fair queue. Second, FIFO does not ensure fairness among the flows while SFQ does. We can see from Figure 3.2b that FIFO is capable of ensuring fairness when competing flows face similar channel conditions. However, when channel conditions differ, for example when one flow has a more lossy channel than the other, FIFO is unable to keep the losing flow from suffering unfairly. This effect is shown in Figure 3.1d. The two flows can achieve 5.85 Mbps (std. 1.03) and 3.58 Mbps (std. 0.75) throughput respectively when the cap on the aggregated throughput is 10 Mbps. But when the cap increases to 18 Mbps, the throughput difference between these two flows doubles although the losing flow's throughput is also increased to 6.18 Mbps (std. 1.44). Beyond that point, the losing flow's throughput starts to drop and fairness deteriorates.

The reason behind this imperfect fairness is that, when the losing flow drops more packets than the winning flow due to increasingly lossy channel conditions, its sender will reduce its sending rate. The FIFO queue thus always contains more packets from the winning flow, and thus delivers more of its packets to the wireless network. On the other hand, the fair queue mechanism divides the throughput between flows regardless of the proportion of their packets queued at the router, as long as no queue is empty. Therefore, increased packet loss does not change the losing flow's packet injection rate, which ensures both flows continue to receive fair access to the wireless network.

3.3.6 Prototype Implementation

Shaper is a three-stage, adaptive control system consisting of three functional sub-systems: *monitor*, *diagnoser* and *throttler*. The monitor sub-system continuously monitors the health of the wireless network and collects traces and statistics for the diagnoser. Based on information from the monitor, the diagnoser infers whether network congestion and unfairness have occurred. If so, it instructs the throttler

to shape the traffic of the affected access points. We have implemented two prototypes of Shaper running on a wireless testbed.

In the first prototype, a wireless end-user device passively monitors the network when it experiences poor performance, indicated by low throughput, frequent 802.11 retransmissions or high latency. During the monitoring period, it captures any packet in the air and records the MAC addresses in the packet header. Then, the device piggybacks these MAC addresses in a "help" message. This message is sent to its access point which in turn forwards it to the backend router. Having received such a message, the router uses the global client-AP association list to determine a list of access points with which the reported devices are associated. Then it uses the throughput accounting tools such as the netfilter/iptables account extension to examine the throughput of these access points. If the aggregate throughput is at the network capacity and there exists uneven bandwidth distribution, then the shaping action will take place.

The second prototype implements the router-does-it-all model. We record the throughput of each access point at the router using the ip_account iptable module [5]. We have also programmed a kernel netfilter module and an iptable extension that implement two monitoring functions at the router. One is to count the number of TCP retransmissions for each access point. When forwarding a packet for a flow, the router uses the sequence number to infer whether a packet delivery or a reception has failed over wireless. The second is to track the round-trip latency between the router and an end-user wireless device, an indication of busy airspace. When these metrics together with throughput division between devices indicate an unfair network bandwidth allocation, then Shaper will start throttling the traffic.

A demonstration of both prototypes are available online.[2] Note that we assume static interference relationship in both prototype implementations. Accurate interference correlation is however a critical challenge in live wireless networks where

[2] Prototype 1: http://www.cs.ubc.ca/~kcai/client_ask_for_help.html
Prototype 2: http://www.cs.ubc.ca/~kcai/router_does_all.html

interference relationship is consistently changing. We will detail and address this challenge in Chapter 4.

3.3.7 The Trade-Offs

The trade-off that our approach makes is between network utilization, latency and fairness.

Overall Network Utilization Tradeoff

We have already seen from Figure 3.2 that Shaper can provide fairness only when the network utilization is under 20 Mbps in our testbed. Beyond that point, even though network utilization continues to increase, fairness can no longer be preserved. In fact, this throttling limit is largely decided by the channel conditions of the losing flow, i.e., how much extra bandwidth the losing flow needs to salvage from the dropped packets to sustain its "fair" bandwidth share. In our controlled testbed, this throttling limit is tuned between 16 Mbps and 20 Mbps.

However, it is worth noting that if packet loss of the losing flow is due to bad channel conditions (e.g., weak signal to the background noise) instead of airspace competition, then throttling other devices' throughput will not improve its performance. In this case, throttling will only reduce the overall network utility because it harms the performance of winning flows without helping the losing flows. The implementation of Shaper should be aware of this pitfall.

Note that when devices are running at different rates, there is a difference between *time fairness* [90] and throughput fairness. The time fairness allows the competing devices to equally share the channel access time, while the throughput fairness guarantees that they will achieve the same throughput. The difference between these two fairness schemes is due to the fact that a wireless device running at a lower rate will take much longer time to send a packet than one using a higher rate. Therefore, achieving a throughput fairness enables the slower devices to occupy much longer channel access time than the others, resulting in significant loss in overall network utility. Therefore, in our opinion, time fairness is the fairness

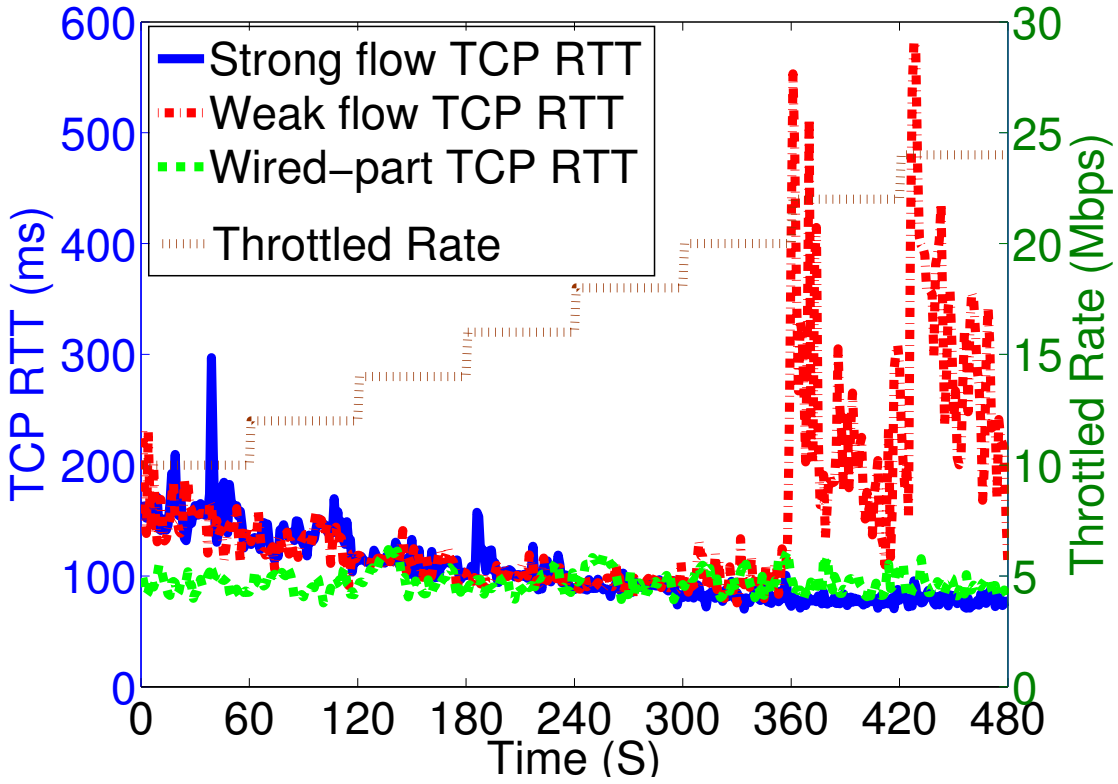


Figure 3.3: The Impact of Shaper on TCP RTT

that Shaper should try to achieve [89]. Implementing such a time fairness system, however, is out of the scope of this thesis.

TCP RTT Tradeoff

Throttling the throughput at the router can delay the time packets being injected into the wireless network and thus has an adverse impact on the network latency. In this section, we conduct an experiment to evaluate the impact of Shaper on the TCP Round Trip Time (RTT). This experiment consists of 40 TCP flows: 20 flows from the winning devices and 20 from the losing devices. All traffic is sent through the router and an extra 50ms delay is added to simulate cross-continent traffic.

In the first run, we remove the wireless links from the routing paths, and disable Shaper. The resulting RTT is used as a baseline metric — the best RTT that can be achieved when 40 flows send at full speed without shaping and SFQ. In the second run, we enable the SFQ Shaper and add the wireless links back to measure the new RTT values. The results of both tests are presented in Figure 3.3. Not shown there is that Shaper helps the flows to achieve fairness when the aggregated throughput is below 20 Mbps.

We can see from Figure 3.3 that Shaper does impose a RTT penalty: compared to the baseline RTTs, throttling the aggregated throughput down to 10 Mbps results in 60-ms (std. 22ms) and 63-ms (std. 25ms) increases in RTT for both flows. This is not surprising since slowing the rate that the router drains its queue will cause additional queue delay to the flows. It is worth noting that the weak flows suffer from high end-to-end latency when unfairness occurs as well. This substantial increase in RTT, much higher than Shaper's latency overhead, is caused by the frequent wireless retransmissions, when the highly-loaded network stops Shaper and 802.11 from working.

3.4 Challenge: Interference Detection

One critical challenge remains for Shaper to be effective in large 802.11 networks: it must determine which nodes are mutually interfering so that it can shape all of them as an unit to deliver fairness to all of them.

The problem of identifying mutually interfering devices has drawn a lot of attention from the research community [74, 77, 82]. However, most of the prior work has considered research testbeds - statically configured networks where nodes are immobile and traffic is constant. Even with these simplifying assumptions, inferring interference relationships between devices is not trivial. The common approach relies on an explicit measurement phase. Each device is asked to broadcast/unicast in turn [77, 82] (or a pair of nodes to send packet simultaneously [74]). Other nodes passively listen to the traffic to obtain RSSI/SNR values of all the other nodes in the network, used for later interference level analysis

and clustering. This approach requires not only careful synchronization between devices but also a shut-down period of regular network traffic.

For the deployment of Shaper, these techniques cannot be applied for a number of reasons. Shaper does not have control of the end-user devices to have them send packets. Moreover, network dynamics can change the interference relationship between devices rapidly. Finally, we believe it is unlikely that users or administrators will find it acceptable to periodically shut out the network activity in an existing wireless network.

Our interference detection system uses online user traffic for interference correlation. Doing so has three advantages. First, it is able to adapt to environment changes such as node mobility or traffic variations. For example, even if a pair of devices are within interference range, Shaper needs to throttle their traffic only when their total throughput demand exceeds the 802.11 network capacity. Second, it does not require control of the end-user devices and thus is easy to be deployed in real networks. Finally, this approach does not require an in-field offline measurements, which can last hours even days to complete.

We have come up with such a passive interference detection system called Wireless Online Interference Detector (VOID), to be described in Chapter 4.

3.5 Summary

The 802.11 protocol is not designed to support the cooperation between multiple autonomous systems. One of the unfortunate consequences of wireless interference is that the shared airspace resource can be unfairly distributed to competing devices.

In this chapter, we emphasize that congestion in 802.11 networks is the immediate cause of throughput unfairness. Based on this correlation, we argue that standard TCP and fair queuing together are sufficient to address unfairness, as long as we can prevent network congestion. A detailed analysis has been provided. Compared to alternative approaches, our approach is easier to deploy since no changes are required on access points or client devices.

We have used a testbed to validate that this cross-layer approach is indeed effective. We show that, even in the worst scenarios such as hidden-terminal and exposed-terminal topologies, throttling the aggregate throughput below the congestion level can allocate bandwidth evenly between all competing devices. We also evaluate the effectiveness of different queuing schemes on wireless fairness.

Chapter 4

Passive Interference Detection Using Online Traffic

As discussed in Chapter 3, knowing which devices interfere is the first step to minimizing interference, improving efficiency and delivering quality networking performance to all competing devices. This knowledge of interference relationship, however, is extremely difficult to obtain without either taking a running network offline for measurements or having client hosts monitor and report airspace anomalies, something typically outside the control of network administrators.

In this chapter we describe a novel technique we have developed to reveal wireless-network interference relationships by examining the network traffic at a wired router that connects wireless domains to the Internet. This approach, which we call Wireless Online Interference Detector (VOID), searches for correlated throughput changes where traffic from one node causes a throughput drop at other nodes in its radio range. By doing so, VOID is able to identify each node's interference neighbors using a single set of performance data collected from a wired-network router.

We can see from Figure 3.2 that the throughputs of the interfering devices are obviously correlated under congestion. That is, when network is saturated, an increase in one device's throughput indicates a throughput decrease for the other

competing devices. VOID explores this adverse correlation in devices' throughputs to identify the interfering devices under congestion.

The crux of VOID is a statistical correlation engine using Multiple Linear Regression (MLR). The input to this statistical engine is the throughput measurements collected from the central router and the output from the statistical engine is a list of interference sets, indicating which and to what extent certain devices are interfering.

In this chapter, we describe this passive online interference detection system in detail and show that VOID is able to accurately correlate interfering devices together and effectively discriminate interfering devices from non-interfering ones. The content in this chapter is mostly based on our prior VOID paper [37]. In Section 4.6, we present our experiment results using live traces collected from UBC wireless network.

We start this chapter with an introduction of the general motivation behind VOID in Section 4.1, and describe its methodology in Section 4.2. The limitations of VOID are discussed in Section 4.3. We then present our evaluation results in Emulab and UBC WiFi networks in Section 4.5 and Section 4.6 respectively. In Section 4.7, we present the interference relationships and patterns we found in the live networks. Finally, we summarize in Section 4.8.

4.1 Introduction

The prerequisite for many interference mitigation systems, including Shaper, to be effective is to be able to answer the following question: given a device that is experiencing significant performance degradation, which nearby devices are its current *interferers*? The state-of-the-art approach to estimate interference relationships is to take the network temporarily offline and to inject synthetic traffic for interference measurements [69, 73, 74, 77, 82]. Each node in turn sends unicast or broadcast packets into the airspace while other nodes record information such as signal strength and packet delivery ratios. These measurements are then used to seed various interference models [69, 77, 82] to predict network perfor-

mance at run-time. The PIE work [88] instructs the access points to record the time and successful transmission rate when they transmit packets, and uses this information to infer which transmitters are mutually interfering. It is an online approach using only the live traffic, however, it still requires driver modication and cooperation from both APs and mobile clients.

In this chapter, we present VOID (Wireless Online Interference Detector), an approach that utilizes statistical methods to recognize the interference patterns from *online* per-node throughput summaries taken from upstream *wired routers*. It requires no network profiling. Its methodology is based on the observation that, when the airspace is congested, changes in a victim node's throughput are more tightly related to its interferers than to other irrelevant devices. The pattern is that a victim node sends more when its interferers send less and vice versa.

VOID requires only information available at the wired network, needing neither cooperation from the end-user devices nor modifications to the access points. Moreover, since the core of VOID is a statistical interference correlation engine based on *multiple regression*, which can detect multiple interferers to a victim node in just one measurement round, VOID is capable of producing an interference map in real time. The complexity of its MLR engine is $O(K^2N)$ where N is the total devices considered and K is the number of throughput samples. We show in our evaluation that, in the worst scenario, VOID is capable of identifying the interfering devices from 100 candidates within 100 seconds. While in live networks, it usually only takes less than a second.

All these benefits are derived from the key difference between VOID and other approaches: VOID does not directly measure the impact of RF characteristics on interference in a given network, rather it statistically *infers* interference relationships by measuring throughput changes in the upper network layer.

However, note that VOID is not designed to produce the static physical-layer interference map as the offline approaches do. Rather, its goal is to find the culprits of throughput degradation for a victim device at a given time. Even when a static interference map indicates that it is possible for a device to interfere with a victim,

VOID does not necessarily flag it as long as it is not currently causing the drop in performance for the victim (when it is not sending data, for example). The interference map VOID produces is *dynamic*; it depends on the current network topology or traffic profile, and is almost always a subset of the static interference map.

We have evaluated VOID in two different settings: One in the Emulab wireless testbed [94], the other in the 802.11 network in two UBC buildings, one academic and one residential. In Emulab where the underlying interference relationship is pre-determined, we set up a variety of controlled multi-hop, multi-flow scenarios to evaluate the effectiveness and accuracy of VOID. In the UBC buildings, we evaluate how frequently interference occurs in live settings with real traffic. One problem with this real workload is that we do not know the ground truth against which our results should be compared. As we discuss in Section 4.6, we overcome this methodological challenge using a complement of techniques that estimate the ground truth in various ways.

Our evaluation results show that VOID is indeed effective, managing to correctly cluster interfering devices in all topologies we set up in Emulab. More importantly, VOID is able to detect hundreds of interfering pairs in each building during a week span, and more than 97% of the detected interfering devices operate on the same channel.

4.2 Methodology

VOID uses a statistical method called Multiple Linear Regression (MLR) to correlate interfering devices by recognizing the interference patterns (*throughput changes*) from the wired logs collected at the central routers. In this section, we describe its methodology in detail.

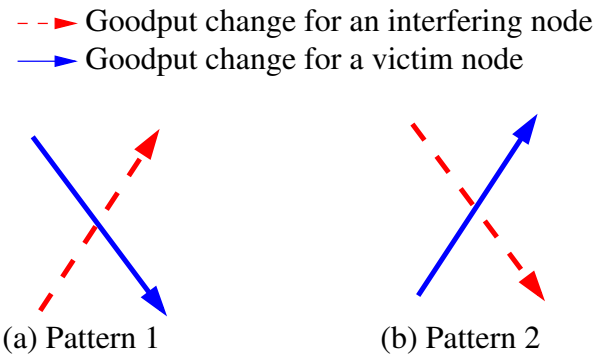


Figure 4.1: Interference Pattern between the Interfering Node and the Victim Node. On the Left, the Victim Node's Throughput Decreases as the Interfering Node's Throughput Increases. On the Right, the Victim Node's Throughput Increases as The Interfering Node's Throughput Decreases.

4.2.1 Interference Patterns Under Congestion

The causes of signal interference can be quite complex to analyze in real networks, since they depend on transmission power, signal propagation, node topology, environment layout or even time of day. But in Chapter 3, we demonstrate that the changes of underlying interference relationships can be reflected in changes to network throughput. Given a victim node and its interferers, these interference patterns are: as the interferers' throughputs increase, the victim's decreases and on the other hand, when the interferers' throughputs decrease, the victim's increases, as illustrated in Figure 4.1.

VOID exploits this correlation to infer interference relationship by searching for these patterns in the wired trace. Note, however, that these patterns only occur when the wireless network is congested since the throughput of the victim node will not drop as long as the overall network traffic does not exceed the network capacity. On the other hand, when the network is not completely congested, the MAC-layer and TCP packet recovery schemes are capable of salvaging dropped packets themselves.

In this section, we first give an overview of the statistical model VOID uses

and then discuss its limitations when being deployed in real settings.

4.2.2 Statistical Methods

In VOID, we assume interference relationships are linear, that is, the throughput that the victim node can achieve is approximately linearly related to the (negative) throughputs of its interferers. When adding a new interferer into the network, its throughput will cause an additional degradation in the victim node's throughput in the presence of other existing interferers.

Note that this linear interference model is not intended to exactly model the MAC-layer interference relationship — MAC-layer retransmission, exponential back-off and interference summarization could lead to a much more complicated model. Rather, this linear model is a simplified interference *estimation* over a series of IP-layer throughput-change samples.

One-Interferer Scenarios

Let us take one victim node and one interferer for analysis. If the victim node can sense the interferer's signal, then it has to back off while the interferer is transmitting. On the other hand, if the interferer is a hidden terminal to the victim node, then the victim's packets will be corrupted by the interferer's signal. In either case, the MAC-layer interference can be modeled by the correlated throughput changes at the upper IP layer. The linear regression engine is thus used to search for these throughput-change patterns in the trace and use them to measure, on average in the given time window, how much damage each of the interferers has caused to the victim node's throughput. If due to any reason that two devices' throughput changes do not seem to be coordinated, then they will not identified as interfering even if they do.

Therefore, in these simple one-interferer scenarios, Correlation Coefficient (CC) can be used to calculate the degree of correlation between two devices' throughputs. The CC value (ρ) can take on any values in the range [-1, +1], where a value close to +1 imples strong positive linear correlation, a value close

$\rho_{1,2}$	$\rho_{1,3}$	$\rho_{1,4}$	$\rho_{1,5}$	$\rho_{1,6}$	$\rho_{1,7}$
0.019	0.035	-0.004	-0.013	-0.37	-0.29

Table 4.1: An Example Showing that Correlation Coefficient is Ineffective in Detecting Interferers for a Victim Node in Many-Interferer Scenarios

to 0 implies no linear correlation, and a value close -1 implies strong negative linear correlation, i.e., interference. Our evaluation in typical two-flow hidden- and exposed-terminal scenarios has shown that a strong interferer can result in a correlation-coefficient value below -0.9.

Interference relationships in real networks can be, however, quite complex. A victim node will be under the influence of many interferers, and interferers themselves can be affected by other nearby interferers. The impact of a specific interferer on the victim node might be hidden from the pair-wise observations used in calculating the CC value.

To demonstrate this problem, we conducted an seven-flow experiment in which all flows mutually interfere with each other. These flows were all streaming with UDP traffic, ensuring that the airspace was saturated. Table 4.1 presents the correlation coefficient values between flow 1 and the other six flows. The interference relationship between flows is not obvious between any pair of devices.

The correlation coefficient is unable to detect interference since it only checks two devices at a time. In a typical environment where the throughput of one device is related to many others, we must resort to a more expensive statistical method, Multiple Linear Regression (MLR), to take into account the simultaneous effect of several interferers.

Many-Interferer Scenarios

To identify the effect of k potential interferers $(I_1, I_2, \ldots, I_k)$ on a victim node (V), we assume a linear relationship between their throughput values. If we let the throughputs of the interferers be $(x_1, x_2, \ldots, x_k)$ and that of the victim node be y,

this linear relationship is expressed in Equation 4.1. We sample these variables n times and feed the values into Equation 4.1 to estimate the coefficients β. Each sample may also introduce independent error ε, a variable assumed with mean zero and constant variance.

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \ldots + \beta_k x_{ki} + \varepsilon_i \ (i = 1, \ldots, n) \tag{4.1}$$

β_0 indicates the throughput the victim node would achieve if all the other interferers are idle, while β_i suggests the *additional* impact of potential interferer I_i on the victim node's throughput when other interferers' throughputs are held constant.

If I_i is currently interfering with a victim node (i.e., I_i is a *true* interferer), then β_i should be a negative number. If β_i is close to 0 or a positive number, then I_i may be a false interferer. A positive number indicates that I_i may in fact be helping the victim node by reducing the throughputs of other interferers.

Solving Multiple Regression. Solving multiple regression is to estimate the coefficient values to best-fit the sampled data. It is convenient to express Equation 4.1 in matrix notation as shown below.

$$y = X\beta + \varepsilon \tag{4.2}$$

where,

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, X = \begin{bmatrix} 1 & x_{11} & \ldots & x_{k1} \\ 1 & x_{12} & \ldots & x_{k2} \\ \vdots & \vdots & & \vdots \\ 1 & x_{1n} & \ldots & x_{kn} \end{bmatrix}, \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \varepsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}.$$

The least squares estimator $\hat{\beta}$ for the regression coefficients in β is

$$\hat{\beta} = (X^T X)^{-1} X^T y \tag{4.3}$$

where $\hat{\beta}$ is calculated to minimize the *residual sum of squares*, SS_{res}.

$$SS_{res} = \sum_{i=1}^{n}(y_i - X\hat{\beta})^2 \tag{4.4}$$

Determining The Best Fit. The coefficient of determination, R^2, measures how well a regress model fits the sampled data. It is a value between 0 and 1; the higher value R^2 is, the better the data fits our linear model. In VOID, for example, when R^2 equals 0.9, this suggests 90% of the victim node's throughput change (y) can be explained by changes in the potential interferers' throughputs (X).

$$R^2 = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2 - SS_{res}}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{4.5}$$

where $\bar{y} = \frac{\sum_{i=1}^{n} y_i}{n}$.

Detecting Interferers. The interferers included in the initial multiple regression model are only the candidates; the resulting regression model might contain false interferers that contribute little or nothing to the victim's throughput variation.

We use *forward selection* to select the most significant interferer to add to the regression model one at a time, until it no longer improves the model coefficient of determination R^2 by adding any more interferer (in other words, the residual sum of squares cannot be reduced any more).

4.3 Limitations of VOID

In contrast to the other RF/MAC models introduced in prior work [69, 77, 82], our interference model is much simpler. This simplicity comes from the fact that our model aims to detect the interference between a potential interferer and a victim node, while the prior models are designed to accurately predict the throughput of every device under interference. Because of this difference in goals, the other

models must capture the details of the MAC layer and radio layer characteristics. VOID, on the other hand, examines only wired-network throughput changes.

As a result, VOID can only detect interference between nodes when (a) their traffic has congested the network, (b) the interfering node is currently able to have a significant impact on the victim node and (c) the victim node is able to promptly respond to the changes in the interferer's throughput. Each of these three limitations is described below.

No Congestion in 802.11 networks. Throughput-change interference patterns only occur when the wireless network is congested. Previous work [34] has shown that the average throughput of the victim node will not drop as long as the demand from both devices does not exceed the network capacity. This is because, when the airspace is not completely congested, the MAC-layer and TCP packet recovery schemes are capable of salvaging dropped packets themselves. The victim node can use the airspace when interfering nodes are idle.

Therefore, VOID is unable to correlate interference in non-congested scenarios. However, it is less interesting to correlate interference in those scenarios since interferers have no real impact on the performance of the victim node.

Insignificant Interferer Under Congestion. Even in a congested environment, the relationship between an interferer and the victim node can still be obscured if its current impact is not significant compared to that of others in a given time window. For example, consider a victim node A that could be affected by two interferers B and C. If B is transmitting at full speed while C does not have much traffic to send right now, the current throughput degradation of A will be caused (mostly) by B. While C would interfere more if it were sending more data, it is currently not a significant source of interference for A and will not be identified by VOID.

In this way, VOID can only detect the significant interferers to a victim given the current traffic profile, not a complete interference map. However, given another window, when the throughput of C increases, VOID will identify C since its

impact on *A* has become detectable in the trace.

Delay in Response. VOID relies on the prompt response of the victim node to changes in the interferer's throughput. In most cases, this will be the case – a victim's throughput will quickly increase with less interference, and decrease with more, however, in some scenarios, it may take a much longer time for a victim node to make use of free network capacity; the longest MAC-layer back-off time for a 802.11b sender is about 21 ms (1024 * 20 + 50 μs). To take this time into consideration, the sampling period should be long enough to detect the response from the victim nodes.

Note that a TCP sender could have been put into TCP starvation due to continuous MAC-layer packet losses. In this scenario, the TCP sender is not attempting to send any packet for tens of seconds or minutes. Since VOID cannot observe traffic for the starved devices at the router, it is not able to correlate the set of interferers. We believe this is not a problem with VOID, but rather a problem with TCP [71], however, solving this problem is out of VOID's scope.

UDP *Traffic.* VOID presumes the IP-layer throughput sampled at the router is an indicator of MAC-layer goodput over the wireless links. This assumption is obviously not always true for UDP traffic.

4.4 Implementation

VOID is implemented with a pipeline consisting of three subsystems: trace collection, throughput analysis and interference correlation. In a small network, trace collection can be simply done using network tools such as tcpdump [9]. In a large network, on the other hand, trace collection overhead can be unloaded from the router using passive port mirroring[1] to transfer traffic along a parallel data plane to a dedicated trace collection machine.

[1] http://en.wikipedia.org/wiki/Port_mirroring

The throughput analysis subsystem needs to produce a time series of throughput stats for all devices observed in the trace. We use WyPy [67], an online traffic analysis tool, to break the captured trace into small time buckets and then aggregate the throughput per device in each bucket. WyPy is able to quickly output the throughput series — it takes less than half a second to analyze a one-second trace collected in all our testbed scenarios and live UBC wireless network. The throughput time series is then fed into the interference correlation subsystem, which uses SciPy [8] to implement the multiple linear regression engine to detect the interferers from all candidate devices.

4.5 Evaluation in Emulab Testbed

In this section, we evaluate VOID's effectiveness using the Emulab wireless testbed at Utah University [94]. We control the interference map in this environment to evaluate if the multiple regression engine of VOID is effective in determining the set of interferers to a victim node, and if it is able to adapt to network dynamics.

4.5.1 Emulab Experiment Settings

The testbed consists of a total of 72 wireless nodes spreading over two floors in the Merrill Engineering Building. We conduct most of our experiments on two clusters, the 36-node pc-600 cluster on the third floor and the 13-node pc-3000w cluster on the fourth floor as highlighted in Figure 4.2. The 36 nodes in the pc-600 cluster are deployed in one 300cm x 224cm grid. All nodes in the grid are able to communicate with each other; in other words, they all interfere. On the other hand, the 13 pc-3000w nodes are deployed in a 360 m x 100 m floor; any node takes at most five hops to reach any other node in the cluster. For reference, we call the pc-600 cluster the "single-hop cluster" and the pc-300w cluster the "multi-hop cluster".

In each experiment, the wireless nodes are grouped into AP-client pairs. These pairs are selected to build various interference topologies. Each wireless node is

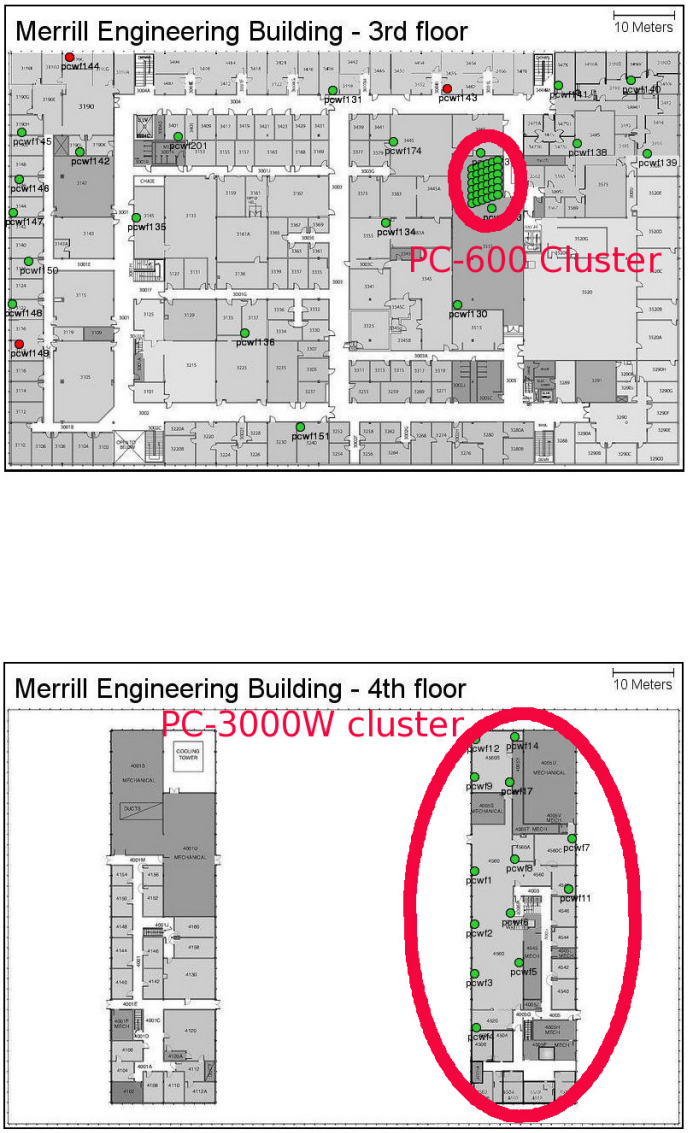


Figure 4.2: The Emulab Wireless Testbed at Utah

equipped with at least one Atheros card with AR5212 chipset, running the MadWifi driver [6]. In all scenarios, the transmission power is set to minimum, the transmitting rate is fixed at 36 Mbps and the autorate function is disabled so that a topology does not change during an experiment.

We use the D-ITG [1] traffic generator to set up 5-minute TCP flows between a server and the wireless clients. Unless stated otherwise, each flow generates an average traffic demand of 22 Mbps. This high throughput demand is set to ensure that 802.11 network congestion does occur. All traffic is configured to be routed through the sole router in the testbed. We use `ipt_account` netfilter kernel module [5] to log the throughput information of all the flows at the router every second. VOID's MLR engine runs over a 5-minute-window trace, providing 300 points per device per time window.

4.5.2 Testbed Experiments

The testbed experiments are conducted to answer the following three key questions: (1) Is VOID able to extract the pair-wise interference relationships in complicated many-to-many interference scenarios? (2) Is VOID able to discriminate true interferers from the false ones? (3) Is VOID able to adapt to network dynamics?

Typical Two-Competing-Flow Topologies

In the experiments described in this section, we consider only two competing flows. In the hidden terminal scenario, four nodes are chosen from the pc-3000w cluster so that the victim flow can only achieve 1 Mbps throughput while the interfering flow achieve a throughput over 20 Mbps. In the exposed terminal scenario, the two wireless flows are chosen from the pc-600 cluster and they split the bandwidth 50-50.

Hidden Terminals. The results from the hidden-terminal experiment are shown in Table 4.2. The coefficient value β_0 (-0.88) indicates that the interference level

β	R^2
$\begin{bmatrix} -0.88 \\ 2.1e+07 \end{bmatrix}$	0.90

Table 4.2: Results of Hidden-Terminal Topology

β	R^2
$\begin{bmatrix} -1.03 \\ 2.55e+07 \end{bmatrix}$	0.85

Table 4.3: Results of Exposed-Terminal Topology

between these two flows are strong. The β_1 with a value of 2.1e+07 indicates that the victim node could have achieved a good throughput of 21 Mbps if it was the only flow in the network. The R^2 indicates that the interferer contributes 90% in the victim node's throughput degradation.

Exposed Terminals. As expected, the results shown in Table 4.3 from the exposed-terminal experiment tell us the same story: the victim flow is strongly interfered by the other interferer in the experiment, even the victim flow in this setting achieves a much better (10 times more) throughput than that in the hidden-terminal case. Note that, even though we fix the rate to 36Mbps, the flows in the pc600 cluster can achieve a throughput of 25 Mbps, in contrast to the flows in the pc3000w cluster can only achieve a throughput of 21 Mbps. This difference is reflected by the parameter β_1 in both settings.

Many-Interferer Scenarios

We conduct this experiment in the single-hop cluster in which seven wireless flows (14 wireless devices) mutually interfere. We can see from Figure 4.3 that the interference relationship among these seven flows is quite complicated — checking any pair of flows reveals only a weak interference relationship between them. For

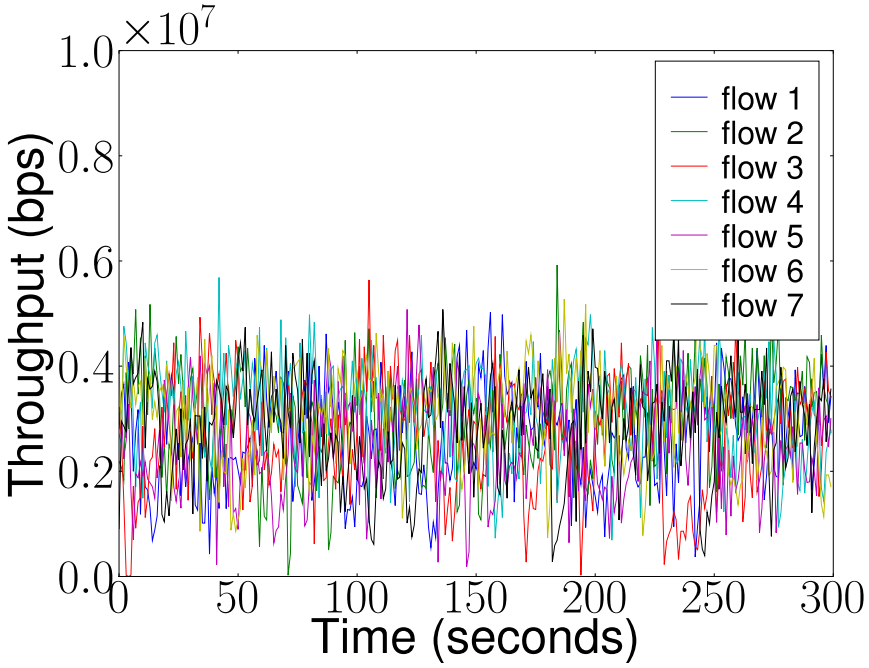


Figure 4.3: Throughput of 7 Mutually Competing Flows.

$\beta_{1,2}$	$\beta_{1,3}$	$\beta_{1,4}$	$\beta_{1,5}$	$\beta_{1,6}$	$\beta_{1,7}$	R^2
-0.78	-0.82	-0.75	-0.86	-0.88	-0.85	0.96

Table 4.4: MLR Coefficients β for Flow 1

example, if we check flow 1 and flow 2 only, the correlation between them is only 0.019 and R^2 is only at 0.08. However, the interference model's significance (R^2) increases when more and more interfering flows are added for consideration. In the final model in which all the seven flows are taken into account together, the β values in Table 4.4 suggest that all the other six flows are interferers to flow 1. The coefficient $\beta_{1,2}$ is now -0.78, indicating a strong interference relationship between flow 1 and 2. The R^2 of the final regression reaches to 0.96, means 96% of the variation in flow 1's throughput can be explained by the throughput changes of the other six competing flows.

Round #	Potential Interferers	β	R^2	Interferer Selected
1	F_2	-0.88	0.97	F_2
	F_3	0.002	$5.7 * e^{-4}$	
	F_4	-0.010	$4.0 * e^{-5}$	
2	F_2, F_3	$\begin{bmatrix} -0.85 \\ -0.02 \end{bmatrix}$	0.89	None
	F_2, F_4	$\begin{bmatrix} -0.87 \\ -0.12 \end{bmatrix}$	0.9	

Table 4.5: A Step-by-Step Illustration of the Forward Selection Process.

Finding The Interferers

In this section, we evaluate whether VOID's repeated forward selection process is effective in filtering the irrelevant active devices from interference clustering.

We select three groups of wireless flows. The first group includes two flows, F_1 and F_2, from the single-hop cluster; the other two groups contain only one flow each, F_3 (`pcwf12` $\rightarrow$ `pcwf14`) and F_4 (`pcwf3` $\rightarrow$ `pcwf5`), selected from the multi-hop cluster. There is no interference between groups. The two flows in the multi-hop cluster do not interfere with the two flows in the single-hop cluster.

We keep all these four flows running at full speed. The two irrelevant flows are able to transmit at 20M bps each while the two interfering flows can only send at 10M bps. The step-by-step forward selection procedure for $F1$ is illustrated in Table 4.5. The first round of VOID runs regression on each of the possible interfering flows against flow F_1. The results of the three pair regressions show that F_2 is the most significant interferer among the three. We can see from Table 4.5 that R^2 of model F_1 and F_2 reaches 0.97, while R^2 of the other two models are at least three degrees of magnitude lower. Therefore, F_2 will be selected to be included in the final interference model.

In the second round of the forward selection process as described in Section 4.2.2, the algorithm considers adding more devices into the interference model. There are two possible three-candidate models by including either F_3 or F_4 into

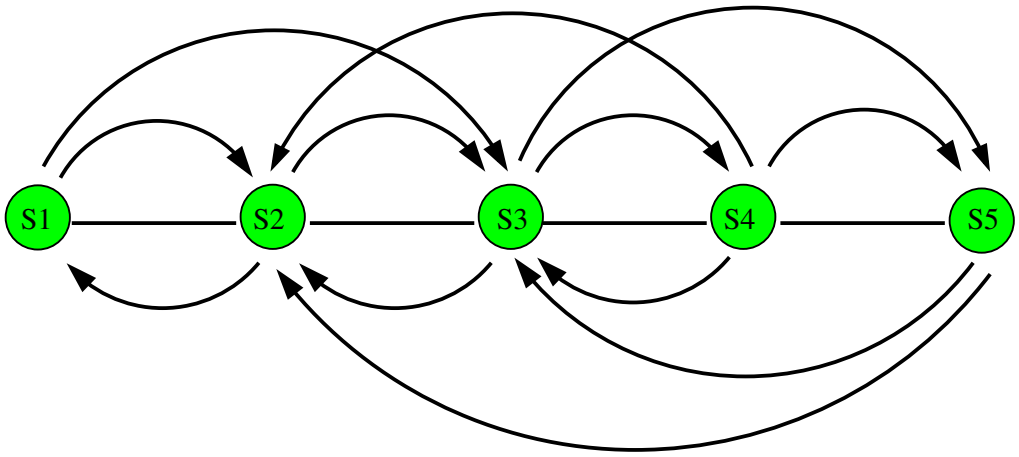


Figure 4.4: Physical Pair-Wise Interference Map

the two-interferer model. But as illustrated in Table 4.5, adding more interferer into the regression actually results in less accurate models. The R^2 of the model including F_3 drops by 0.08 while that of the model including F_4 drops by 0.07. VOID therefore will not include any additional device into the final interference model, and conclude that only F_2 is the only interfering flow to F_1.

Dynamic Interference Maps

In this section we demonstrate the VOID's ability to adapt to network changes such as traffic variations. We select five flows (`pcwf12` → `pcwf14`, `pcwf9` → `pcwf17`, `pcwf7` → `pcwf8`, `pcwf7` → `pcwf11`, `pcwf3` → `pcwf5`) from the multi-hop cluster and conduct pair-wise experiments to measure the pair-wise interference relationship. The full (static) interference map is shown in Figure 4.4 as a reference. Each arrow in the figure indicates the existence of interference and its direction. Note that the two flows in the middle F_2 and F_3 are affected by all the other flows.

We start from a completely congested network and then gradually reduce the throughput demands of the flows. Figure 4.5 illustrates three interference maps that are completely different, due to the changing traffic profiles. These maps,

however, are all subsets of the underlying physical pair-wise interference map.

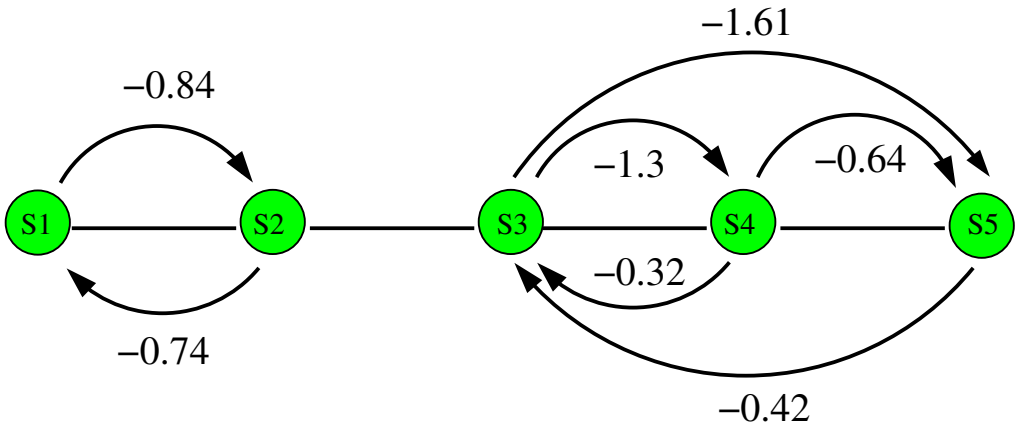
In a completely congested environment, the bandwidth allocation is 20 Mbps, 0.5 Mbps, 0.5 Mbps, 20 Mbps and 12 Mbps, for these five flows respectively. Given this traffic profile, the interference between F_2 and F_3 is not significant and thus cannot be detected by VOID. We can see from Figure 4.5a that these flows are grouped into two clusters in the final interference map: F_1 is the main culprit for F_2's throughput degradation while F_4 and F_5 dominate F_3.

We then reduce the average throughput demand of each flow to 8 Mbps for each flow. The three dominating flows F_1, F_4 and F_5 do *not* experience any bottlenecks in this scenario; the bandwidth available to them is greater than what they need; these flows do not appear to be affected by any other flows. However, they do impose interference of 24 Mbps in total on F_2 and F_3, the two flows that cannot meet their throughput demands.

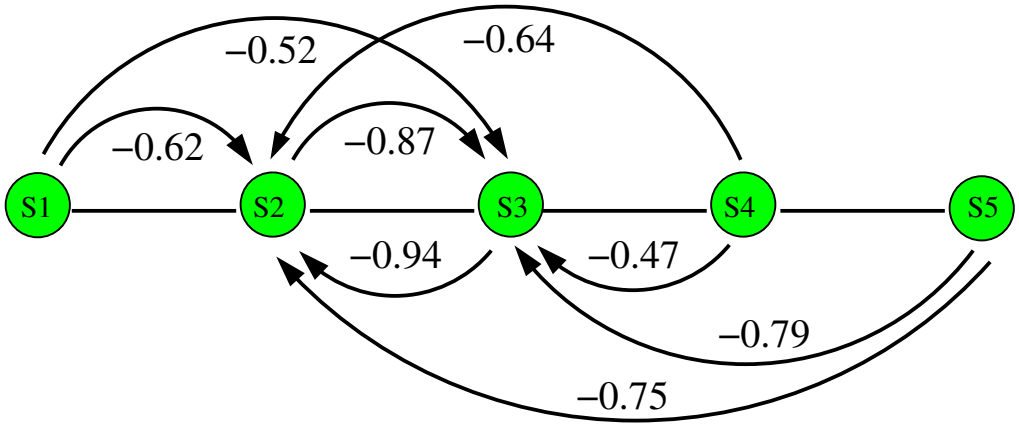
Finally, we reduce the individual throughput demand to 4 Mbps. Now whether F_2 can achieve its throughput demand depends largely on F_3 and vice versa. Therefore, only the throughput of this pair of nodes has the interference patterns that VOID looks for.

4.6 Evaluation in Live UBC WiFi Networks

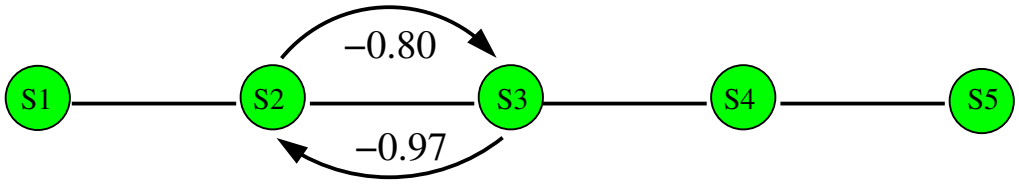
We are able to evaluate VOID's effectiveness using UBC campus wireless networks and, in the section, we present the evaluation results with live traffic. UBC wireless network consists of more than 1700 APS running with Cisco equipment and software. We chose two representative buildings for our analysis. One is UBC Highrise — a residential building was built in the 's that does not provide wired Internet access. It is, however, covered by 27 APs, all 14 floors but one are equipped with two APs. Another building is a newly-built research center, Life Sciences Center (LSC). It is the largest academic building in UBC, consisting of five floors and covered by 40 APs. These wireless APs are already carefully placed and operate on overlapped channels on both 2.4G and 5G band to minimize the interference in the network. Moreover, the Cisco Wireless Control System (WCS) deploys the



(a) Interference Map in a Congested Environment. The β Values below -1 Are Due To MAC-Layer Retransmissions.



(b) Interference Map in a Medium-Congested Environment



(c) Interference Map in a Low-Congested Environment

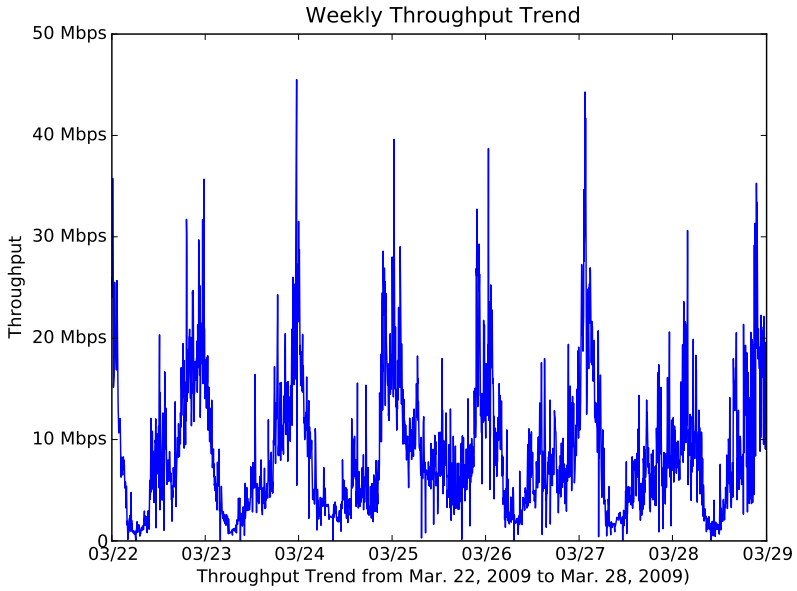
Figure 4.5: Dynamic Interference Map

CleanAir technology that can make automatic channel adjustments to optimize wireless coverage around the interference, including those from non-WiFi devices such as microwaves or wireless phones.

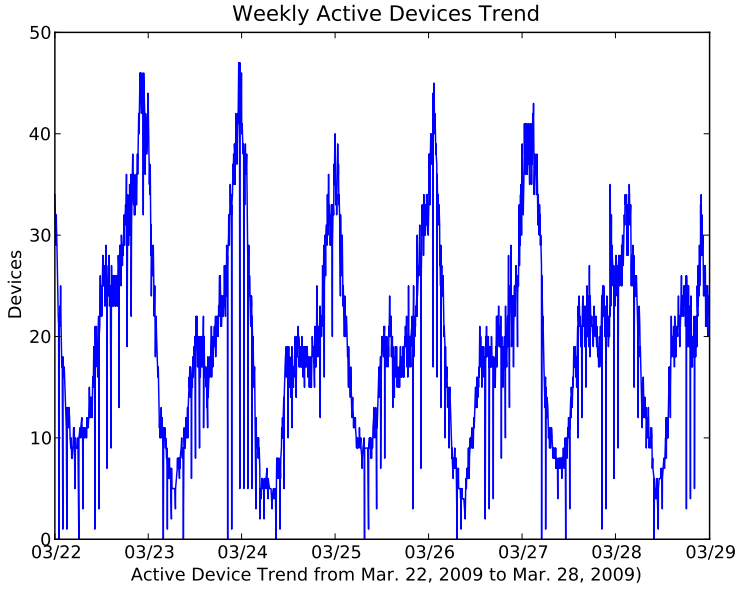
We captured one-week wireless traffic from each of the two buildings at the central controllers. The traces were collected in Highrise from Mar. 22, 2009 to Mar. 28, 2009, and those in LSC were collected from Apr. 13, 2009 to Apr. 19, 2009. These traces were later anonymized. Note that the traffic captured is only what the controllers see on the wire (i.e., traffic between the controllers and the wireless APS), it does not capture any last-hop wireless traffic on the air such as 802.11 ACK, beacons, etc. The internal wireless traffic such as a file transfer between two clients within the same building are also not captured. In these two week monitoring period, we were able to capture 1.06 billion packets in Highrise and 0.65 billion packets in LSC.

In addition to capturing the traces from the central controllers, we also pulled a stream of the SNMP data from server, including wireless clients' MAC addresses, channels and their associated APS. This information is not used by VOID, but we use this information for our false-positive and false-negative analysis and interference map correlation. For instance, we can determine if two devices found to be interfering are operating on the same channel. The SNMP polling frequency is throttled at once every 10 minutes as negotiated with the UBC wireless network management group to not cause any performance issue during peak hours.

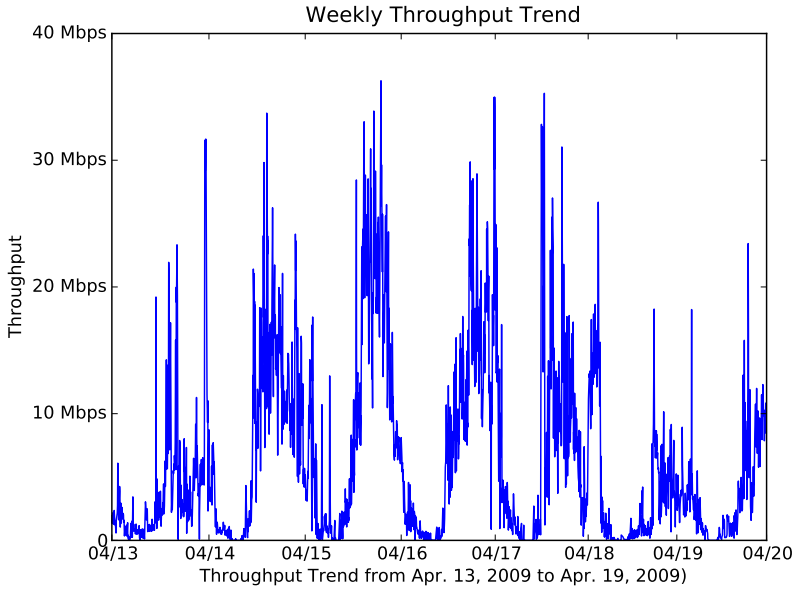
Figure 4.6 illustrates the weekly throughput and active-device trend in Highrise and LSC. The trends in both buildings clearly follow a diurnal cycle: LSC has a higher network usage during the day while Highrise sees its traffic peak at night. It is easily understood as LSC is a research building where there are more activities during the day time while Highrise is a residential building where people leave home for work for the day and come back home at night. Based on this information, we conduct most of our experiments in Highrise during the 6-hour peak hours from 6:00 PM to 11:59 PM and from 10:00 AM to 5:59 PM in LSC. Note that in both buildings, the aggregated throughput peak is usually below



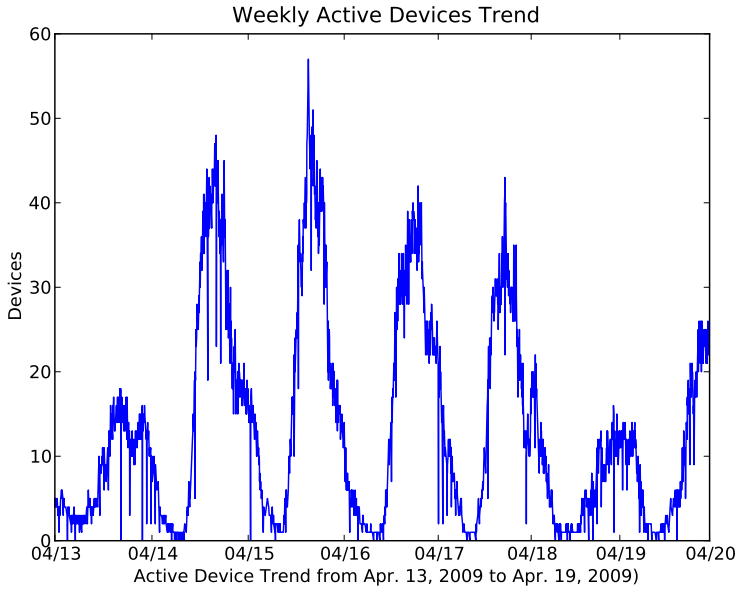
(a) Highrise Throughput



(b) Highrise Active Devices



(c) LSC Throughput



(d) LSC Active Devices

Figure 4.6: Weekly Trend

40Mbps and the number of active devices are below 50, the coverage of 27 and 40 APs should be able to provide enough network capacity for this traffic demand in both buildings. Therefore, we expect to see only localized interference between a few pairs of devices.

The challenge to evaluate VOID in this live network is that, unlike the controlled evaluation we conducted in the testbed, we do not know the underlying interference topology in these networks. Therefore, it is difficult to answer the following two questions:

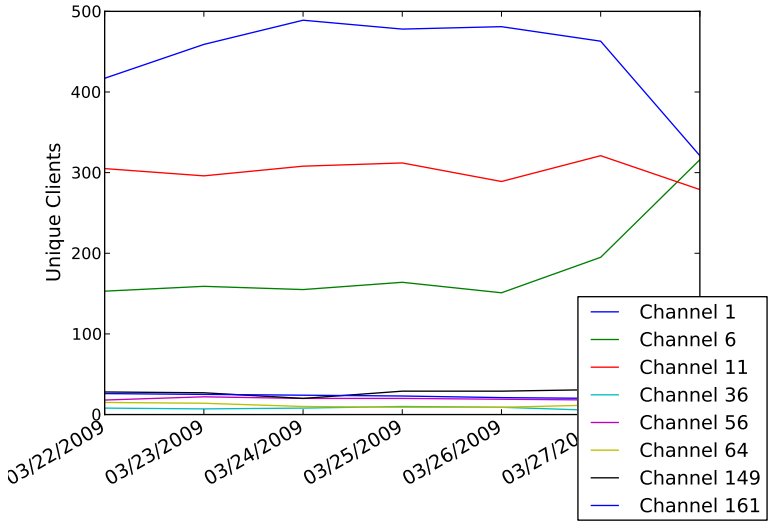
- *False positives*: How many devices are incorrectly labeled as interfering?
- *False negatives*: How many interfering devices are not detected?

4.6.1 False Positives

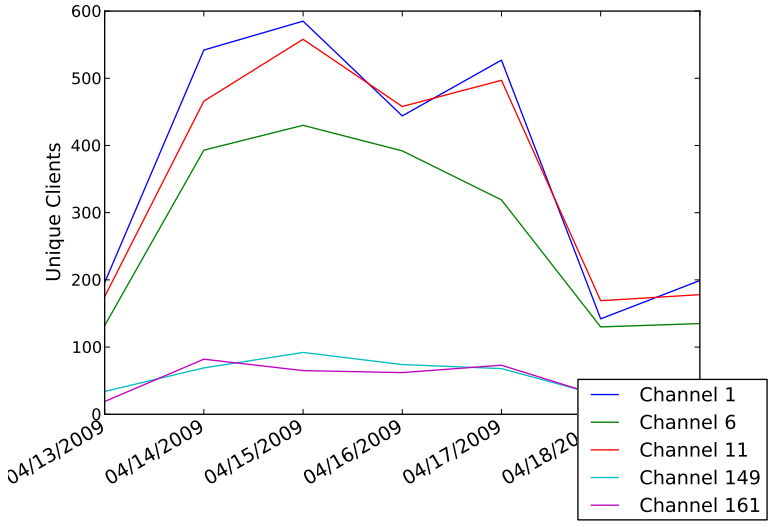
We use channel information to infer which pairs of de'vices detected by VOID are accurate. The one-week wireless client channel distribution is illustrated in Figure 4.7, from which we can see that the majority of wireless clients in both buildings operated on channels 1, 6 and 11, and only a small subset were operating on the 5G frequency band. The channels are more evenly distributed in LSC than in Highrise, nevertheless there are hundreds of clients on each of the three 2.4G channels.

Assuming devices operating on orthogonal channels do not interfere, if a pair of devices are found as interfering but are operating on orthogonal channels, then this detection is counted as a mistake against VOID. Otherwise, if they operate on the same channel, then we treat this pair as true interfering candidate.

We use False Positive Ratio (FPR) to evaluate VOID's accuracy, as defined by the ratio of orthogonal-channel interference detections to the total detections. The more accurate VOID is, the more interference found by VOID should be on the same channel and thus the false positive ratio should be lower. Note that some of these same-channel interferences detected by VOID could also be false positives and thus the FPR can be higher in reality than what we estimated. Although it is



(a) Highrise



(b) LSC

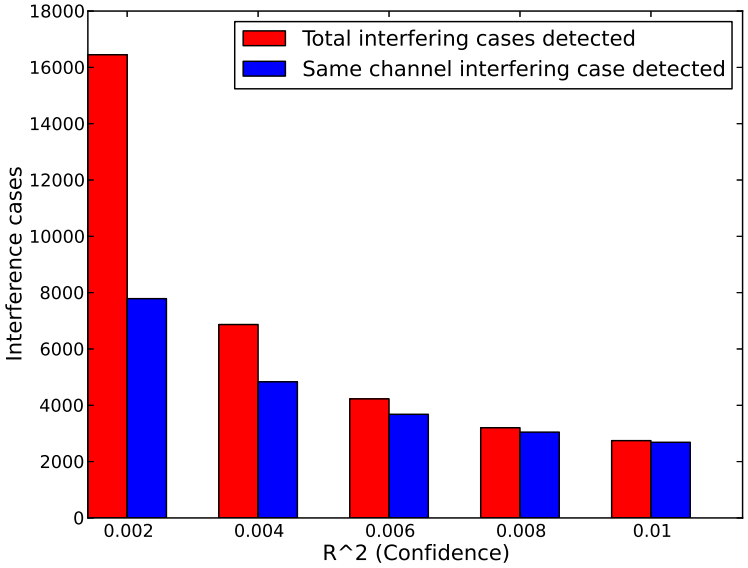
Figure 4.7: Channel Distribution in Highrise and LSC

impossible for us to get the real FPR number due to lack of underlying accurate interfering map, we confirmed in Section 4.7.2 that more than 99% of the detected same-channel interferences occur between devices within close proximity.

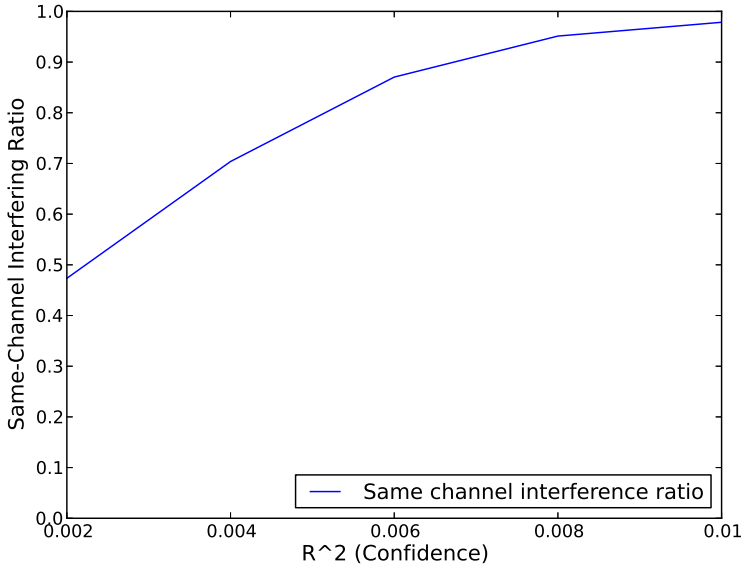
We use two heuristics, R^2 and VTBR, to differentiate the true interferers from the false ones. First, as in the experiments conducted in Emulab, we require at least a minimum value of R^2 to ensure the candidate device is a significant interferer to the victim device. However, R^2 value alone is not sufficient due to the burstiness of traffic characteristics in live networks. As we later show in Figure 4.9, with our default R^2 value of 0.01, more than 66% of the interfering devices being detected were still on orthogonal channels. Therefore, we have to deploy the second heuristic, called Valid Time Bucket Ratio (VTBR). VOID divides the two-minute time window into eight time buckets, and runs regression on all these time buckets. Only if the devices show consistent interference in these time buckets, VOID will treat them as true interferers.

The impacts of these two heuristics on FPR are now demonstrated in Figure 4.8 and Figure 4.9 respectively. When the R^2 and VTBR values are low, R^2 at 0.002 or VTBR at 0.1, for example, the same-channel interference ratio is below 50%. These values indicate that more than 50% of the detected interferences are wrong, as the correlated devices operate on the orthogonal channels. However, the same-channel interference ratio grows as we increase R^2 and VTBR values. In the first experiment, we keep VTBR at 0.5 and increase R^2 from 0.002 to 0.01, resulting the same-channel interference ratio in Highrise to increase from 47% to 98% and that in LSC from 60% to 95%. In the second experiment, we vary the VTBR from 0.1 to 0.5 while keeping R^2 at 0.01 and the same-channel interference ratio increases by 57% in Highrise and 47% in LSC, as shown in Figure 4.9.

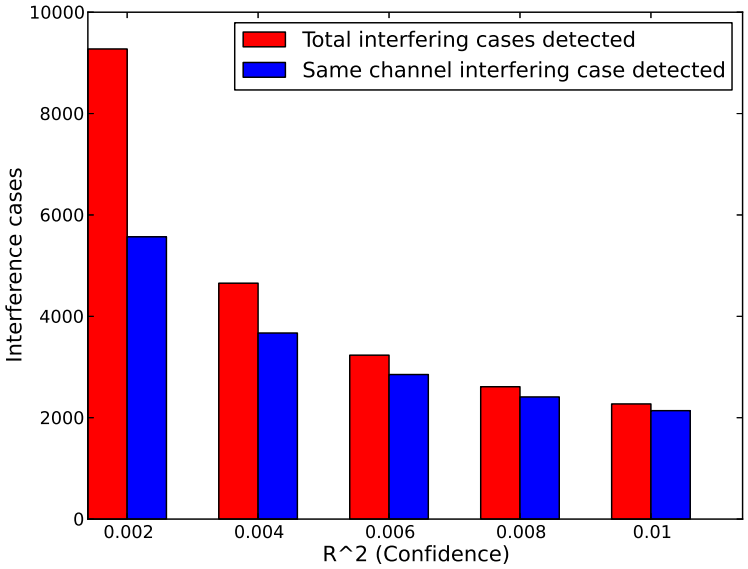
We keep R^2 at 0.01 and VTBR at 0.5 for interference correlation to ensure VOID accuracy. With this configuration, VOID is able to keep the same channel interference ratio under 5% in both buildings, indicating a small number of false positives. On the other hand, this brings up the next question: how often does VOID miss the opportunity to detect an interference by using this R^2 and VTBR



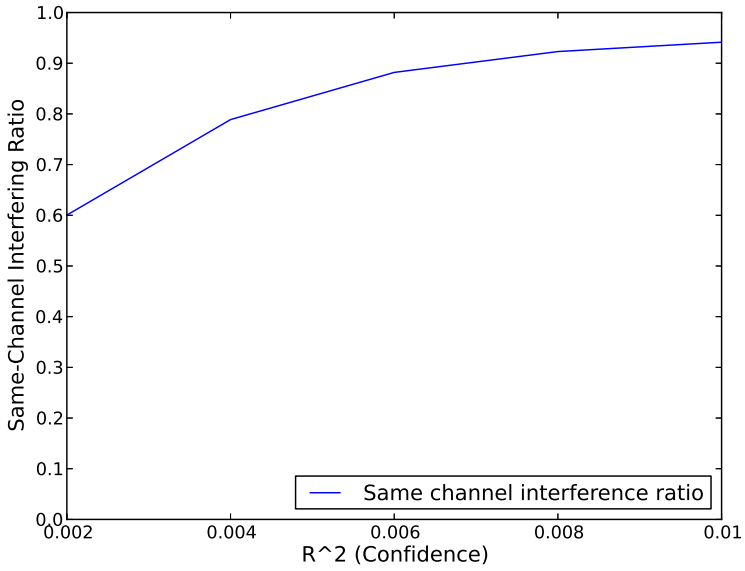
(a) Interference Cases in Highrise



(b) Same-Channel Interference Ratio in Highrise



(c) Interference Cases in LSC



(d) Same-Channel Interference Ratio in LSC

Figure 4.8: Impact of R^2 on VOID Accuracy, VTBR Is Kept at 0.5.

	Reason	Percentage
Testbed Pair in Highrise	Inactive	1.5%
	Orthogonal Channels	0.0%
	Topology Change	0.0%
	Other Interferers	17.5%
	Bursty Traffic	1.0%
	No Congestion	2.9%
	Interference Detected	73.7%
	Unknown (**False Negatives**)	3.4%

Table 4.6: False-Negative Analysis on a Testbed in Highrise

setting?

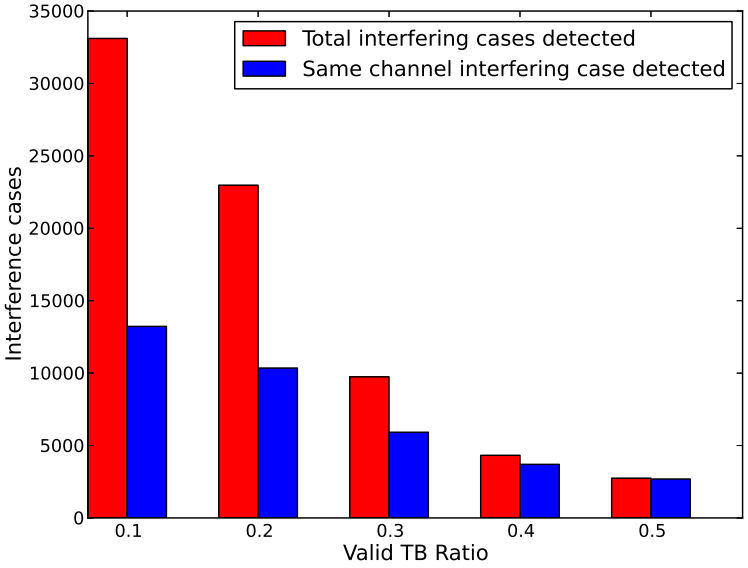
4.6.2 False Negatives

To determine VOID's false negative rate accurately requires knowing the ground truth about when devices actually interfere with each other. Unfortunately this sort of information is difficult to collect from live networks, which is of course a key motivation for our work. As a result, we performed the following two experiments to approximate a false-negative evaluation for this data.

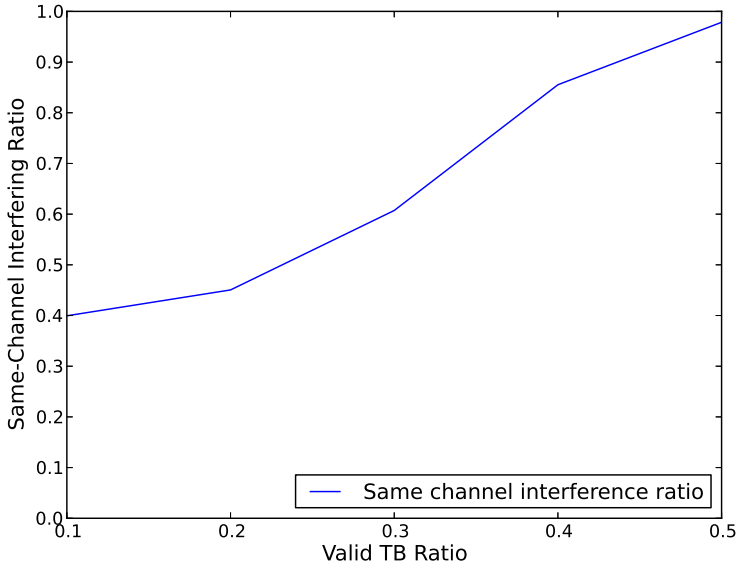
Testbed Evaluation

We first created a controlled, testbed experiment on the twelfth floor of the Highrise building. We placed two devices within interference range of each other, operating on the same channel, and we streamed TCP traffic to both. The bandwidth demand of these two streams was sufficient to congest the wireless network the two devices shared. The stream traffic to these two devices lasted for 6 hours from 18:00 to 23:59 on Apr. 2nd 2009, forcing them to compete for airspace with each other and with other nearby wireless clients.

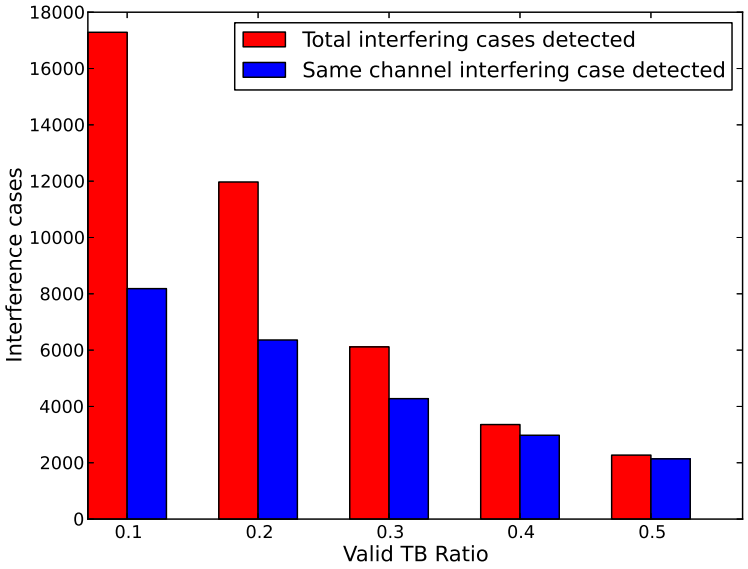
We analyzed all 720 time buckets during these six hours and classify them into the eight categories listed in Table 4.6. We can see that VOID was able to detect



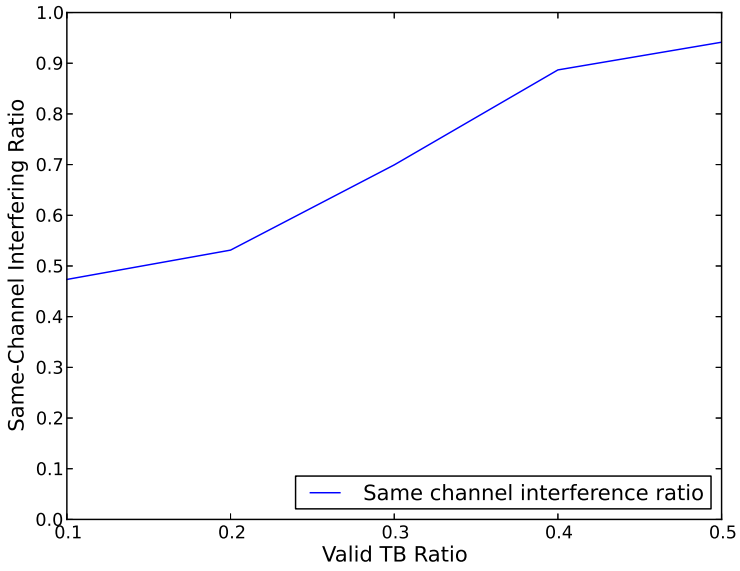
(a) Interference Cases in Highrise



(b) Same-Channel Interference Ratio in Highrise



(c) Interference Cases in LSC



(d) Same-Channel Interference Ratio in LSC

Figure 4.9: Impact of VTBR on VOID Accuracy, R^2 Is Kept at 0.01.

interference in the testbed 73.7% of the time. 17.5% of the interference is masked by the live traffic sent by other Highrise residents' WiFi devices. We found five such external wireless devices, associated with three different APS operating on the same channel on floor 9 and floor 12, and during these time buckets, the interfering testbed pair were only able to send half of their streaming traffic. There are also times when the pair are not sending any/enough traffic due to either streaming software crashes or TCP congestion control.

The false-negative ratio for the two controlled nodes was 3.4% in this experiment. In many ways this setting represents a best-case scenario for VOID, because node proximity meant that the nodes were consistent strong interferers and because the constant TCP stream ensured that data-rate fluctuations seen by VOID were mostly due to interference and not traffic variation.

Consistency Analysis

To get a better handle on additional false negatives that might occur in uncontrolled environments such as those captured by our UBC wireless traces, we devised an indirect false-negative approximation that measures the consistency of VOID's detection algorithm.

In this experiment we track all node pairs that VOID detects as interfering at least three times at any point during the trace. We then classify the activity of these node pairs for the full trace duration to conservatively identify intervals in which VOID may have failed to detect interference between them. This analysis embeds two conservative assumptions for these node pairs. First we assume that the initial VOID detection is accurate, which is likely since VOID has relatively few false positives; if it is not, we over count false negatives. Second we assume detected node pairs are potential interferers for the entire duration of the trace, which may not be true due to node mobility or changes in transmission rate, power adaptation, or channel conditions etc. The only parameters that we know are operating channel and access-point association. Of course, a limitation of this approach is that it will not count false negatives for node pairs VOID has never

detected as interfering.

We examined seven days of trace data collected from the active periods of Highrise (i.e., 18:00 to 23:59) and LSC (i.e., 10:00 to 18:00) and identified 25 interfering pairs in the Highrise trace and 35 in the LSC trace. We treat these pairs as true interfering devices, and look for cases where VOID fails to correlate them when it should. For example, if one of the pair is not sending any traffic, then it is obviously that they indeed do not interfere during that time. We list the full list of causes below. Whenever we cannot find explanations for an undetected case using any of the known reasons, we treat it as a false negative. In other words, this experiment is designed to bucket all undetected cases into two classes. One is the known interference undetected cases where we known there is no interference (no traffic, for example) or VOID is incapable of detecting interference (traffic is low, for example). The other is the unknown interference undetected cases and we assume all of them are false negatives and are counted against VOID. We use the following categories:

- **Inactive:** One or both devices were not sending any traffic.
- **Orthogonal Channels:** Devices were operated on non-interfering, orthogonal channels.
- **Topology Change:** Devices switched off to different APS; topology change affected the underlying interference relationship.
- **Other Interferers:** Devices were found to interfere with other devices, as an interfering device can be masked by other significant interferers.
- **No Congestion:** Devices are sending a lower total throughput than the minimal throughput when they were found interfering, so the network is likely not congested.
- **Bursty Traffic:** Devices were not sending longer enough traffic, i.e., no more than one-minute traffic in each two-minute time bucket, to provide sufficient samples to VOID for analysis.

	Pairs	Reason	Percentage of Time Buckets	
			Mean (Std.)	Median
Highrise	25	Inactive	60.5% (22.6%)	61.9%
		Orthogonal Channels	12.3% (19.5%)	3.9%
		Topology Change	1.0% (4.0%)	0.0%
		Other Interferers	2.4% (5.1%)	0.4%
		Bursty Traffic	1.3% (1.8%)	0.4%
		No Congestion	18.7% (15.5%)	10.4%
		Interference Detected	3.0% (7.6%)	1.3%
		Unknown (**False Negatives**)	1.0% (1.9%)	0.0%
LSC	35	Inactive	79.3% (15.1%)	82.7%
		Orthogonal Channels	10.2% (10.4%)	6.7%
		Topology Change	1.1% (1.6%)	0.0%
		Other Interferers	2.4% (3.6%)	0.2%
		Bursty Traffic	0.4% (1.0%)	0.1%
		No Congestion	4.1% (4.5%)	2.3%
		Interference Detected	1.8% (2.7%)	0.9%
		Unknown (**False Negatives**)	0.7% (1.6%)	0.0%

Table 4.7: False-Negative Analysis in Live Networks

The analysis results are summarized in Table 4.7. We can see that the top three reasons when the pairs were not identified as interfering are (1) one of the devices is inactive (2) they are operating on orthogonal non-interfering channels and (3) there is no congestion. These three causes together account for more than 90% of the cases when interference is not detected. It is also noteworthy that the no-congestion and bursty time buckets are not exclusive; in fact, more than 90% of the no-congestion time buckets are also bursty-traffic buckets. Finally, there are 1% and 0.7% un-explained time buckets in each building when VOID is unable to detect interference.

We define False Negative Ratio (FNR) as the ratio of false negatives to the total possible interferences (the time interference is detected + the false negatives), and

it is 25% and 28% in Highrise and LSC respectively.[2] These numbers suggest that VOID might have missed one out of four interferences in these live networks.

4.6.3 Discussion on Trade-Offs

Trade-Off between False Negatives and False Positives

It is possible to improve this false negative number by relaxing two of VOID's key parameters: R^2 and VTBR. The previous experiments used values of 0.01 for R^2 and 0.5 for VTBR; lowering either of these numbers increases the number of interference cases that VOID reports and thus potentially decreases the number of false negatives.

To investigate the impact of changes to these two parameters, we re-ran the experiments with settings of 0.002 for R^2 and 0.1 for VTBR. With these settings, the analysis described above reported a 10% false negative ratio; down from 25%. However, as we can see from Figure 4.8 and Figure 4.9, this change also increases the false positive ratio from 3% to more than 50%.

Therefore, the network administrators need to understand the costs of false positives and false negatives before making a trade-off between them. The cost of false positives is to falsely identify a group of devices as interfering. The impact on Shaper is to throttle these devices unnecessarily, and reduce the overall 802.11 network capacity. The cost of false negatives is to unsuccessfully identify the interfering devices to a victim device and thus it can continuously suffer from poor performance, even bandwidth starvation.

In our experiments, we tune the settings in favor of low false positives so that the throttling (i.e, Shaper) can be enabled with high confidence. In environment where network administrators are more concerned about fairness and poor performing devices, they could tune the settings in favor of low false negatives instead. Furthermore, when deployed in real networks, VOID can be run with different settings in parallel and provide network administrators with different sets of

[2] $FNR_{Highrise} = \frac{1\%}{1\%+4\%}; FNR_{LSC} = \frac{0.7\%}{0.7\%+1.8\%}$

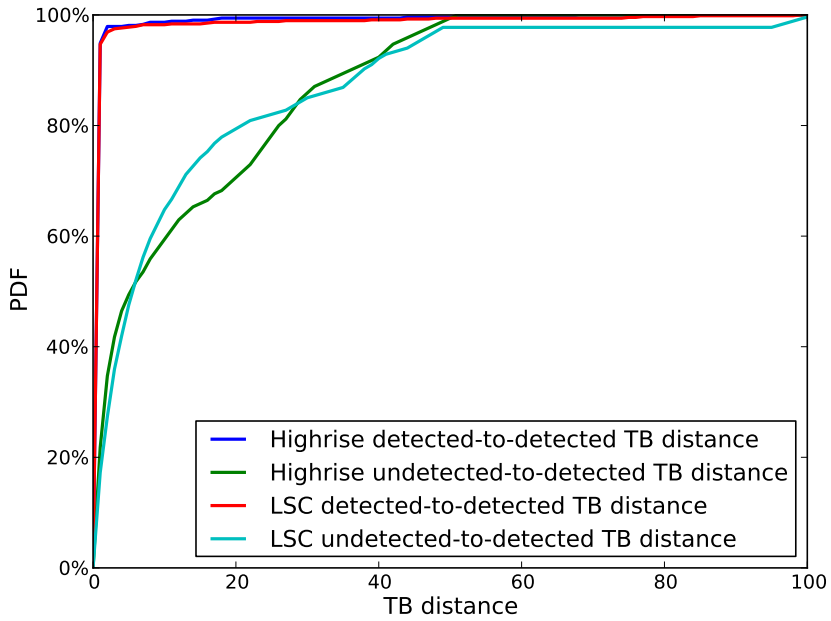


Figure 4.10: Closest Distance Distribution from a Detected TB and False-Negative TB to Its Nearby Detected TB. Note that Neighboring TBs Are 30 Seconds Apart.

interfering devices and associated false-positive and -negative values. This work, however, is left for future work.

Trade-Off between Long-Lived and Short-Lived Interference

Another approach that may reduce the number of false negatives is to focus only on long-lived interference, ignoring false negatives that occur for only transient periods. This approach is motivated by the observation that long-lived interference is more costly than transient problems and is more amenable to repair.

To examine the impact of this approach on false negatives, we computed the time distance between each false negative our original analysis detected to its closest successful detection. The results are illustrated in Figure 4.10. We can see that time distance in both buildings follow a similar pattern: 95% of the interfer-

ence detections are immediately adjacent to another successful detection, while the false-negatives are a little farther from the nearby detections. However, half of the false-negatives are within 3 minutes of a detection, and three quarters of them are within 10 minutes.

And so for this dataset, an operator concerned with congestion periods of no less than three minutes would see the algorithm miss only 12%, as half of the false negatives would have been close enough to a positive indication to not matter. Similarly, in cases where the minimum interference interval of interest is 10 minutes, the algorithm has a false-negative rate of only 7%.

Depending on how an interference mitigation system uses the interference map produced by VOID, VOID could be configured to treat the detected pairs as interfering for a longer period once they are detected. Take Shaper for example, it can make effective use of VOID to trigger traffic shaping for a period of 3 to 10 minutes, where VOID's false negative ratio is lower. The reasons that Shaper can tolerate these longer interference periods are two fold. First, since the false positive rate of VOID is as low as 3%, Shaper has high confidence that it is throttling the correct pairs as instructed by VOID. Second, short-lived interference problems will typically resolve themselves quickly, so Shaper need to kick in only when interference is persistent. Moreover, VOID is more likely to miss interference close to nearby positive detections. Therefore, to be certain to address long-lasting interference in the network, it is preferable to keep traffic throttling running for a relatively longer period once Shaper is triggered.

4.7 Interference in Live Networks

In this section, we summarize what we observed and learned from the real network traces and the detected live interferences. In particular:

- Interference occurs more often when network traffic is high, indicating that VOID is able to detect interference when congestion occurs.
- Interference occurs between devices close-by – 99% of the interference de-

tections in Highrise happened between APS 3-floors apart and 66% of them are between devices associated with the same AP.

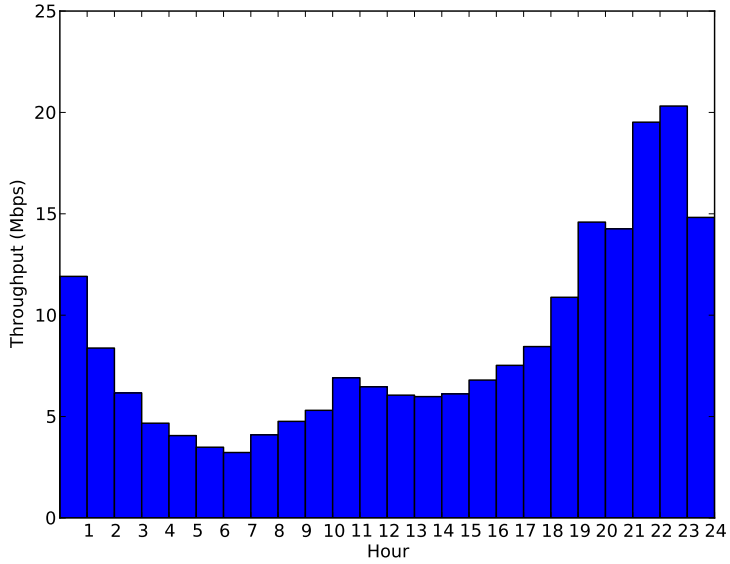
- Interferences detected last for 10 minutes on average, the longest interference lasts a few hours.
- VOID detected interference 2686 times in Highrise and 2139 times in LSC, most of which are pair-wise interferences. But VOID has found interferences including up to 5 devices.
- The real traffic is very bursty: more than 60% of the device activities last less than 1 second. Only 0.45% of device activities last more than 1 minute, but they contribute to more than 80% of the overall network throughput.

4.7.1 Interference Hourly Trend

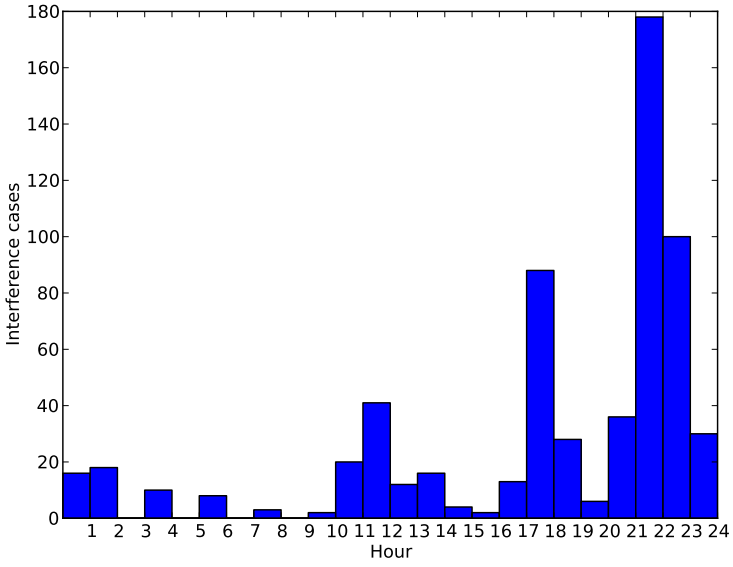
Figure 4.11 illustrates the correlation between throughput and interference in Highrise and LSC: Figure 4.7a and Figure 4.10c show the hourly mean-throughput variations in these two buildings over the five weekdays[3] and Figure 4.7b and Figure 4.10d represent the average interference detections in every hour.

We can clearly see from Figure 4.11 that the interference trend in both buildings follows the throughput variation closely: VOID is able to detect more interferences when traffic is high and less when traffic is low. In Highrise, the majority of interferences are detected at night after 5PM; while almost all interferences in LSC are detected between noon to 8PM. This difference is because that Highrise is a residential building where activity is higher at night when people are at home while LSC is a academic building where the activity happens mostly during the day time. This correlation between throughput and interference detection indicates that VOID is more likely to find interference when network is more congested, and it is effective in both environments.

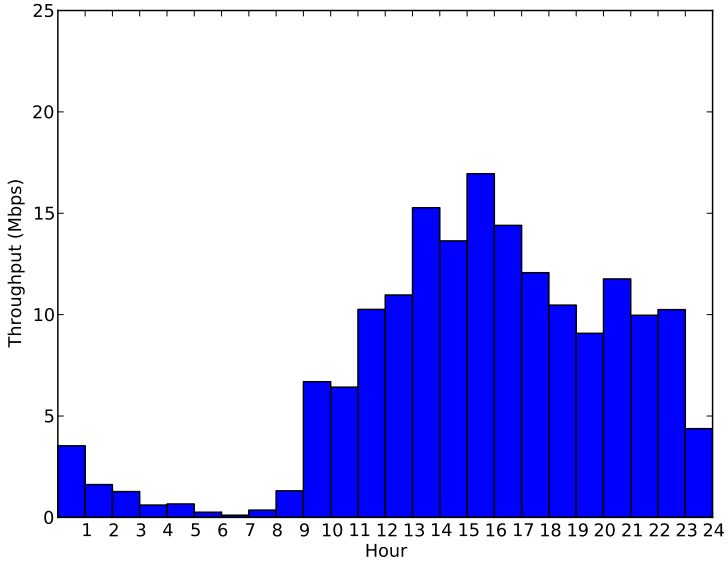
[3] 03/23/2009 to 03/27/2009 in Highrise and 04/13/2009 to 04/17/2009 in LSC.



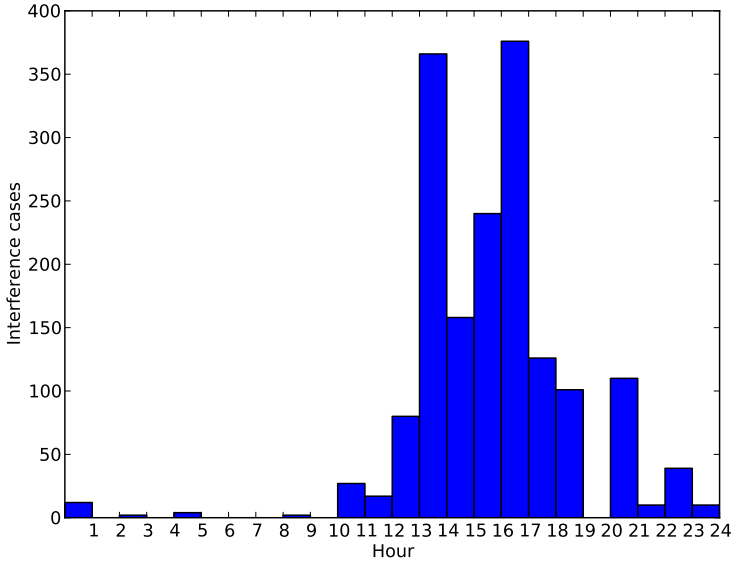
(a) Highrise Throughput



(b) Highrise



(c) LSC Throughput



(d) LSC

Figure 4.11: Average Throughput and Interference Detection Hourly Trends

4.7.2 Interference Map

We use the interferences detected by VOID to understand the interference relationships between APS in Highrise. We chose Highrise instead of LSC in this experiment because the AP placement in Highrise is more strict and organized in LSC due to space constraints, and therefore the AP interference in Highrise is mostly determined by floor distance. In Highrise, each floor is equiped with two APS (except the first floor) along the elevator wall. In LSC, there are around 10 APS on each floor and thus the interference map is really a 3D map, and interference distance (hops) is harder to generalize.

We analyzed the total 1621 same-channel multiple-device interferences detected by VOID in Highrise from Mar. 22, 2009 to Mar. 28, 2009. We correlated the APS to which these detected devices are associated and in total there are 2686 pair-wise AP interference cases were identified. Figure 4.12 shows that 99.1% of the interferences occurred between two APS that are fewer than 3 floors apart. This result is not surprising that APS should be less likely to interfere if they are farther away. Note that 2068 out of the 2686 interferences occurred on the same floor and 1769 of them are the same-AP interferences.

4.7.3 Interference Groups and Duration

VOID has detected a total of 150 interference pairs and 2686 occurrences in Highrise, and 126 interfering pairs and 2139 occurrences in LSC. On average, each pair is found to be interfering 18 times in Highrise and 17 times in LSC, which is about 10 minutes in length. We also found that the longest interference in Highrise lasts 9 hours and that in LSC lasts around 2 hours.

Finally, even though the majority of interference cases in these live networks are pair-wise only, there are still 381 and 303 three-device interference cases, 15 and 89 four-device cases in Highrise and LSC respectively. VOID has even detected five-device interfering cases for 10 times in LSC.

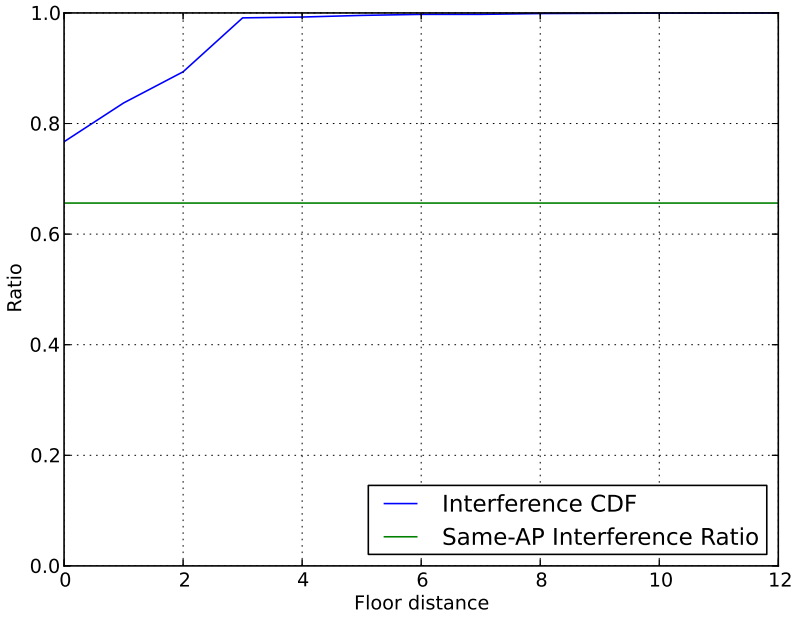


Figure 4.12: The Interference Map in Highrise.

4.7.4 Traffic Burstiness and Class

In our earlier false positive analysis in Section 4.6.1, we show that setting VTBR at 0.5 is necessary to achieve a false positive ratio less than 5%. This setting, however, excludes devices with traffic burst shorter than a minute out of our interference analysis. In this section, we investigate how much traffic and devices are affected by this filter.

We present two cumulative distributions in Figure 4.13: the probability distributions of devices' activity duration and their corresponding total throughput. These statistics are extracted from a total of 3.6 million of device transmissions (1.85 million in Highrise and 1.76 million in LSC) over the 5-weekday traces. We can see from Figure 4.13 that the devices' transmission durations range from less than 1 second to as long as almost 27 hours in both buildings and they follow the similar pattern: more than 60% of devices' activities, shown as the blue

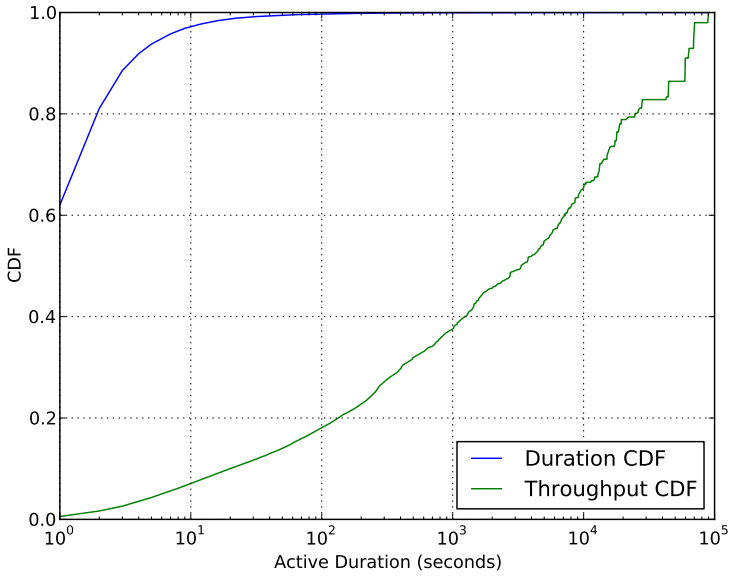
lines in Figure 4.13a and Figure 4.13b, last less than 1-second. Only 0.45% of the activity durations in Highrise and 0.35% in LSC are longer than 60 seconds. However, these longer than 60-second activities contribute to more than 85% of total network traffic in Highrise and 78% of that in LSC. We finally looked into the traffic class in both buildings, and found that 75% of the flows in LSC and 65% in Highrise operate on the unassigned and dynamic ports. The second largest traffic is HTTP/HTTPS traffic: 15% in LSC and 22% in Highrise, followed by DNS, MSNP and ICMP protocols.

This data shows that even though VOID has only selected a small set of devices in the network for correlation, it is still quite useful in UBC live networks since it has included the majority of traffic for interference analysis. In fact, VOID is designed to operate only when network is saturated, excluding most inactive devices is not going to impact its effectiveness in practice.

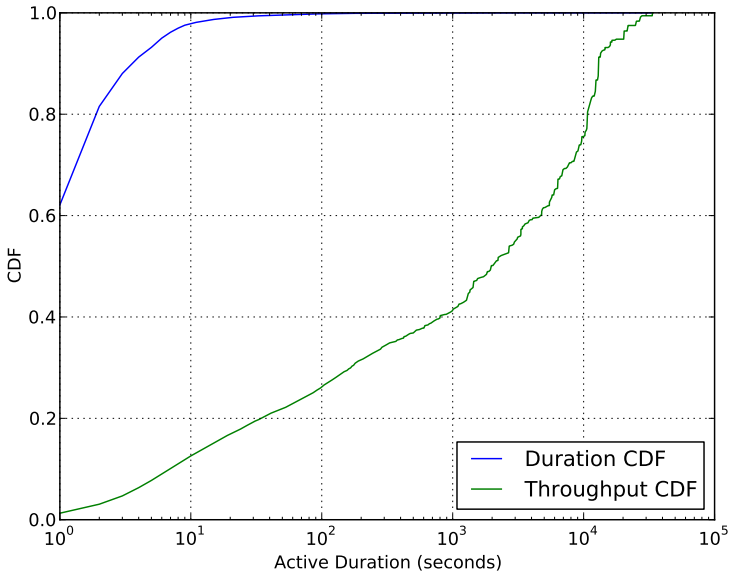
4.7.5 Scalability

Given k interferers and n throughput samples ($n > k$), the upper-bound complexity for solving one MLR regression is $O(n^3)$. The formula to solve multiple regression is $\hat{\beta} = (X^T X)^{-1} X^T y$, bounded by the Gaussian elimination procedure used to invert a matrix. We use `hotshot` [3] to profile a scenario where the numbers of interferers and samples are set to 100 and 2400 (all the throughput points) respectively. In this scenario, VOID takes 0.05 seconds to solve one regression on an Intel Xeon 1.6GHz machine.

The time for VOID to converge is also determined by the total number of regressions needed to remove all the false interferers from the model. In a typical wireless campus network we can expect a list of 100 potential devices (5 nearby APS working on the same channel with 20 associated clients each) where 90% of them are false interferers. VOID then needs 50 seconds to run 1000 regressions to identify these 10 interferers. In the live Highrise and LSC buildings, the number of active devices included for regression is only about 20 and the majority is pair-wise interference, so VOID needs only one tenths of a second to converge.



(a) Highrise Duration CDF



(b) LSC Duration CDF

Figure 4.13: Duration CDF in LSC and Highrise

4.8 Conclusion

In this chapter, we propose an online, passive interference detection approach for enterprise wireless networks called VOID. It takes in throughput traces collected from a central router and outputs a list of interferers to a victim node. The salient features of VOID are that it uses online traces for analysis and that it is fast enough to track interference relationship changes in real time. We have conducted a variety of experiments on the Emulab wireless testbed and the live wireless UBC networks, and the results have shown that VOID is able to accurately and quickly detect the true interfering devices in changing environments.

Chapter 5

Related Work

My thesis is inspired by many of the prior research work. This section provides an overview of the related work on interference analysis, mitigation and diagnosis.

5.1 Interference Analysis

The performance woes of 802.11 competing flows, i.e., inefficiency and unfairness, are fundamentally caused by the protocol design. 802.11 an autonomous protocol that wireless devices operate using the CSMA/CA protocol. A sender senses airspace activity before transmitting a packet – the sender will only start transmission if the airspace is idle and backs off otherwise. However, as we showed in Chapter 2, when competing flows span over multiple sensing ranges, CSMA/CA is no longer effective: Senders can be out of range so that their transmissions can still collide or one sender can compete with more devices than the others.

Prior studies have identified one main culprit: problematic topologies. One attempt is to distinguish between hidden-terminal and exposed-terminal topologies [30, 31, 62]. Chen et. al. further points out the importance of incomplete channel status assessment and inconsistent channel status [40]. Incomplete channel information leads to packet collisions; inconsistent channel information leads

to unfair channel sharing. These categorizations are both correct, however, they are not specific enough to help wireless devices to adapt to continuously changing environments.

Jian et.al also point out that sensing range is not the same as transmission range and Additive Increase Multiplicative Decrease (AIMD) of 802.11 cannot achieve MAC-layer fairness in various settings [60]. They propose a new rate control protocol, called Proportional Increase Synchronized multiplicative Decrease (PISD), to achieve fairness in 802.11 networks. The key idea is two fold: one is to enable the victim nodes to jam the airspace so that all devices, including the winning devices, will back off synchronously; the other is to assign weighted fairness for different MAC flows so that they have different additive increase rate to achieve the fairness an administrator desires.

There have also been several analysis works [32, 54, 64, 104] on 802.11 MAC DCF protocol performance including throughput, delay, queue performance, etc. These models are, however, mostly analytical rooted at the Markov Chain model proposed by Bianchi [32]. The work from Kim et al. [64] is based on a fluid model. Our models, on the other hand, follow a more experimental methodology to examine the fairness and performance issues when competition occurs.

If we consider only two sending nodes, there are only two possible ways that they can interfere with each other: (1) *receiving interference* and (2) *carrier sensing interference* [73]. Receiving interference occurs when two senders are out of radio range of each other, but one sender's packets are corrupted at the receiver by another sender's signals; carrier sensing interference happens when the victim node is able to sense the interfering node's signal so that it cannot send when the interferer is transmitting. These pair-wise sender/receiver interference models are the building blocks needed to estimate the interference level between multiple interfering nodes and the victim node [69, 77, 82].

The work from Garetto et. al. [50] analyzes 16 two-competing-flow topologies, and categorizes them into four interference models. In Chapter 2, we extended their work to show that, in addition to node topology, signal strength and

traffic type can also impact the way devices interfere with each other. We investigate 698 scenarios and propose 16 more models, and it is impossible to enumerate all possible interference cases even for only two competing flows — the number of scenarios continues to grow exponentially as we include more environmental parameters into the model.

5.2 Interference Mitigation Approaches

IEEE 802.11 standards board committees continue to develop new 802.11 protocols to provide better network performance to wireless users. The MIMO technology [79] deployed in IEEE 802.11n [12] takes advantage of multi-path signal propagation and CRC encoding to correct corrupted signal at a receiver. The 802.11k standard [11] associates a wireless station with a weaker-signal but underutilized AP if the strongest-signal AP has already been over subscribed. These protocols however suffer from the same interference issue when airspace is already saturated. Note that 802.11ad [16] standard can operate on the unlicensed 60 GHz frequency band and deliver data transfer rates up to 7Gbps. However, its effective bandwidth can be significantly reduced when operating with legacy 802.11 devices. Given that there are already hundreds of millions of legacy 802.11 devices sold worldwide, 802.11 network congestion problem is expected to continue for quite some time.

Researchers have proposed different solutions to alleviate the impact of wireless interference. Self-adaptation in 802.11 networks has drawn a lot of attention to improve performance for 802.11 devices. Some have investigated physical carrier sensing in detail [45, 57, 97, 98, 103]. Their intention is to adjust or disable carrier sensing function so as to maximize spatial reuse and avoid packet collisions. The other adaptation mechanisms include altering the MAC backoff durations [90, 92], enabling RTS/CTS virtual carrier sensing [40, 61], switching from sender-initiate mode to receiver-initiate mode [40, 48], and adapting the transmission rate and time scheduling [63, 84]. These approaches, however, require changes made to the driver or firmware, which is hard to implement and adopt.

Also, most of the present-day 802.11 devices lower their sending rate when experiencing poor performance. This is because lowering the rate can increase the transmission range so that the interfering nodes are more likely to sense/capture its packets and consequently back off. Also, packets sent at lower rate are more robust to interference or noise. However, previous work has also shown that lowering the sending rate results in significantly low network utilization [89] and, in certain cases, can even make the unfairness problem worse [95].

Improving network performance and fairness for cooperative wireless LANs has attracted a lot of research attention. Some require all the wireless devices to communicate with each other (or with a master node) to come up with a globally consistent schedule for each device to access the network [2, 65, 91, 92]. Some use load-balancing schemes that shuffle wireless users between different access points according to the changing network and traffic condition [25, 27]. Others limit the number of clients associated with each AP to avoid network congestion from happening in the first place [11, 27, 59].

The CENTAUR work from Shrivastava et al. [87] treats downlink traffic and uplink traffic differently in enterprise WLANS. It uses a centralized scheduling approach, called DET, to handle all downlink hidden and exposed terminal packets, and uses 802.11 for all other packets including uplink and other non hidden or exposed terminal downlink packets. When DET detects the possibility of a hidden terminal conflict, it delays the transmission of some packets to avoid collision and packet loss. For an exposed terminal conflict, DET introduces an approach called packet staggering to keep these packets in sync to be transmitted simultaneously. CENTAUR requires significant changes to APS to keep them in sync and to construct the interference map using micro-probing [21].

Comparing to all above approaches, Shaper emphasizes the correlation between 802.11 network congestion and unfairness, and points out that a much easier way to address unfairness is to alleviate congestion, requiring no modifications to either APS or end-user WiFi devices. Note that our Shaper does not intend to compete with any of the above proposals but could co-exist to address WLAN

problems together.

Traffic limiting has been proposed to address many different problems in the literature. Gambiroza et. al. [49] propose to use rate limiting in order to achieve per-TAP (Transit Access Points) fairness between long-hop flows and short-hop ones in multi-hop wireless mesh networks. Portolés et. al. [76] use shaping to avoid a disconnecting/de-associating station from causing undesirable impact on the other devices within the same network. Chiasserini et. al. [44] also suggest to regulate bluetooth traffic to improve performance for 802.11 flows. Our Shaper, on the other hand, is designed to address unfairness caused by asymmetric topologies or inequitable channel conditions between competing devices in WLANs, especially when these devices are associated with different access points.

5.3 Interference Correlation

To effectively mitigate interference in the network, the first step is to understand the interference relationship between the devices quickly and accurately. Many prior researchers have proposed to automate problem diagnosis in 802.11 networks [19, 24, 38, 41, 42, 52, 58, 69, 86]. These systems are designed to detect network anomalies such as rogue APs, association and authentication failures and signal interference using network statistics collected from distributed sniffers.

DAIR [24, 39], WiFiProfiler [38], MOJO [86] and [19] require cooperation from wireless clients. In most of these systems, the clients and APs collect network statistics and send them to a central point for problem diagnosis. WiFiProfiler enables mobile clients to exchange sensing information and help each other in an ad-hoc way.

Unlike these systems, Jigsaw [41, 42] and WIT [69] deploy dedicated monitors near the APs, with the goal to systematically capture all the link-layer packets in the airspace. These systems (as well as MOJO [86]) the merge all the traces to generate a single unified picture of network activity. This time-synchronized, detailed trace allows these systems to diagnose implicit physical layer anomalies such as hidden-terminal topologies and the capture effect. These systems require

networking administrators to set up dedicated monitoring nodes throughout the network, doubling the deployment and maintenance cost.

The studies from Padhye and Woo [74, 96] have demonstrated that interference and conflict maps based on real measurements are more accurate than those based on theoretical RF models [56]. In the measurement phase, the interference level between pairwise devices is calculated by instructing each device (or a pair of nodes) to broadcast or unicast messages in turn, while keeping the other nodes observing. Dragoş Niculescu has shown that the network performance in complex-traffic and multiple-sender scenarios can be predictable using these simple, low complexity pairwise measurements [73].

Conducting these measurements, on the other hand, is quite time consuming. Therefore, some prior work [69, 77, 82] has proposed to combine together the best part of both approaches together — real measurements and theoretical interference models. This approach uses measurements to accurately capture the RF characteristics of a given wireless network. The measurement results are then used to seed a variety of interference models such as RSSI/PDR [82], 802.11 state machine [69] or Markov chains [77] to predict run-time performance given traffic input.

Researchers have also proposed to use online traffic to generate the conflict graph for a network [88, 93]. The conflict graph work [93] requires each sender to maintain a local conflict table, updated by the feedback (packet delivery probability) from its receiver. This table is then exchanged among senders. If the conflict table suggests that a sender does not experience poor packet delivery ratio while sending simultaneously with another sender, in a typical exposed terminal topology for example, then it should ignore the other sender's signals to improve the overall network utilization.

Ahmed et al. proposed a technique called micro-probing [21] to construct an interference map without taking the network offline. The access points use MAC-layer CTS-to-self or ACK packets to silence the airspace before initializing an interference measurement. The interference analysis is carried out quickly,

taking under 20 seconds to complete in a 20-node network. However, this airspace silence approach is not very effective in a real network due to two reasons. One, not all WiFi devices have properly implemented the 802.11 standard. Second, some devices could have failed to decode the silencing packets.

The later PIE work [88] instructs the access points to record the time and successful transmission rate when they transmit packets, and report this information to the central controller. The controller then uses this information to infer which transmitters are mutually interfering. Their approach, however, still requires driver/protocol modification and cooperation from both APS and mobile clients.

5.4 Recent WiFi Interference Research Work

5.4.1 Interference Analysis

We continuously observe the evolution of WiFi standards over the last few years to provide very high link bitrates. The 802.11ac WiFi standard [17], for example, promises a bitrate of up to 1.3Gbps. The significant theoretical performance improvement is largely due to two factors: (1) denser modulation type and coding rate techniques and (2) Multiple Input and Multiple Output (MIMO) that multiplies the capacity of a radio link using multiple transmit and receive antennas.

However, Bharadia et.al [28] point out that users often do not realize these performance improvement in practice, experiencing raw speeds that are one to two orders of magnitude less than the advertised speeds. They identify two causes: propagation loss and MIMO rank degradation. They show that wireless devices at the edge in a home experience SNRs between 0-6dB, leading to a 4x bitrate reduction. Furthermore, they show that in many indoor and urban scenarios, there is only a single strong path exists between an AP and a client, and the rest are weak or non-existent, thus rendering MIMO useless.

Recent work [75] collects wireless performance traces from 30 homes in Chicago for a period of 6 months for performance measurement and analysis. Their results

show that around 8% of clients experience poor performance for greater than 10% of their active periods. They also show that in a dense deployment of private APS, high airtime utilization from neighboring APS was the major cause of performance degradation while low performance due to weak signal strengths were more prevalent in a centralized home deployment. They also show most of the traffic is short-lived and thus hidden-terminal interference can occur intermittently. But long-term stream traffic such as Netflix can lead to long term hidden-terminal interference, as long as 87 minutes. Note that these findings nicely matches our findings in UBC campus networks as described in Section 4.7.

Finally, studies in 2010 have shown that non-WiFi interference sources are major problems for 802.11 networks [13, 14] — the highpower interferers like baby monitors and cordless phones can cause 802.11n networks to experience a complete loss of connectivity. Customer survey from Cisco [15] revealed that RF interference is the top issue causing wireless network performance problems, and about 50% of the interference is from non-WiFi interference sources.

5.4.2 Interference Mitigation

Recent researches take advantage of recent technological advances, specifically Interference Alignment and Cancellation (IAC), beamforming and wider spectrum, to mitigate interference for 802.11 devices. In this section, we discuss recent interference mitigation systems utilizing these key ideas.

Interference Alignment and Cancellation

A MIMO sender cannot send more packets concurrently to its receiver than the number of receiving antennas. Otherwise, the receiver will not be able to decode these concurrent transmissions apart from each other. IAC [51] is proposed to improve network throughput by enabling even more concurrent transmissions. IAC requires the senders to align their transmissions in a special way so that all the interfering packets to a receiver is aligned, and thus the receiver is able to decode one packet successfully. Then the receiver will send the decoded packet to the

other receivers so that they can perform interference cancellation to subtract the effect of the known packet from the signals they received to decode the remaining packets. With this approach, 802.11 network throughput is no longer bounded by the number of antennas. Recent work [66] extends IAC work by enabling MIMO devices to compete for airspace without central coordination, even in the presence of ongoing transmissions, using multi-dimensional carrier sense on orthogonal channels. Bharadia et. al. [29] introduce full duplex ratios that use novel analog and digital cancellation techniques to cancel the self interference to the receiver noise floor. This approach ensures no degradation to the received signal and therefore a radio can now send and receive signals simultaneously.

Recent work from Bansal et. al. [26] proposes a system called Symphony, a packet recovery architecture that encourages collisions among transmitters, and utilizes the unused capacity in the backbone to transmit recovered data packets and coordinate the efficient recovery of collided packets. It does not require each device to have multiple antennas, but it does require APS are all connected over ethernet and able to receive decoded packets from each other for interference cancellation. BBN [105] takes advantages of the high density of APS in a single domain and requires a subset of APS to retransmit the received signals (not packets) wirelessly to nullify the interference at other APS. Once any packet is successfully decoded, the packet will be transmitted to other APS via ethernet. Both Symphony and BBN are designed for up-link transmission only.

Multi-user Beamforming

Multi-user Beamforming (MUBF) [99] is designed to serve many users simultaneously. It multiplies each individual data stream by its appropriate beamforming weight vector, aggregates the resulting streams to one, and then transmits the summed streams in parallel using an antenna array. The weight vectors are chosen carefully based on independently fading channel conditions to the receivers so that mutual interference among different streams is reduced (or even eliminated) at each receiver. Recent work [23] demonstrates that MUBF performs very well

even for small-sized mobile devices even with high mobility and limited power. The authors also show that, to make a tradeoff between power and throughput, an optimal number of active antennas (or an optimal beamforming vector size) can be chosen based on the channel conditions and throughput demands. An adaptive algorithm, called BeamAdapt, is proposed to iteratively adjust the beamforming size at each sender solely based on the SINR at its own receiver.

Rahul et. al. [78] present a system, called Jointed Multi-user Beamforming (JMB), which enables independent APS to beamform their signals and communicate with their clients on the same channel as if they were one large MIMO transmitter. The challenge, however, is to synchronize the phases of multiple APS in a distributed manner so that, at each client, the signals intended for the other clients can cancel each other out. JMB uses a simple approach by choosing one AP as a lead and using its phase as a reference to align the phases of all other APS in the network.

Recent work from Yu et. al. [100] propose a system called CoaCa to combat inter-cell interference in 802.11ac-based multi-user MIMO networks. It is a two-step approach that utilizes both interference cancellation and beamforming technologies. The first step is to let each AP and client optimize the use of its antennas for either data communication or inter-cell interference cancellation, to maximize the total number of deliverable streams in the network. Once the active antennas are determined and channel conditions are overheard, each node can obtain enough channel knowledge to optimize its beamforming weights.

Spectrum Management

The new 802.11 standards support wider channel width (spectrum) for higher throughput. IEEE 802.11n [12] supports up to 40 MHz channels and IEEE 802.11ac [17] further increases the channel width to 160 MHz. However, wider channels are more susceptible to interference in reality, and thus bandwidth/spectrum management needs to be dynamic and adaptive.

Rayanchu et. al. [81] have shown that spectrum allocation needs to take into

account the interference parameters like carrier sensing, hidden terminals etc., which depend on the combinations of frequencies and channel widths used, as well as the specifics of topology and traffic demand. They also develop a modeling framework to efficiently compute the conflict graph for an N node network with k bitrates employing flexible channelization using only $O(N.k)$ empirical measurements. Finally, a central controller assigns the center frequencies and widths to the APS on the fly, depending on the actual traffic demand. Recent work from Yun et. al. [101] further improves the above system by enabling spectrum adaptation on a per-frame basis and allowing an AP to transmit multiple frames simultaneously.

Chapter 6

Conclusion and Future Work

In this thesis, I present our research results on interference analysis, mitigation and detection. This section concludes our approaches and contributions, and provides possible future research directions.

6.1 Conclusion

Interference is omnipresent in 802.11 networks, and it is already well known that strong interference can lead to severe performance degradation and extreme unfairness for wireless devices. The cause is tied with the fundamental design of 802.11 protocol. The CSMA/CA scheme allows each sender to make its own decision on when to access the network locally, requiring no arbitrators for coordination. However, these individual decisions together might result in non-optimal bandwidth utilization and distribution especially in multi-hop networks where local sensing might miss transmissions out of its range. This thesis aims to understand the cause of interference better, detect interference quicker and easier, and mitigate its impact in enterprise wireless networks.

Chapter 2 investigates the cause and impact of wireless interference in more dimensions. In addition to topology, we consider sensing link state, traffic type and signal strength in our models, and in our evaluations, we show that slight

changes in one of the parameters can completely change the throughput outcome. We simulate all 648 possible two-flow scenarios by enumerating all possible combination of these three parameters, and propose 19 models based on the bandwidth utilization and fairness. Our experiments also manifest that these models are accurate enough to predict the performance for two competing flows in a testbed environment. On the other hand, while adding these new parameters provides more insights on how wireless devices perform in live networks, it becomes clear that the complexity of interference grows exponentially as one takes into account more environmental factors. Chapter 2 has already considered 648 scenarios, and we have intentionally left out factors such as sending rate, signal propagation model, more competing flows, to simplify our analysis. Therefore focusing and addressing just one network parameter might not work very well in reality.

In Chapter 3, we focus on the cross-layer interactions between the network layer and the MAC layer. The key observation is that wireless unfairness occurs only when competing devices attempt to send more traffic than the 802.11 network capacity. The experiment demonstrates that, even in the worse topology such as the hidden terminal scenario, as long as the aggregate throughput does not exceed 85% of the network capacity (e.g., 20Mbps out of 24Mbps in our testbed), both flows can always achieve fair bandwidth allocation. Therefore, we propose a system called Shaper that throttles the aggregate throughput of the interfering devices in favor of better fairness. The idea is that TCP and fair queuing algorithms such as SFQ can slow down the sending rate of the winning flows, and allow the losing flows to salvage the dropped packets using TCP or MAC retransmissions.

Finally, to identify which devices are interfering in the network, we propose an online, passive interference detection system called VOID in Chapter 4. Unlike many previous work that requires in-field measurements, VOID identifies interference relationships by correlating the throughput variations at the central wired router. This interference detection scheme works in congested networks where the throughputs of interfering devices are adversely correlated, i.e., when one's goes up, the others' go down. We demonstrate the effectiveness of VOID in both a

controlled testbed (Emulab) and live wireless networks in UBC. In both environments, VOID is able to detect the true interfering devices and filter out the false ones.

6.1.1 Contribution

The main contribution of my thesis is to understand and exploit the correlation between congestion and interference. We realize that the impact of interference, i.e., performance degradation and unfair bandwidth allocation, becomes worse when airspace is congested. If there is some bandwidth unused, the TCP and MAC layer retransmission mechanisms are still able to help the disadvantaged devices to recover their packet losses. However, as soon as 802.11 network congestion occurs, they are no longer effective. Also, under congestion, the throughputs of interfering devices follow a very clear pattern – the throughput of one flow is adversely correlated to that of the others.

Based on these observations, we propose two systems called Shaper and VOID. They aim to detect interference and mitigate its impact in enterprise wireless networks. Shaper throttles the aggregated throughput in the central router to trade overall throughput for better fairness, while VOID monitors the throughput variation at the router to infer interference relationships in the airspace.

6.2 Future Work

There are several future directions that the ideas in this thesis could be taken. It will be very interesting to deploy VOID and Shaper in live wireless networks. Also, we have to use heuristics such as channel information to infer VOID's accuracy in Chapter 4, it will be interesting to run VOID where monitoring systems such as JigSaw [41] is available so that we can have the real underlying interference map for the reference.

Another possibility for future work is to automatically determine the tradeoff between throughput and fairness. One of the parameters that affects the throt-

tling limit is the channel conditions of the losing flows; the worse their channel conditions are, the lower the throttling upper bound will be for perfect fairness. To one extreme, if one losing flow experiences many packet losses even without competing flows, then throttling is not going to be of any use but to reduce the overall network utility. The implementation of Shaper should be aware of this pitfall. What Shaper might do is to throttle only the winning flows to their fair share instead of trying to achieve perfect fairness, for better overall network utilization. In addition, a cost function can be used to increase the cost of throttling as the aggregated throughput declines.

Similarly, one of the issues that Shaper needs to consider in real deployment is what fairness it should achieve in a given wireless network. As we explained earlier, the way Shaper works is to stop congestion in the last hop and thus avoid the 802.11 protocol from making bandwidth-allocation decision. Thus, if all the devices run at the same rate, then a max-min fairness should be achieved, thanks to the TCP protocol. However, unlike the wired links using fixed rate for packet transmission, the wireless devices are likely to operate at different rates depending on various channel conditions they face, even if they associate with the same access point. When devices are running at different rates, there exists a big difference between *time fairness* and *throughput fairness* [90]. The time fairness allows the competing devices to equally share the channel access time, while the throughput fairness guarantees that they will achieve the same throughput. The difference between these two fairness schemes is due to the fact that a wireless device running at a lower rate will take much longer time to send a packet than those using a higher rate. Therefore, to achieve a throughput fairness will enable the slower devices to occupy much longer channel access time than the others, resulting in significant loss in overall network utility.

Finally, the most interesting and challenging future work is to integrate VOID and Shaper for online interference diagnosis and mitigation. It should be a continuous feedback pipeline consisting of four subsystems: (1) trace collection (2) throughput aggregation (3) VOID and (4) Shaper. As we discussed in Section 4.4,

we use port mirroring techniques to copy live traffic to another switch port for trace capture so that it will not impair the central routers' performance when serving live traffic. We then use WyPy [67] to aggregate the throughput stats for all devices seen in the trace and feed the throughput time series to VOID for interference correlation. Once the set of interferers are identified, Shaper will be notified to throttle the aggregated throughput for the affected devices.

The latency of this pipeline is mostly determined by throughput aggregation and interference correlation, i.e., the WyPy and VOID subsystems. The trace capture and traffic throttling are both almost instantaneous. On the other hand, in our live UBC networks, WyPy takes about 10 seconds to generate the throughput time series for a 2-minute trace while VOID can take up to 60 seconds to converge since it has to run multiple regressions on each of the candidate devices. The end-to-end latency of this pipeline is, therefore, at most 60 seconds.

We were able to run the first three subsystems using UBC live traces — we captured 2-minute traces and fed that to WyPy and VOID to correlate interferers. Unfortunately the permission was not granted to run Shaper on UBC central routers. Nevertheless, we were able to generate the online interferer set for the two campus buildings every minute. This latency can be further greatly reduced by running all these regressions against each candidate in parallel. In the future, we would like to evaluate effectiveness of VOID and Shaper when they operate together.

Bibliography

[1] D-itg traffic generator. URL http://traffic.comics.unina.it/software/ITG/. → pages 76

[2] Packet scheduling and qos for wireless networks. URL http://frottle.sourceforge.net/. → pages 111

[3] hotshot – high performance logging profiler. URL https://docs.python.org/2/library/hotshot.html. → pages 105

[4] Hierarchical token bucket. URL http://luxik.cdi.cz/~devik/qos/htb/. → pages 50

[5] ipt_account netfilter module. URL http://www.intra2net.com/de/produkte/opensource/ipt_account/. → pages 57, 76

[6] Madwifi. URL http://madwifi-project.org/. → pages 76

[7] Qstream. URL http://wiki.nss.cs.ubc.ca/QStream/Developers. → pages 6, 49

[8] Scipy. URL http://www.scipy.org/. → pages 74

[9] Tcpdump and libpcap. URL http://www.tcpdump.org. → pages 73

[10] The institute of electrical and electronics engineers, ieee std 802.11g - 2003 — ammendment to ieee std 802.11, 1999 edition (reaff 2003), 2003. → pages 30

[11] The institute of electrical and electronics engineers, ieee std 802.11k - 2008 — amendment 1: Radio resource measurement of wireless lans., 2008. → pages 110, 111

[12] The institute of electrical and electronics engineers, ieee std 802.11n - 2009 — ammendment 5: Enhancements for higher throughput., 2009. → pages 5, 110, 117

[13] Evaluating interference in wireless lans: Recommended practice, 2010. URL http://www.bizreport.com/whitepapers/evaluating_interference_in_wireless_lans.html. → pages 115

[14] Understanding the effects of radio frequency (rf) interference on wlan performance and security, 2010. URL http://www.techrepublic.com/resource-library/whitepapers/understanding-the-effects-of-radio-frequency-rf-interference-on-wlan-performance-and-security. → pages 115

[15] Wireless rf interference customer survey results, 2010. URL http://www.webtorials.com/main/resource/papers/cisco/paper163/wireless_RF_interference_survey.pdf. → pages 115

[16] The institute of electrical and electronics engineers, ieee std 802.11ad - 2012 — amendment 3: Enhancements for very high throughput in the 60 ghz band., 2012. → pages 110

[17] The institute of electrical and electronics engineers, ieee std 802.11ac - 2013 —amendment 4: Enhancements for very high throughput for operation in bands below 6 ghz., 2013. → pages 114, 117

[18] More than 120 million wi-fi chipsets shipped in 2005, last known valid in 2006. URL http://www.wi-fi.org/news/pressrelease-112805-120millionchipsets/. → pages 1, 44

[19] A. Adya, P. Bahl, R. Chandra, and L. Qiu. Architecture and techniques for diagnosing faults in ieee 802.11 infrastructure networks. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2004. → pages 112

[20] N. Ahmed and S. Keshav. Smarta: a self-managing architecture for thin access points. 2006. → pages 5

[21] N. Ahmed, U. Ismail, S. Keshav, and K. Papagiannaki. Online estimation of rf interference. 2008. → pages 111, 113

[22] A. Akella, G. Judd, S. Seshan, and P. Steenkiste. Self-management in chaotic wireless deployments. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2005. → pages 44

[23] E. Aryafar, N. Anand, T. Salonidis, and E. W. Knightly. Design and experimental evaluation of multi-user beamforming in wireless lans. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2010. → pages 116

[24] P. Bahl, J. Padhye, L. Ravindranath, M. Singh, A. Wolman, and B. Zill. Dair: A framework for managing enterprise wireless networks using desktop infrastructure. In *ACM Workshop on Hot Topics in Networks (HotNets)*, 2005. → pages 112

[25] A. Balachandran, P. Bahl, and G. M. Voelker. Hot-spot congestion relief and service guarantees in public-area wireless networks. *SIGCOMM Comput. Commun. Rev.*, 32(1), 2002. → pages 111

[26] T. Bansal, B. Chen, P. Sinha, and K. Srinivasan. Symphony: Cooperative packet recovery over the wired backbone in enterprise wlans. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2013. → pages 116

[27] Y. Bejerano, S.-J. Han, and L. E. Li. Fairness and load balancing in wireless lans using association control. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2004. → pages 111

[28] D. Bharadia and S. Katti. Fastforward: Fast and constructive full duplex relays. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 2014. → pages 114

[29] D. Bharadia, E. McMilin, and S. Katti. Full duplex radios. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 2013. → pages 116

[30] V. Bharghavan. Performance evaluation of algorithms for wireless medium access. In *In Proceedings of IEEE Performance and Dependability Symposium*, 1998. → pages 2, 13, 108

[31] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang. Macaw: A media access protocol for wireless lans. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 1994. → pages 2, 13, 108

[32] G. Bianchi. Performance analysis of the ieee 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, 18(3), 2000. → pages 109

[33] I. Broustis, K. Papagiannaki, S. V. Krishnamurthy, M. Faloutsos, and V. Mhatre. Mdg: measurement-driven guidelines for 802.11 wlan design. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2007. → pages 5

[34] K. Cai, M. Blackstock, R. Lotun, M. J. Feeley, C. Krasic, and J. Wang. Wireless unfairness: alleviate mac congestion first! In *Proceedings of the second ACM international workshop on Wireless network testbeds, experimental evaluation and characterization*, 2007. → pages 43, 72

[35] K. Cai, M. Feeley, and B. Cully. Understanding performance for two 802.11 competing flows. In *Proceedings of IEEE International Conference on Mobile Adhoc and Sensor Systems*, 2007. → pages 12

[36] K. Cai, J. Wang, R. Lotun, M. J. Feeley, M. Blackstock, and C. Krasic. A wired router can eliminate 802.11 unfairness, but it's hard. In *Proceedings of the 9th workshop on Mobile computing systems and applications*, 2008. → pages 43

[37] K. Cai, M. Blackstock, M. J. Feeley, and C. Krasic. Non-intrusive, dynamic interference detection for 802.11 networks. In *Proceedings of the Internet Measurement Conference on Internet Measurement Conference (IMC)*, 2009. → pages 64

[38] R. Chandra, V. N. Padmanabhan, and M. Zhang. Wifiprofiler: cooperative diagnosis in wireless lans. In *Proceedings of The International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2006. → pages 112

[39] R. Chandra, J. Padhye, A. Wolman, and B. Zill. A location-based management system for enterprise wireless lans. In *Proceedings of The*

USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2007. → pages 112

[40] C.-C. Chen, , C. cheng Chen, and H. Luo. The case for heterogeneous wireless macs. In *ACM Workshop on Hot Topics in Networks (HotNets)*, 2005. → pages 2, 13, 108, 110

[41] Y.-C. Cheng, J. Bellardo, P. Benko, A. C. Snoeren, G. M. Voelker, and S. Savage. Jigsaw: solving the puzzle of enterprise 802.11 analysis. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 2006. → pages 4, 44, 112, 121

[42] Y.-C. Cheng, M. Afanasyev, P. Verkaik, P. Benkö, J. Chiang, A. C. Snoeren, S. Savage, and G. M. Voelker. Automating cross-layer diagnosis of enterprise wireless networks. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 2007. → pages 112

[43] C. cheng Chen, E. Seo, H. Kim, and H. Luo. Self-learning collision avoidance for wireless networks. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2006. → pages 45

[44] C. F. Chiasserini and R. R. Rao. Performance of ieee 802.11 wlans in a bluetooth environment. In *Proceedings of IEEE Wireless Communications and Networking Conference*, 2000. → pages 112

[45] J. Deng, B. Liang, and P. K. Varshney. Tuning the carrier sensing range of ieee 802.11 mac. In *Proceedings of IEEE Global Communications Conference (GLOBECOM)*, 2004. → pages 110

[46] M. Fisk and W. chun Feng. Dynamic right-sizing in tcp. In *Proceedings of the Los Alamos Computer Science Institute Symposium*, 2001. → pages 52

[47] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. Netw.*, 1(4), 1993. → pages 48

[48] F.Talucci, M.Gerla, and L.Fratta. Macabi (maca by invitation): A receiver oriented access protocol for wireless multiple networks. In *Proceedings of*

The IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), 1997. → pages 110

[49] V. Gambiroza, B. Sadeghi, and E. W. Knightly. End-to-end performance and fairness in multihop wireless backhaul networks. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2004. → pages 112

[50] M. Garetto, J. Shi, and E. W. Knightly. Modeling media access in embedded two-flow topologies of multi-hop wireless networks. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2005. → pages 2, 12, 13, 17, 109

[51] S. Gollakota, S. D. Perli, and D. Katabi. Interference alignment and cancellation. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 2009. → pages 115

[52] S. Gopal and D. Raychaudhuri. Experimental evaluation of the tcp simultaneous-send problem in 802.11 wireless local area networks. In *Proceeding of the ACM SIGCOMM workshop on Experimental approaches to wireless network design and analysis (EWIND)*, 2005. → pages 112

[53] R. Gummadi, D. Wetherall, B. Greenstein, and S. Seshan. Understanding and mitigating the impact of rf interference on 802.11 networks. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 2007. → pages 5

[54] N. Gupta and P. R. Kuman. A performance analysis of the 802.11 wireless lan medium access control. *Communications in Information and Systems*, 4(4), 2004. → pages 109

[55] T. Henderson, D. Kotz, and I. Abyzov. The changing usage of a mature campus-wide wireless network. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, New York, NY, USA, 2004. → pages 3, 13, 44

[56] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu. Impact of interference on multi-hop wireless network performance. In *Proceedings of the annual*

international conference on Mobile computing and networking (MobiCom), 2003. → pages 113

[57] K. Jamieson, B. Hull, A. Miu, and H. Balakrishnan. Understanding the real-world performance of carrier sense. In *Proceeding of the ACM SIGCOMM workshop on Experimental approaches to wireless network design and analysis (EWIND)*, 2005. → pages 110

[58] A. P. Jardosh, K. N. Ramachandran, K. C. Almeroth, and E. M. Belding-Royer. Understanding link-layer behavior in highly congested ieee 802.11b wireless networks. In *Proceeding of the ACM SIGCOMM workshop on Experimental approaches to wireless network design and analysis (EWIND)*, 2005. → pages 112

[59] A. P. Jardosh, K. Mittal, K. N. Ramachandran, E. M. Belding, and K. C. Almeroth. Iqu: Practical queue-based user association management for wlans. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2006. → pages 111

[60] Y. Jian, M. Zhang, and S. Chen. Achieving mac-layer fairness in csma/ca networks. *IEEE/ACM Transactions on Networking (TON)*, 19(5): 1472–1484, 2011. → pages 109

[61] H.-J. Ju, I. Rubin, and Y.-C. Kuan. An adaptive rts/cts control mechanism for ieee 802.11 mac protocol. In *Proceedings of The Vehicular Technology Conference*, 2003. → pages 110

[62] P. Karn. Maca: A new channel access method for packet radio. In *Proceedings of Computer Networking Conference*, 1990. → pages 13, 108

[63] H. Kim and J. C. Hou. Improving protocol capacity with model-based frame scheduling in ieee 802.11-operated wlans. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2003. → pages 110

[64] H. Kim and J. C. Hou. A fast simulation framework for ieee 802.11-operated wireless lans. In *Proceedings of the International Conference on Measurements and Modeling of Computer Systems (SIGMETRICS)*, 2004. → pages 109

[65] R. R. Kompella, S. Ramabhadran, I. Ramani, and A. C. Snoeren. Cooperative packet scheduling via pipelining in 802.11 wireless networks. In *Proceeding of the ACM SIGCOMM workshop on Experimental approaches to wireless network design and analysis (EWIND)*, 2005. → pages 111

[66] K. C.-J. Lin, S. Gollakota, and D. Katabi. Random access heterogeneous mimo networks. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 2011. → pages 116

[67] R. Lotun. wypy : an extensible, online interference detection tool for wireless networks. *M.Sc. Thesis, University of British Columbia*, 2008. → pages 74, 123

[68] S. Lu, V. Bharghavan, and R. Srikant. Fair scheduling in wireless packet networks. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 1997. → pages 45

[69] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Analyzing the mac-level behavior of wireless networks in the wild. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 2006. → pages 64, 71, 109, 112, 113

[70] P. McKenney. Stochastic fairness queuing. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 1990. → pages 48

[71] A. Mondal and A. Kuzmanovic. Removing exponential backoff from tcp. In *ACM SIGCOMM Computer Communication Review*, 2008. → pages 73

[72] T. Nandagopal, S. Lu, and V. Bharghavan. A unified architecture for the design and evaluation of wireless fair queueing algorithms. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 1998. → pages 45

[73] D. Niculescu. Interference map for 802.11 networks. In *Proceedings of the Internet Measurement Conference on Internet Measurement Conference (IMC)*, 2007. → pages 12, 64, 109, 113

[74] J. Padhye, S. Agarwal, V. N. Padmanabhan, L. Qiu, A. Rao, and B. Zill. Estimation of link interference in static multi-hop wireless networks. In *Proceedings of the Internet Measurement Conference on Internet Measurement Conference (IMC)*, 2005. → pages 5, 60, 64, 113

[75] A. Patro, S. Govindan, and S. Banerjee. Observing home wireless experience through wifi aps. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2013. → pages 114

[76] M. Portolés, Z. Zhong, and S. Choi. Ieee 802.11 downlink traffic shaping scheme for multi-user service enhancement. In *Proceedings of The IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2003. → pages 112

[77] L. Qiu, Y. Zhang, F. Wang, M. K. Han, and R. Mahajan. A general model of wireless interference. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2007. → pages 60, 64, 71, 109, 113

[78] H. S. Rahul, S. Kumar, and D. Katabi. Jmb: Scaling wireless capacity with user demands. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 2012. → pages 117

[79] G. Raleigh and J. Cioffi. Spatio-temporal coding for wireless communication. *IEEE Transactions on Communications*, 46(3):357–366, 1998. → pages 110

[80] P. Ramanathan and P. Agrawal. Adapting packet fair queueing algorithms to wireless networks. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 1998. → pages 45

[81] S. Rayanchu, V. Shrivastava, S. Banerjee, and R. Chandra. Fluid: Improving throughputs in enterprise wireless lans through flexible channelization. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2011. → pages 117

[82] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Measurement-based models of delivery and interference in static wireless

networks. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 2006. → pages 60, 64, 71, 109, 113

[83] M. Rodrig, C. Reis, R. Mahajan, D. Wetherall, and J. Zahorjan. Measurement-based characterization of 802.11 in a hotspot setting. In *Proceeding of the ACM SIGCOMM workshop on Experimental approaches to wireless network design and analysis (EWIND)*, 2005. → pages 4, 44

[84] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly. Opportunistic media sccess for multirate ad hoc networks. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2002. → pages 45, 110

[85] J. Semke, J. Mahdavi, and M. Mathis. Automatic tcp buffer tuning. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 1998. → pages 52

[86] A. Sheth, C. Doerr, D. Grunwald, R. Han, and D. Sicker. Mojo: a distributed physical layer anomaly detection system for 802.11 wlans. In *Proceedings of The International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2006. → pages 112

[87] V. Shrivastava, N. Ahmed, S. Rayanchu, S. Banerjee, S. Keshav, K. Papagiannaki, and A. Mishra. Centaur: realizing the full potential of centralized wlans through a hybrid data path. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2009. → pages 111

[88] V. Shrivastava, S. Rayanchu, S. Banerjee, and K. Papagiannaki. Pie in the sky: online passive interference estimation for enterprise wlans. In *Proceedings of The USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2011. → pages 65, 113, 114

[89] G. Tan and J. Guttag. Time-based fairness improves performance in multi-rate wlans. In *Proceedings of the annual conference on USENIX Annual Technical Conference*, 2004. → pages 59, 111

[90] G. Tan and J. Guttag. Long-term time-share guarantees are necessary for wireless lans. In *Proceedings of the 11th workshop on ACM SIGOPS European workshop*, 2004. → pages 45, 58, 110, 122

[91] L. Tassiulas and S. Sarkar. Maxmin fair scheduling in wireless networks. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2002. → pages 111

[92] N. H. Vaidya, P. Bahl, and S. Gupta. Distributed fair scheduling in a wireless lan. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2000. → pages 110, 111

[93] M. Vutukuru, K. Jamieson, and H. Balakrishnan. Harnessing exposed terminals in wireless networks. In *Proceedings of The USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2008. → pages 113

[94] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar. An integrated experimental environment for distributed systems and networks. In *Proceedings of the USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2002. → pages 66, 74

[95] S. H. Y. Wong, S. Lu, H. Yang, and V. Bharghavan. Robust rate adaptation for 802.11 wireless networks. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2006. → pages 45, 111

[96] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *Proceedings of The ACM Conference on Embedded Networked Sensor Systems*, 2003. → pages 113

[97] K. Xu, M. Gerla, and S. Bae. How effective is the ieee 802.11 rts/cts handshake in ad hoc networks. In *Proceedings of IEEE Global Communications Conference (GLOBECOM)*, 2002. → pages 110

[98] X. Yang and N. H. Vaidya. On the physical carrier sense in wireless ad-hoc networks. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2005. → pages 110

[99] T. Yoo and A. Goldsmith. On the optimality of multiantenna broadcast scheduling using zero-forcing beamforming. *IEEE Journal on Selected Areas in Communications*, 24:528–541, 2006. → pages 116

[100] H. Yu, O. Bejarano, and L. Zhong. Combating inter-cell interference in 802.11ac-based multi-user mimo networks. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2014. → pages 117

[101] S. Yun, D. Kim, and L. Qiu. Fine-grained spectrum adaptation in wifi networks. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2013. → pages 118

[102] X. Zeng, R. Bagrodia, and M. Gerla. Glomosim: A library for parallel simulation of large-scale wireless networks. In *in Workshop on Parallel and Distributed Simulation*, 1998. → pages 27, 30

[103] H. Zhai and Y. Fang. Physical carrier sensing and spatial reuse in multirate and multihop wireless ad hoc networks. In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2006. → pages 110

[104] H. Zhai, Y. Kwon, and Y. Fang. Performance analysis of ieee 802.11 mac protocols in wireless lans. *Wireless Communications and Mobile Computing, Special Issue on Emerging WLAN Technologies and Applications*, 4, 2004. → pages 109

[105] W. Zhou, T. Bansal, P. Sinha, and K. Srinivasan. Bbn: Throughput scaling in dense enterprise wlans with bind beamforming and nulling. In *Proceedings of the annual international conference on Mobile computing and networking (MobiCom)*, 2014. → pages 116