

**Measurement-based Modeling of Interference in Wi-Fi Networks: Techniques and
Applications**

A Dissertation Presented

by

Anand Kashyap

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Computer Science

Stony Brook University

December 2008

Stony Brook University

The Graduate School

Anand Kashyap

We, the dissertation Committee for the above candidate for the Doctor of Philosophy degree, hereby recommend acceptance of this dissertation.

Samir R. Das, Dissertation Advisor

Associate Professor, Department of Computer Science

Himanshu Gupta, Chairperson of Defense

Associate Professor, Department of Computer Science

Jennifer L. Wong, Third Member

Assistant Professor, Department of Computer Science

Samrat Ganguly, External Member

Director, Business Development, NEC Corporation of America

This dissertation is accepted by the Graduate School.

Lawrence Martin
Dean of the Graduate School

Abstract of the Dissertation

Measurement-based Modeling of Interference in Wi-Fi Networks: Techniques and Applications

by

Anand Kashyap

Doctor of Philosophy

in

Computer Science

Stony Brook University

2008

Characterizing interference is critical to understanding the performance of a wireless network. Many protocol and algorithmic work fundamentally depend on such characterization. However, current research considers interference models that are either oversimplified or too abstract with unknown parameters limiting their use in practice. We address this issue in connection with WiFi networks (i.e., IEEE 802.11-based) due to their widespread use.

We first develop a practical, measurement-based model to estimate the capacity of any given link in the presence of any given number of interfering links in an actual deployed 802.11 network, carrying any specified amount of offered load. For a network with N nodes, only $O(N)$ measurement steps are needed to gather metrics for individual links that seed the model. We provide two solution approaches: one based on direct simulation (slow, but accurate) and the other based on analytical methods (faster, but approximate). We also show that as a by-product of our research we can create a highly accurate simulation model

(e.g., using a packet level simulator such as ns2) of a real deployed network by seeding the simulator with measurement data.

In an application of the above-mentioned capacity model, we address the issue of supporting voice-over-IP (VoIP) calls in a wireless mesh network. Specifically, we propose solutions for call admission control (CAC) and route selection for VoIP calls. Call admission decisions are made by using the capacity model to predict whether the capacity constraints at various nodes will be satisfied if a new call is admitted with a given route. We also develop a polynomial-time algorithm to search for feasible routes. In addition to studying feasibility, we study several routing metrics such as shortest feasible path and maximum residual feasible path.

The above modeling approach requires active measurements. Also, it requires instrumentation access to network nodes. These could be impractical in many deployment scenarios. To address this issue, we develop an approach to estimate the interference between nodes and links in a live 802.11 network by passive monitoring of wireless traffic using a distributed set of sniffers. We model the 802.11 protocol as a Hidden Markov Model (HMM), and use a machine learning approach to learn the state transition probabilities in this model using the observed wireless traffic traces. This in turn helps us to deduce the interference relationships. We show the effectiveness of this approach via simulations and real experiments.

TABLE OF CONTENTS

	Page
LIST OF FIGURES	ix
LIST OF TABLES	xii
CHAPTER	
1 INTRODUCTION	1
1.1 KEY COMPONENTS	2
1.2 CONTRIBUTIONS	4
1.3 OUTLINE	5
2 MEASUREMENT-BASED LINK CAPACITY MODEL	6
2.1 INTRODUCTION	6
2.2 RELATED WORK	9
2.3 MODELING APPROACH	11
2.3.1 PROBLEM FORMULATION	11
2.3.2 OVERVIEW OF APPROACH	12
2.4 MODELING 802.11 BEHAVIOR	13
2.5 ANALYTICAL APPROACH	16
2.5.1 BASELINE NOTATIONS	17
2.5.2 SENDER-SIDE INTERFERENCE	18
2.5.3 RECEIVER-SIDE INTERFERENCE	21
2.5.4 CAPACITY OF LINK	22
2.6 SOLVING EQUATIONS	23
2.6.1 EXAMPLES: TWO AND THREE TRANSMITTERS	24

2.7	EXTENSIONS	26
2.7.1	NON-BACKLOGGED INTERFERERS	26
2.7.2	MODELING UNICAST	27
2.8	EXPERIMENTAL PROCEDURE	29
2.8.1	PROFILING EXPERIMENTS	31
2.8.2	MODEL EVALUATION AND VALIDATION EXPERIMENTS	33
2.8.3	COMPUTATION TIME	36
2.8.4	VALIDATION FOR NON-BACKLOGGED INTERFERERS	37
2.9	CONCLUSIONS	38
3	MEASUREMENT-BASED APPROACHES FOR ACCURATE SIMULATIONS	40
3.1	INTRODUCTION	40
3.2	RELATED WORK	42
3.3	APPROACH	43
3.3.1	PROPAGATION	43
3.3.2	DEFERRAL AND RECEPTION	44
3.3.3	MODELING STRATEGY	44
3.4	MEASUREMENT-BASED MODELS	46
3.4.1	MODELING PROPAGATION	46
3.4.2	MODELING DEFERRAL	48
3.4.3	MODELING PACKET RECEPTION	50
3.5	EVALUATION	51
3.6	CONCLUSIONS	53
4	ESTIMATING INTERFERENCE USING PASSIVE MONITORING	55
4.1	INTRODUCTION	55
4.1.1	APPROACH	56
4.2	RELATED WORK	57
4.2.1	ANALYZING INTERFERENCE	57

4.2.2	USING DISTRIBUTED SNIFFERS	58
4.3	OVERALL APPROACH	59
4.3.1	PROBLEM STATEMENT	59
4.3.2	DISCUSSIONS	60
4.3.3	APPROACH	64
4.4	HIDDEN MARKOV MODEL FOR 802.11 MAC INTERACTIONS	65
4.4.1	MARKOV CHAIN	66
4.4.2	OBSERVATION SYMBOLS	69
4.4.3	FORMAL SPECIFICATION AND LEARNING	71
4.4.4	LEARNING DEFERRAL PROBABILITY	73
4.5	EVALUATION	73
4.5.1	COMPARISON POINTS	74
4.5.2	MICRO-BENCHMARKING WITH TWO NODES	75
4.5.3	EXPERIMENTS ON DEPARTMENTAL WLAN	77
4.5.4	SIMULATIONS	80
4.6	CONCLUSIONS	82
5	VOIP ON WIRELESS MESH NETWORKS	85
5.1	INTRODUCTION	85
5.2	RELATED WORK	88
5.3	ARCHITECTURE OVERVIEW	89
5.4	VOIP CALL ADMISSION CONTROL	90
5.5	MODELING CAPACITY UTILIZATION	93
5.5.1	INTERFERENCE MAP	93
5.5.2	CURRENT OFFERED LOAD	94
5.5.3	CAPACITY UTILIZATION AT NODES	94
5.6	ROUTE COMPUTATION	96
5.6.1	EDGE-PAIR ALGORITHM	98

5.6.2	ROUTING USING CALL STATISTICS	100
5.7	PERFORMANCE EVALUATION	102
5.7.1	EXPERIMENTAL TESTBED	103
5.7.2	EVALUATION OF CAPACITY UTILIZATION MODEL	103
5.7.3	EVALUATION OF CALL ADMISSION CONTROL	104
5.7.4	SIMULATION SETUP	105
5.7.5	FEASIBLE ROUTE CALCULATION EVALUATION	106
5.7.6	EVALUATING ROUTING USING CALL STATISTICS	108
5.8	CONCLUSIONS	108
6	CONCLUSIONS	110
	BIBLIOGRAPHY	112

LIST OF FIGURES

2.1	Example of problem.	6
2.2	Overview of the modeling approach.	12
2.3	State transition diagram for 802.11 on the sender-side.	14
2.4	Locations of the nodes on the floor map and links with more than 90% delivery ratio. Width of the map is 60m.	30
2.5	Profile for functions $f(\cdot)$, probability of deferral, and $g(\cdot)$, probability of capture.	31
2.6	CDF of error between the estimated and measured transmission capacity of senders, c_i for node z_i	31
2.7	CDF of error between the estimated and measured throughput capacity on links, $C_x^{z_i}(Z - z_i)$ for link from z_i to x	32
2.8	Summary error statistics for different models for different numbers of inter- ferers.	35
2.9	Computation time for analytical and simulation approaches with increasing number of interferers.	36
2.10	Capacity of a link (1,3) in presence of one non-backlogged interferer. Inter- ferer 2 contends for channel with 1; 4 causes collisions, while 11 does not effect the link.	38
3.1	Versions of the simulators considered and the models used by them.	45
3.2	Locations of the nodes on the floor map and links with more than 90% delivery ratio. Width of the map is 60m.	47
3.3	(a) Measurement data showing received signal strength vs. distance and also the least-square fit. (b) Empirical estimation of the shadowing model.	47

3.4	Determining (a) deferral and (b) packet reception probabilities.	49
3.5	Capture probability versus SINR in presence of one interferer.	51
3.6	CDF of error between the estimated and measured transmission capacity of senders.	51
3.7	CDF of the error between the estimated and measured throughput capacity on links.	52
4.1	State transition diagram for a single sender. CS = 0 (CS=1) means that the carrier is sensed idle (busy). Q = 0 (Q =1) means that the interface packet queue is empty (non-empty).	65
4.2	Markov chain modeling the combined MAC Layer behavior of two nodes (sender side only). Note that some arrows are bidirectional.	67
4.3	CDF of observed inter-frame times in a recorded trace for a single saturated sender.	70
4.4	Combined performance results for 11 chosen scenarios for two node experiments.	75
4.5	Locations of APs, sniffers and clients shown projected on 2D. The nodes are distributed actually over two floors – APs on ceilings, sniffers on floors and clients on floors or on tables.	78
4.6	Estimated and measured probabilities of deferral for the 16 test cases with the departmental WLAN.	79
4.7	Performance results for 20-node scenarios with the extended ns-2 simulation. Distribution of probabilities of deferral for node pairs and CDF of interference estimates for low and high traffic are shown for three network densities.	84
5.1	Architectural block diagram for the approaches developed in this chapter. The number in the parantheses in each block indicates the section where it is discussed.	89

5.2	On a 2-hop path, the graph shows how R score depends on capacity. Once the capacity threshold is reached delay increases due to queuing delay and R score goes down.	91
5.3	Transmission range (R) = carrier sense range (S). In the worst case scenario, two nodes A and E 4 hops apart in a path do not share any node in their carrier sense ranges.	97
5.4	Fast heuristic algorithm to find a feasible path in the graph G	99
5.5	Algorithm for routing using call statistics in the network graph G for a call between nodes a and b	102
5.6	Evaluation of CAC for various topologies.	105
5.7	Evaluating feasibility improves performance.	107
5.8	Comparison of shortest path (SPF) with max residual feasible path (MRFP) for different load.	107
5.9	Comparison of routing using call statistics (RCS) with shortest feasible path (SFP) and max residual feasible path (MRFP) for a) heavily skewed load (2% nodes are callers), b) lightly skewed load (10% nodes are callers) in Random topology.	107

LIST OF TABLES

2.1	Example contention scenarios for a three node set up: z_0 in black, z_1 in red, z_2 in green. The corresponding nodes in the testbed are 6, 4 and 8. A line between nodes shows that they do not interfere with each other.	25
5.1	Number of calls supported in a linear network	104

CHAPTER 1

INTRODUCTION

Wi-Fi is the common name for the popular wireless technology consisting of a suite of IEEE 802.11 protocols (802.11n, 802.11b, 802.11g, 802.11a, etc.). These protocols are defined for both the physical and MAC layer for wireless networking. Wi-Fi operates in the frequency bands of either 2.4GHz or 5GHz. Using stock antennas, the range of Wi-Fi connectivity is limited to a range of 100–1000 meters, which is much less than cellular wireless technologies, but higher than short-range wireless technologies like Bluetooth, Zigbee nodes, etc. The feature which gives Wi-Fi significant advantage over medium-to-long range wireless technologies is the achievable bandwidth, which can be up to a raw data rate of 54Mbps using 802.11g or 802.11a, and even as high as 108Mbps using 802.11n. The above features, along with the ability to operate in unlicensed spectrum, have made Wi-Fi the wireless technology of choice for several home and business applications, such as internet access, gaming, video streaming, etc. The popularity of Wi-Fi has driven down the price of such radio interfaces, due to which Wi-Fi can be found in several consumer devices, such as laptop computers, PDAs, mobile phones, etc.

Owing to the ubiquitous presence of cheap Wi-Fi devices, several Wi-Fi networks have been deployed in recent years. Such networks can generally be classified as infrastructure-mode network, such as wireless LANs or wireless mesh networks, or infrastructure-less network, such as ad hoc networks. A wireless LAN (WLAN) consists of Wi-Fi access points (APs) deployed on a wired backbone, and provide access to Wi-Fi clients on the last hop. WLANs are deployed on small scales in places such as coffee shops, conference halls, and homes, as well as on large scales, such as in campuses, airports, hotels, and offices. A

wireless mesh network (WMN) is similar to a WLAN except that the WiFi APs communicate wirelessly as well. There has been a recent initiative for deploying community-wide and city-wide wireless mesh networks. Unlike WLANs and WMNs, a wireless ad hoc network is a decentralized network, where instead of a client-AP interaction, each participating node becomes a part of the network. Such networks are especially useful in emergency scenarios, like military operations or natural disasters, and in cases where a quick deployment is needed.

The major technical challenge faced by all these Wi-Fi networks is the phenomenon of wireless interference. In this dissertation, we look at ways to characterize interference and develop applications using such characterization. Interference limits the aggregate capacity of Wi-Fi networks, causes starvation and collisions, and reduces the quality of user experience. A lot of research has been done to develop techniques to mitigate interference. Several methods have been proposed in the area of topology control, radio resource assignment, MAC layer enhancements, routing and even application design to reduce interference. However, while interference can be reduced, it cannot be eliminated, and this motivates our work in this dissertation. We argue that protocols and algorithms for wireless networks should be developed by incorporating an accurate model of interference. With this regard, we develop a measurement-based model to estimate the impact of interference in a wireless network. We also show the benefit and accuracy of such modeling by developing interference-aware applications which take advantage of such models.

1.1 KEY COMPONENTS

In this dissertation, we focus on suggesting ways to improve user performance in real and deployed networks. This leads to several practical considerations, which are discussed in this section. Following are some of the key components of our work.

Interference from multiple interferers – Prior research has often characterized interference as a parameter between a pair of nodes or links. This gives the representation of the

interference model as a conflict graph [37]. Due to its simplicity, the conflict graph model has been used extensively in developing algorithms for channel assignment [11, 69], power control [11], and routing. It has also been used for developing models to estimate capacity of a network. Such algorithms have often been shown to perform poorly in real deployed networks. This is because, in practice, interference is not just a pairwise entity. A link may suffer from interference from multiple interferers at once. This has been modeled as the physical interference model proposed in the seminal work of Gupta and Kumar [34]. In our work, we consider the more realistic physical interference model, thus considering the effect of multiple interferers. We build the model for capacity and the interference-aware applications by considering the physical interference model rather than a pairwise interference relationship.

Non-binary interference – In addition to using a conflict graph model, interference has often been considered as a binary value, which means that either two nodes, or links, interfere or they do not. This is based on the simplified assumption of the physical layer of a wireless network. This further simplifies protocol and algorithm development because the conflict graph considered is just an unweighted graph. In reality, due to the irregularities of the wireless medium, the interference relation between links may vary with time. Padhye, et. al. [65] ask the question that if two links interfere, then “how much” do they interfere? In our work too, we use a non-binary notion of interference, and estimate the probability that a set of nodes/links interfere with each other.

Measurement based modeling of physical layer – Many theoretical and analytical models have been proposed to predict the capacity of a wireless network. In addition to using a simplified interference model explained above, they use a simplified physical layer model. They make assumptions on the propagation environment and the interface characteristics and use various model parameters (e.g., path loss exponent) that are hard to instantiate. Since our emphasis is on realism of modeling, we do not make any

such assumptions. We rely on measurements done over a real network to characterize interference at the physical layer. Recent years have shown an increasing emphasis on measurements to evaluate wireless networks [71, 65, 43]. We use measurement based modeling to develop accurate wireless simulators, and we use measurements to instantiate our capacity model.

Real and deployed Wi-Fi networks – We target our work for real deployed Wi-Fi networks, such as wireless mesh networks, or enterprise wireless LANs. Thus, the models we develop need to be accurate, and the applications should have low overhead and should have good performance in practice. This motivates the choice of measurement-based modeling of interference. The applications we have proposed in this work are developed for such networks. We develop algorithms to support VoIP calls in wireless mesh networks, and we develop a radio resource management for enterprise WLANs.

1.2 CONTRIBUTIONS

We make the following contributions in this dissertation

- We develop a measurement-based model to predict the capacity of any given link in a 802.11-based wireless network in the presence of any given number of interferers carrying any specified amount of offered load [45]. Only $O(N)$ measurement steps are needed to gather metrics for individual links that seed the model. We provide two solution approaches – one based on direct simulation (slow, but accurate) and the other based on analytical methods (faster, but approximate).
- We address the issue of unrealistic simulations of wireless networks using a measurement-based approach. The idea is to use empirical modeling using measurement data as a mechanism to model physical layer behavior. We demonstrate the power of this approach for 802.11-based networks using ns2, a packet-level simulator by replacing the physical layer with measurements from a real testbed.

- We develop an approach to estimate the interference between nodes and links in a live 802.11 network by passive monitoring of wireless traffic using a distributed set of sniffers. We model the 802.11 protocol as a Hidden Markov Model (HMM), and use a machine learning approach to learn the state transition probabilities in this model using the observed wireless traffic traces. This in turn helps us to deduce the interference relationships. We show the effectiveness of this approach via simulations and real experiments.
- We study the problem of supporting VoIP calls in a wireless mesh network [47]. Specifically, we design solutions for call admission control (CAC) and route selection for VoIP calls. We address this issue via a measurement-based modeling effort to model mutual interference between wireless links. The modeling approach evaluates whether capacity constraints (or, required QoS metrics) will be satisfied if a new call is admitted with a given route.

1.3 OUTLINE

The rest of this dissertation is organized as follows. We first develop a measurement-based capacity model for 802.11-based networks in chapter 2. We then present a method to make ns-2 more accurate for wireless simulators using the ideas from the modeling work in chapter 3. In chapter 4, we present the machine learning approach for estimating pairwise interference in Wi-Fi networks. In chapter 5, we present the first application for the measurement-based modeling - supporting VoIP calls over wireless mesh networks. We present our conclusions in chapter 6.

CHAPTER 2

MEASUREMENT-BASED LINK CAPACITY MODEL

In this chapter, we present a practical, measurement-based model that captures the effect of interference in 802.11-based mesh networks. We model the capacity of a given link in the presence of any given number of interferers in a deployed network, carrying any specified amount of load. A link capacity model has several applications in deployed networks, such as for radio frequency resource assignment, and for wireless network management.

2.1 INTRODUCTION

Practical models for predicting the wireless link capacity are crucial to an efficient operation and deployment of wireless network. The performance of network protocols and algorithms such as QoS routing, load balancing, admission control and channel assignment can be significantly improved with an accurate model of link capacity. Capacity models are

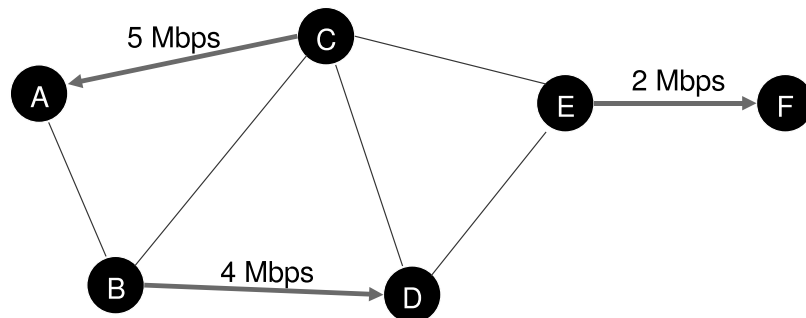


Figure 2.1: Example of problem.

also required as analysis tools to efficiently explore a gamut of network configurations and traffic load scenarios for performance evaluation.

Recently, the proliferation of 802.11 based wireless LAN and mesh networks has lead to several research efforts focussing on predicting the capacity of an *802.11-specific* wireless link [65, 71]. What makes the accurate estimation of 802.11 link capacity an inherently challenging task is that the link capacity is an ensemble effect of physical layer behavior, complex CSMA-based MAC layer interaction, and interference effect from multiple active sources.

The objective of our work is to characterize and model the impact of interference caused by active traffic from *multiple* surrounding nodes on the link capacity. For example, referring to Figure 2.1, consider a set of active links (CA, BD and EF) with specified amounts of offered traffic loads (in Mbps, for example). Our goal is to create a model that can predict the throughput capacity of any given link (e.g., BC or AB), i.e., the maximum amount of traffic (in Mbps) that the link can carry. Unlike the plethora of modeling work in existing literature [16, 32, 55] that uses purely analytical approaches, our end goal is to *estimate link capacities in a real deployed network*.

Characterizing the impact of interference: Interference impacts the sender by reducing its maximum sending rate as determined by the CSMA based 802.11 MAC layer interaction. Interference also impacts the receiver by reducing the probability of successful packet reception by causing collisions at the receiver. The specifics of the MAC protocol (e.g., random backoff) as well as implementation-specific physical layer components such as *carrier sense threshold* (i.e., what received power must be sensed to decide that the medium is busy) and *packet capture threshold* (i.e., threshold of signal-to-noise-plus-interference ratio to be able to receive a packet successfully) are other factors which affect the interference-limited capacity of a wireless link.

Existing models for single-hop [16, 55] and multi-hop [32] 802.11 networks suffer from the limitation that they are based on the assumption of idealized channel condition

where each link is lossless. They also assume that interference is ‘pairwise’ (i.e., happens between node or link pairs only) and ‘binary’ (i.e., interference is either present or absent). The popularly used protocol model of interference [34] is an example of such interference modeling. However, recent measurement studies [65, 24, 76] have shown that interference is neither pairwise or binary. The effect of multiple interferers and effect of realistic channel and interface behavior must be accounted for accurate modeling.

Measurement-based capacity model: Evidently, a model built on actual measurement of appropriate metrics can avoid the unrealistic assumptions. However, such models must be of a reasonable measurement complexity to be practical and must also be robust to potentially changing operating conditions. To that end, a recent model based on measuring just signal strengths between node pairs has been proposed by Reis *et al.* [71] to predict capacity of a link. Their model however is described for the case of single interferer and does not address the general and realistic case where the effect of simultaneous multiple interferers on link capacity must be considered. The case for multiple interferers is challenging because of the following reasons. The model has to consider *every possible* combination of interfering transmitters, because any number of them could be transmitting at a time. The model also has to capture the effect of any possible traffic load scenarios at the interferers.

Main contributions: The contributions in our work are as follows.

- i. We develop a general framework for modeling 802.11 networks (Sections 2.3 and 2.4). This presents a novel “coupled” approach, where a MAC-layer model uses a measurement-based PHY-layer model and seeds it using measurements from the target network. These measurements consist of easily measurable link metrics and can be done in $O(N)$ steps for an N node network.
- ii. We develop a tractable analytical solution approach for the model (Sections 2.5 and 2.6), that – while approximate relative to direct simulations – presents an excellent tradeoff for speed and accuracy. We show how this approach is able to estimate

the throughput capacity of a given link with any number of interferers with given traffic loads.

- iii. We provide extensive validations using direct measurements from the testbed (Section 3.5). Our validation results show, for example, that the model is able to predict the link capacity for over 90% of cases within an error less than 10% of the channel bitrate. We present validation results up to 5 interferers providing a very complete study.
- iv. We also demonstrate why modeling approaches like ours is important. Existing algorithmic and analytical work uses very simple and unrealistic capacity models for evaluation. We pick three such models and show that such models often fare very poorly in estimating link capacities in real networks relative to our approach (Section 2.8.2).

2.2 RELATED WORK

The capacity of a wireless link depends upon the quality of the link and the amount of interference. Several measurement studies [23, 9, 51, 17] have been done in literature to study the link quality in 802.11-based wireless networks. Similarly, several works have looked at the issue of interference in such networks in addition to link quality [38, 24, 65, 71, 18]. In [38], authors investigated the impact of carrier sensing. In [18], the authors developed a model for the physical layer capture. In [65], Padhye *et al.* developed a measurement-based methodology to characterize link interference in 802.11 networks. They pointed out that interference between links is not “binary” in practice unlike assumed in many analytical work that use simple graph-based conflict models. In [24], the authors showed that pairwise interference modeling is often not accurate and multiple interferers must be accounted for.

The work by Reis *et al.* [71] is the most related to our current work. They proposed a model to use the measured signal strength between pair of nodes, thus requiring only $O(N)$ experiments, to characterize link quality as well as to create a physical layer model

for deferral and collision. The model, though useful for a single interferer case, is not trivially extensible to multiple interferer scenarios. Our approach is similar to that of [71] in terms of measurement complexity. However, the main focus of our work is to develop a very general model that captures the effect of multiple interferers and any loading scenario for the interferers.

There have been several studies in characterizing and evaluating the capacity of wireless networks using analytical modeling. The capacity in this context is the network capacity for multihop flows. Prominent examples include asymptotic capacity modeling in [34] and capacity modeling using concepts from network flow maximization in [37, 56]. They all use various abstract link interference models – from pairwise models, such as protocol model, to more general models, such as physical interference model, based on SINR (signal to interference plus noise ratio). Typically, simple path loss models are assumed for RF propagation. Even with the most realistic models, instantiating such models in a real network is hard without actual measurements, as models come with several unknown parameters. The papers in this category are interested in performance bounds and typically do not use any MAC protocol model except slotted TDMA scheduling.

Finally, several papers have considered analytical modeling of 802.11 MAC protocol in multihop context to determine throughput and fairness characteristics. For example, Garetto *et al.* [32] extended Bianchi’s single hop analytical model [16] to a multi-hop 802.11 network to derive the per-flow throughput in a multi-hop network. Gao *et al.* [31] have proposed another analytical model to determine the end-to-end throughput capacity of a path carrying a flow in a multi-hop 802.11 network. However, all these works still use simple pairwise (or protocol) model of interference. The advantage of using such pairwise model is that a node that is not an interferer in isolation cannot become an interferer in conjunction with other nodes. However, in SINR-based physical interference model, this is a possibility.

Our work is complementary to many of these analytical approaches as it provides a vehicle to characterize interference modeling via real measurements. A fresh modeling

approach is needed to enable use of real measurements due to the requirement that we handle SINR-based physical interference unlike the above analytical approaches.

2.3 MODELING APPROACH

In this section, we formally present the problem we are addressing, and present our approach towards solving it.

2.3.1 PROBLEM FORMULATION

We are interested in determining the capacity of a specific link in a 802.11 network given the offered load on a set of other links. More formally, assume an N node network with all nodes in the same channel and using the same PHY-layer bit rate. Assume a subnetwork with $n + 1$ nodes consisting of a set of n transmitters, $Z = \{z_1 \dots z_n\}$, and a receiver, x . We are interested in evaluating the throughput capacity of the link from one of the transmitters (say, z_i) to the receiver x . In this case, z_i acts as sender and all nodes in $Z - \{z_i\}$ act as interferers. All other nodes in the network outside the subnetwork above are assumed silent. We will use the notation $C_{\text{receiver}}^{\text{sender}}(\text{set of interferers})$ to designate throughput capacity of the link. Thus, we are interested in determining the throughput capacity, $C_x^{z_i}(Z - \{z_i\})$, of the link z_i to x , given the offered load l_i on each transmitter in Z .

The capacity of an 802.11 wireless link depends on the following factors – (i) channel quality that determines the bit error rate for a given PHY-layer bit rate (governed by modulation used); this translates to packet loss rate from the point of view of an upper layer protocol; (ii) interference from other transmissions in the network that influences how the 802.11 MAC protocol behaves at the sender side and whether packet collisions occur at the receiver side. Our goal is to develop a measurement based model that captures the “time averaged” behavior of the physical and MAC layers in 802.11, and thereby predicts the throughput capacity of a wireless link in presence of any number of interferers and with

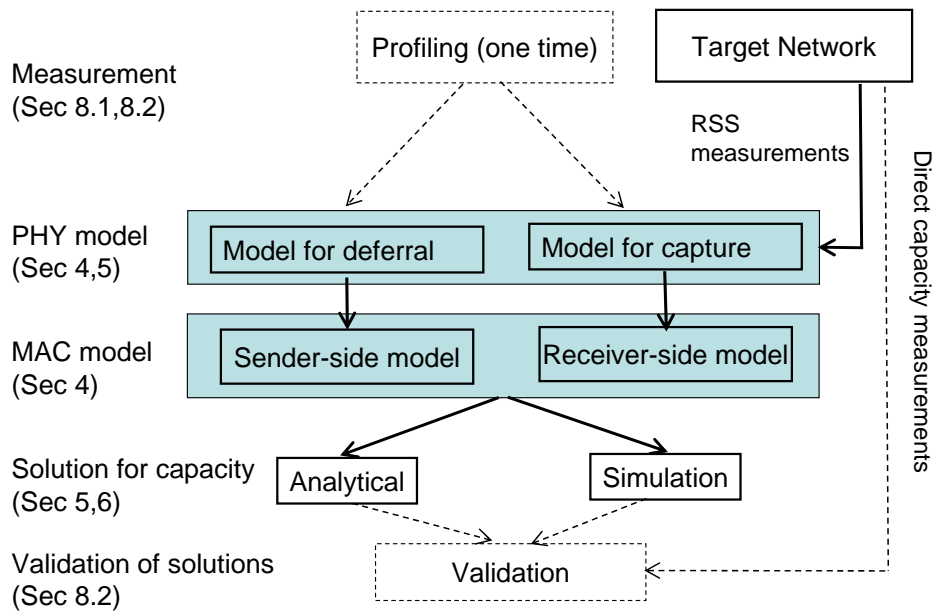


Figure 2.2: Overview of the modeling approach.

any given traffic load matrix. Note that given the time varying nature of wireless channels, “instantaneous” behaviors are very hard to model using measurement based approaches.

2.3.2 OVERVIEW OF APPROACH

A high level block diagram of our approach is shown in Figure 2.2 with pointers to sections where different parts are described in this chapter. The centerpiece is a MAC-layer model of 802.11 that is fed by a PHY-layer model. The PHY layer model models two behaviors that MAC depends on: (i) *deferral*, whether enough interference power is received to indicate carrier busy, (ii) *packet capture*, whether the SINR is high enough such that packet is received correctly. These dependencies are modeled via measurements in a one-time profiling experiment. The profiling is done for each interface card model or type, and can be reused.

These models are seeded by link-wise measurement of RSS (received signal strength) values in the target wireless LAN or mesh network. The RSS values can be measured by having each node taking turn and sending a set of broadcast packets. For a given broad-

casting sender, rest of the nodes record RSS. For an N node network, the measurement requires $O(N)$ measurement steps and provides the metrics for all the $N(N - 1)$ links. This seeding now makes the MAC-layer model amenable to numeric solution. *The solution evaluates how long the model stays in appropriate states that contribute to capacity.* We propose two solution approaches - (a) analytical method and (b) simulation. The analytical method (Section 2.5) translates the model to a set of coupled equations that are solved using numerical methods. The method uses certain (reasonable) assumptions to make it analytically tractable, which also makes the solutions approximate. Simulation, on the other hand, accurately follows the MAC-layer model (described in Section 2.4), but results in much slower computation. We will demonstrate this further in the evaluation section in Section 3.5.

We validate the entire approach by comparing the link capacities estimated via this modeling approach with direct measurements on the target mesh network testbed. Note that the dotted blocks in Figure 2.2 are not needed for capacity evaluation in a deployed network. The profiling is to be done one time only and should be available as a library for different interface card models. The validation step is also not necessary. It is used only to demonstrate the power of our approach in this chapter and also for comparison with other approaches of estimating link capacities.

2.4 MODELING 802.11 BEHAVIOR

We begin by stating an assumption that we have made in most of the chapter for modeling convenience. We assume that 802.11 is using only broadcasts, i.e., implementing unicast using broadcasts. Broadcast does not have link-layer ACKs, and exponential backoffs. This simplifies the model to some extent. It has also been shown that interference between links carrying unicast traffic can be well predicted by the amount of interference computed when they carry broadcast traffic [65, 71]. Note that we are merely using this simplification for

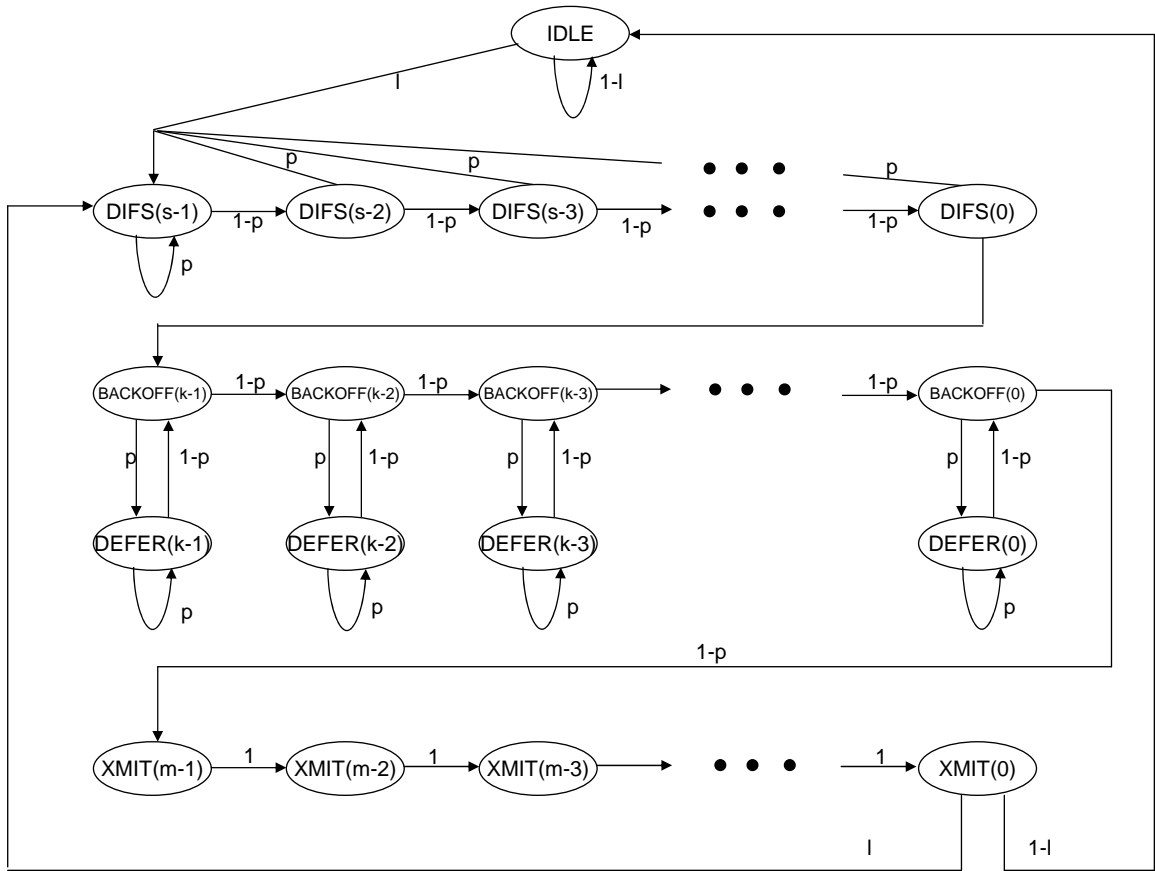


Figure 2.3: State transition diagram for 802.11 on the sender-side.

brevity. The modeling approach is general and can be extended to unicasts, as will be shown in Section 2.7.2.

We present the behavior of 802.11 MAC protocol from the point of view of a single node as a discrete time Markov chain (see Figure 2.3). For this we discretize time, albeit somewhat artificially, into slots. These slots are different from 802.11 slots. The size of the slots is chosen such that they are small enough that the protocol state does not change within a slot, and the duration of any protocol state has only integer number of slots.

There are five possible states – IDLE, DIFS, BACKOFF, DEFER or XMIT. Each of these states consists of many sub-states denoting the number of slots they span. We need multiple sub-states because the sub-states are not independent of each other. When the node is not attempting any transmission, it is in the IDLE state. When in IDLE state, in every

slot the node checks if it has any packet to transmit. This depends on the offered load l_i for the node z_i , and represents the probability to begin packet transmission. When traffic is backlogged, a node never enters the IDLE state. When, the node has a packet to transmit, it moves to the DIFS state (this is an inter-frame spacing defined in the protocol standard), which has s sub-states, where s is the number of slots a node has to be in DIFS state. If the node senses the channel busy during this period, it goes back to the beginning of DIFS, i.e., $DIFS(s - 1)$. The probability of channel being busy is given as p , also called the *probability of deferral*. This probability is a PHY-layer aspect and depends on the aggregate power from other nodes reaching this node. This in turn depends on the current state of the other nodes.

After successful completion of the DIFS period, i.e., upon reaching $DIFS(0)$, the node chooses a random BACKOFF period, spanning k slots, where $0 < k < CW_{min}$, and moves to the sub-state $BACKOFF(k - 1)$. It then counts down the BACKOFF timer, and thus progressing from one BACKOFF sub-state to the other, but only if the channel is sensed idle. If the channel is sensed busy (again with probability p), the node goes into the DEFER state, where it freezes the BACKOFF timer. It remains in the DEFER state as long as the channel is busy. The node goes back to the BACKOFF state with the probability of the channel being idle (probability $1 - p$). Having counted down the BACKOFF timer to 0, the node starts transmitting the packet. This brings it to the XMIT state. Assume that the XMIT state stays for m slots depending on the PHY-layer bit rate and packet size. After completing the packet transmission, the node goes back to IDLE state if there is no other packet to transmit, or prepares for the next transmission with another DIFS.

One key approximation made in this model is that the deferral probability p is assumed to be constant during the evolution of the Markov process.¹ This probability depends on the activity of the other nodes. Thus, the state transitions of other nodes are closely coupled. When we solve this model using a direct simulation (i.e., simulating the Markov chain)

¹Note similar approximations are used in popular models of 802.11 *albeit* in a different context, e.g., in [16].

we do not make such constant p assumption and use the value p as computed at that slot. When we solve the chain using the analytical approach in the following section, p is the “average” deferral probability. This averaging works due to an inherent approximation used in the analytical solution approach to be described momentarily.

So far we have described only the transmitter side. On the receive side, the model is simpler. A node not in XMIT state can receive a complete packet slot by slot, assuming it receives it error-free in each slot. The probability of error-free reception of a complete packet (*packet capture probability*) depends on the bit-error rate (BER) in the PHY-layer which in turn depends on the SINR (signal to interference plus noise ratio). Ignoring error correction coding, the probability of packet capture is $(1 - \text{BER})^b$, where b is the packet size in bits. Thus, packet capture probability depends on SINR.

Both probabilities for deferral and packet capture are functions of one or more powers (signal, interference and noise). They are input to the model. We will determine these functions via profiling experiments and seed them by power measurements in the target network.

2.5 ANALYTICAL APPROACH

Due to the coupling of the Markov chains of individual nodes as mentioned before, solving an equivalent Markov chain for the network as a whole is computationally hard. This is because of a state-space explosion, as all possible combinations of states for all nodes can be a potential state in the combined Markov chain. Direct simulation of the Markov chain is of course viable, and we will indeed use simulation as our one solution approach. However, as we will see later in our evaluation, simulations are slow. In this section, we develop an alternative solution approach using analytical modeling.

The analytical approach makes an approximation that the current state of the process does not depend on the previous state. This is similar to the approximation made in [32] for modeling tractability. With this approximation, the process can move to any of the above

five states (ignoring sub-states for now) based on a constant probability at the end of a slot. These probabilities depend only on the average behavior of network nodes. Much of the work in the modeling here is formulating these probabilities. Once formulated, one can write up the steady state equations, one for each of the n transmitters, and then solve these equations to derive the fraction of time a node is in the XMIT state, thus giving the transmission capacity of this node.

On the receiver side, the approach is similar. Instead of bit-error rate, *packet capture probability* is used directly. This again depends on the activities of other nodes. Any receiver x in a slot receives correctly a packet on the air (only one slot worth) from a designated sender z_i with this probability. This contributes to the throughput capacity of the link from z_i to x .

Going forward, we start by assuming a *saturated traffic* regime. This means that all transmitters are always backlogged. This saturated traffic assumption is useful as it eliminates traffic load from the model and eliminates the IDLE state. We will later show in Section 2.7.1 that the analytical approach is easily amenable to consideration of non-saturated traffic.

2.5.1 BASELINE NOTATIONS

Consider an observation interval of Γ slots, where $\Gamma \rightarrow \infty$. In each slot, a subset of the n transmitters in $Z = \{z_1, \dots, z_n\}$ may attempt transmission. The set Z does not change during the duration of Γ slots. Let us first define the following notations:

- I_i is the set of time slots in which node z_i is idle. This is when node z_i is in the IDLE, DIFS or BACKOFF states.
- D_i is the set of time slots in which node z_i defers because it can sense the transmission of other nodes. This is the period where z_i freezes its backoff timer and goes into the DEFER state.
- T_i is the set of time slots in which node z_i transmits, denoted by the XMIT state.

- $i_i = |I_i|/|\Gamma|$, is the fraction of time node z_i is idle.
- $d_i = |D_i|/|\Gamma|$, is the fraction of time node z_i defers.
- $c_i = |T_i|/|\Gamma|$, is the fraction of time node z_i transmits. So, c_i is the normalized transmission capacity of node z_i .
- c_Y , where $Y \subseteq Z$, is the fraction of time all nodes in set Y transmit. Thus,

$$c_Y = \left| \bigcap_{z_i \in Y} T_i \right| / |\Gamma|. \quad (2.1)$$

- t_Y , where $Y \subseteq Z$, is the fraction of time when all nodes in Y transmit, while none of the other nodes (in $Z - Y$) transmit. Thus,

$$t_Y = \left| \bigcap_{z_i \in Y} T_i - \bigcup_{z_j \in Z - Y} T_j \right| / |\Gamma|. \quad (2.2)$$

If Y consists of a single node, say z_i , we abuse the notation slightly to represent it as t_i to represent $t_{\{z_i\}}$. t_i is thus the fraction of time node z_i transmits, and no other node in Z transmits.

- p_i^Y , where $Y \subseteq Z - \{z_i\}$, is the conditional probability that when all nodes in Y transmit in a slot, z_i defers its transmission because it senses the channel to be busy. When Y has just one node, say z_j , then we again abuse the notation to represent it as p_i^j .

Interference affects link capacity by limiting the transmission rate at the sender side and causing packet collisions at the receiver side. We denote these aspects as “sender-side interference” and “receiver-side interference” respectively and model them separately.

2.5.2 SENDER-SIDE INTERFERENCE

To compute the impact of sender-side interference, we determine the transmission capacity (c_i) of each node in Z . Using the notations defined above, I_i , D_i and T_i are disjoint sets.

Also, every slot is at least in one of these three sets for every node. Thus, $I_i \cup D_i \cup T_i = \Gamma$. This implies that

$$i_i + d_i + c_i = 1. \quad (2.3)$$

In the saturated traffic scenario, a node is idle only during DIFS or backoff period. This happens for every packet transmission. DIFS is constant; however the backoff period is random, uniformly chosen between 0 and CW_{min} slots of, say, size σ for broadcast packets.² Knowledge of packet size and channel bit rate can now provide an expression for the ratio (α) of the idle and transmit times, on average:

$$\alpha = \frac{i_i}{c_i} = \frac{DIFS + \frac{1}{2}CW_{min}\sigma}{(P + H)/W}. \quad (2.4)$$

Here, P is the packet payload size, H is the size of the headers, W is the channel bit rate. Using the standard values of DIFS, slot sizes, CW_{min} and various headers, we determine α at the lowest bit rate for 802.11b (1 Mbps) for 1400 byte packet payloads. This comes to 0.03 for 802.11b.

Equation 2.3 can now be re-written as

$$(1 + \alpha)c_i + d_i = 1. \quad (2.5)$$

In the above expression, d_i is the fraction of time slots node z_i defers due to the transmission of other nodes. In each slot, there can be a set of nodes (say, Y) that transmit. For each slot the conditional probability that z_i defers to Y , given that all nodes in Y are transmitting is p_i^Y . We can now add up the deferral probabilities in each slot for all possible combinations of Y to obtain d_i . Note that t_Y is the fraction of time slots in which all nodes in Y transmit. Thus,

$$d_i = \sum_{Y \in \mathcal{P}(Z - \{z_i\})} p_i^Y t_Y, \quad (2.6)$$

where $\mathcal{P}(S)$ is the power set of set S . This leaves us with p_i^Y and t_Y to be determined for each possible Y , such that $Y \subseteq Z - \{z_i\}$.

²Note that here there is no exponential backoff as there is no retransmission.

DETERMINING p_i^Y

Recall that p_i^Y is the conditional probability that z_i defers when all nodes in Y are transmitting. Here, we need to model the MAC protocol's interaction with the physical layer, as this probability should depend on the aggregate signal powers received at z_i from all nodes in Y . To make further progress, the relationship between the deferral probability and received signal strengths must be modeled. Since this is intimately related to the actual radio interface used, we use a measurement driven strategy here.

The first step is to create an empirical relationship for the probability of deferral between two nodes based on received signal strengths. We express this relationship as a function $f(\cdot)$, such that $p_i^j = f(rss_i^j)$, where rss_i^j denotes the average of measured signal strength value of packets transmitted from z_j and received at z_i . We determine function $f(\cdot)$ from a prior profiling study. Note that this function models interface properties rather than wireless propagation in an actual deployment. Thus, such prior profiling study is possible. However, *in our experience, individual cards do not need to be profiled in this fashion, only card types or card models need to be profiled.* These profiles can be reused from a library for different modeling applications. This is in contrast to a similar profiling approach used in [71], where individual cards are profiled. Note that our approach is general and is not restricted to a homogenous system using identical cards. However, for brevity, our experimental results show results from a homogeneous deployment. The profiling methodology to determine $f(\cdot)$ will be discussed in Section 3.5.

Once the function $f(\cdot)$ describing the relationship between the deferral probability and signal strengths is determined, p_i^Y can be expressed as in the following.

$$p_i^Y = f\left(\sum_{z_j \in Y} rss_i^j\right). \quad (2.7)$$

This is true since the deferral only depends on the aggregate signal strengths. Now, if the measurements of the pairwise rss_i^j values in the deployed network are available, p_i^Y can

be determined for any Y . Note that measuring all rss_i^j values requires $O(N)$ measurement steps.

DETERMINING t_Y

Recall from equation 2.2 that t_Y is the fraction of time all nodes in set Y transmit, and all nodes in the complement set $Z - Y$ remain silent. c_Y on the other hand is the fraction of time nodes in Y transmit, but nodes in set $Z - Y$ may or may not transmit. We determine t_Y in terms of c_Y using equations 2.1 and 2.2. From these equations,

$$t_Y = c_Y - \left| \left(\bigcap_{z_i \in Y} T_i \right) \cap \left(\bigcup_{z_j \in Z-Y} T_j \right) \right| / |\Gamma|.$$

The second term on the right hand side can be expanded using the principle of inclusion and exclusion of set theory, which after evaluation reduces to the following –

$$t_Y = \sum_{X \in \mathcal{P}(Z-Y)} (-1)^{|X|} c_{Y \cup X}, \quad (2.8)$$

where $\mathcal{P}(S)$ denotes the power set of S .

We still need to determine c_Y , which is the fraction of time nodes in Y transmit together. Nodes in Y transmit together when every node in Y does not defer for every other node in Y . Thus, c_Y can be expressed as,

$$c_Y = \prod_{z_i \in Y} (1 - p_i^{Y-z_i}) c_i. \quad (2.9)$$

Equations 2.6, 3.2, 2.8 and 2.9 can be used to obtain d_i and then used in equation 2.5 to write an equation consisting of c_i 's and rss_i^j as the only unknowns. rss values come from the measurements, leaving only c_i 's as unknowns. Now, for each value of the subscript i (i.e., a transmitter) one such equation is obtained, giving n equations for n transmitters. We solve these equations to derive the normalized transmit capacity c_i for each transmitter.

2.5.3 RECEIVER-SIDE INTERFERENCE

So far, we have modeled transmission capacity of the transmitter. We now need to model receiver-side interference to determine how much of the transmission capacity actually

translates into throughput. Receiver-side interference causes collisions. Thus, if the sender and multiple interferers transmit concurrently, we need to model the probability of packet capture at the receiver. As discussed before, this is done by deriving a relationship between the capture probability and the SINR. This is done in the same fashion as in the case of deferral probabilities in the previous section. Exactly as before, we relate packet capture probabilities to SINR via a function $g(\cdot)$ that is profiled via independent measurements. The profiling methodology to determine $g(\cdot)$ will be discussed in Section 3.5.

Define delivery ratio dr_i^j from z_j to z_i as the fraction of packets received by z_i that are transmitted by z_j in the absence of any other interfering transmitter. Let us define $dr_i^j(Y)$ as the delivery ratio from z_j to z_i in presence of the set of interferers Y . Our first task is to model dr_i^j as $dr_i^j = g(rss_i^j/\text{noise})$. This simply relates packet capture probability to SNR, the ratio of the received signal strength and noise. Here rss_i^j denotes the average signal strength of packets received from z_j to z_i in absence of interference. We have observed that the function $g(\cdot)$ does not change even if we consider multiple interferers for a link, and the signal strengths of all interferers can be summed up to calculate SINR. This is in contrast to the results of [76] for the Mica2 motes with CC1000 radios.

Once the function $g(\cdot)$ has been modeled, $dr_i^j(Y)$ can be expressed as follows:

$$dr_i^j(Y) = g(SINR_i^j(Y)), \quad (2.10)$$

where,

$$SINR_i^j(Y) = \frac{rss_i^j}{\sum_{k \in Y} rss_i^k + \text{noise}}. \quad (2.11)$$

As in the case of equation 3.2, the above equation also requires only pairwise measured rss values in the deployed network.

2.5.4 CAPACITY OF LINK

Now, we combine the sender and receiver-side interferences to determine the capacity of the link. Let us choose z_i as the designated sender from the set Z , and let x be the receiver. All

the other transmitters are interferers for this link. Assume that only a subset Y of the set of interferers $Z - \{z_i\}$ is active in a slot and the others are silent (due to deferral or idleness). By definition, t_Y is the fraction of slots with this property. $t_{\{z_i\} \cup Y}$ is thus the fraction of time the sender z_i transmits along with some subset of the interferers. This models the packets that are transmitted from the sender notwithstanding sender-side interference. This quantity multiplied by $dr_x^i(Y)$ models how many of them are captured at the receiver x notwithstanding receiver-side interference.

Thus, the overall link capacity (in bits per sec) from the sender z_i to receiver x in the presence of a set of interferers $Z - \{z_i\}$ is given by,

$$C_x^{z_i}(Z - \{z_i\}) = \frac{P}{P + H} \times W \times \sum_{Y \in \mathcal{P}(Z - \{z_i\})} dr_x^Y \times t_{\{z_i\} \cup Y}. \quad (2.12)$$

The first term models the header overhead and the second term specifies the channel bit rate. The third term models the above argument. Consideration of the power set is necessary as any set of interferers can be active in a slot. The summation over all these possibilities works as they are all mutually exclusive.

In Section 2.5.2 we indicated how to compute c_i 's. t_Y 's can be determined using equations 2.8 and 2.9. dr 's come from the measurement-based modeling directly. Thus, the link capacity C can be determined using equation 2.12. The approach of solving equations is described in the following section.

2.6 SOLVING EQUATIONS

The first and hardest step in the solution is solving for the sender-side model as described at the end of Section 2.5.2. This generates a set of non-linear equations involving c_i 's as the only unknowns, which need to be solved to determine numeric values for c_i 's. This is the computationally intensive part of the model solution. Once c_i 's are determined, the rest of the steps needed to determine the capacity $C_x^{z_i}(Z - \{z_i\})$ is relatively straightforward, as they need only value substitutions. Thus, for brevity, we only discuss the sender-side solution (determining c_i 's).

There are n equations, one for each transmitter z_i . The number of terms in each equation can be exponential in n involving all possible combinations of c_i 's in a product form, i.e., terms like c_i , $c_i c_j$, $c_i c_j c_k$, etc., going upto $c_1 c_2 \dots c_n$. The equations are solved using numerical methods. More on this is in Section 2.8.3. In our validation work, we have often had opportunities to simplify the equations that reduces the number of terms involved and thus the computation time. Two types of simplifications are possible (see below). This is easily understood by looking at equation 2.6.

- $p_i^Y = 0$: This means that the node z_i does not defer for the nodes in Y . In such cases, the term $p_i^Y t_Y$ becomes 0.
- $p_j^k = 1$ and $p_k^j = 1$: This means that node z_k and z_j can hear each other perfectly, and their transmissions never overlap each other ($t_{\{z_j, z_k\}} = 0$). In such a case, the term $p_i^{\{z_j, z_k\}} t_{\{z_j, z_k\}}$ becomes 0.

Also, these terms do not need to be perfectly 0 or 1 to be eliminated. Terms close enough to 0 or 1 can be approximated as 0 or 1. In our testbed, we found many such opportunities to reduce the number of terms in each equation.

2.6.1 EXAMPLES: TWO AND THREE TRANSMITTERS

To get a better understanding about these equations, we will consider two sets of examples below – one with 2 transmitters (z_1 and z_2), and other with 3 transmitters (z_1 , z_2 and z_3). For notational convenience, we will write $t_{\{z_i, z_j\}}$ as $t_{i,j}$. Similarly, we write $p_i^{\{z_j, z_k\}}$ as $p_i^{j,k}$.

The equations for two transmitters case are:

$$\begin{aligned} (1 + \alpha)c_1 + p_1^2 c_2 &= 1 \\ (1 + \alpha)c_2 + p_2^1 c_1 &= 1 \end{aligned} \tag{2.13}$$

The solutions are

$$c_1 = \frac{(1 + \alpha) - p_1^2}{(1 + \alpha)^2 - p_1^2 p_2^1}, c_2 = \frac{(1 + \alpha) - p_2^1}{(1 + \alpha)^2 - p_2^1 p_1^2}.$$

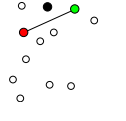
Nodes in testbed	Equations	Predicted c_0, c_1, c_2	Delivery-based c_0, c_1, c_2	Distance-based c_0, c_1, c_2	Measured c_0, c_1, c_2
	$1.03c_0 + c_1 + c_2 - c_1c_2 = 1$ $1.03c_1 + c_0 = 1$ $1.03c_2 + c_0 = 1$	0.01 0.97 0.97	0.33 0.5 0.5	0.33 0.5 0.5	0.09 0.9 0.9

Table 2.1: Example contention scenarios for a three node set up: z_0 in black, z_1 in red, z_2 in green. The corresponding nodes in the testbed are 6, 4 and 8. A line between nodes shows that they do not interfere with each other.

Let us consider two special cases, one in which both nodes can hear each other perfectly ($p_1^2 = p_2^1 = 1$), and another, where neither can hear the other other ($p_1^2 = p_2^1 = 0$). The solution for 802.11b ($\alpha = 0.03$) is ($c_1 = 0.49, c_2 = 0.49$) and ($c_1 = 0.97, c_2 = 0.97$) respectively.

The three transmitter case is a little more involved. As an example, the equation for a single node (z_1) is

$$(1 + \alpha)c_1 + p_1^2 t_2 + p_1^3 t_3 + p_1^{2,3} t_{2,3} = 1, \quad (2.14)$$

where

$$t_2 = c_2 - c_{2,3}, \quad t_3 = c_3 - c_{2,3}, \quad t_{2,3} = c_{2,3},$$

$$c_{2,3} = (1 - p_2^3)(1 - p_3^2)c_2c_3, \quad p_1^{2,3} = f(rss_1^2 + rss_1^3).$$

To show how our model can detect the starvation caused due to the classical ‘flow-in-the-middle’ problem [32], we present the set of equations for a similar scenario from our testbed shown in Figure 3.2. Nodes 4, 6 and 8 form a scenario where node 6 can hear both nodes 4 and 8 perfectly, which are hidden from each other. This leads to the starvation of node 6, which loses out of transmission opportunities because it has to defer for both nodes 4 and 8. Simple capacity models based on distance or delivery, as described later in Section 2.8.2 are unable to predict this, while our model predicts the starvation of node 6.

The equations, their solutions and the measured capacity values for this case are shown in Table 2.1.

2.7 EXTENSIONS

Now, we will pay our attention to the two simplifying assumptions we have used so far. The first is related to the assumption of saturated traffic in the analytical solution approach. The second is the consideration of broadcast transmission only. We will now discuss how to handle these issues.

2.7.1 NON-BACKLOGGED INTERFERERS

To model non-saturated conditions, we will need to account for the IDLE state in Figure 2.2. Assume first that there are only two transmitters z_0 and z_1 . Assume that z_1 , the interferer, is not backlogged and has packets to transmit only l fraction of times. In other words, the normalized offered load at z_1 is l . Let us now represent the capacity of link z_0 to x in presence of such an unsaturated interferer as $C_x^{z_0}(z_1, l)$, with a little abuse of notation.³ We show how $C_x^{z_0}(z_1, l)$ depends on $C_x^{z_0}(z_1)$, the capacity in presence of a saturated interferer.

If l is greater than c_1 , z_1 's transmission capacity, the case is similar to the saturated interferer because node z must be always backlogged to satisfy its offered load. If l is less than c_1 , node z_1 's demand is satisfied, and z_0 can use the silent period of z_1 for transmitting packets. The fraction l/c_1 , thus, can be seen as the fraction of time the two transmitters behave as if they are in backlogged conditions. The remaining fraction of time, $1 - l/c_1$ is monopolized by z_0 's transmissions. Thus,

$$C_x^{z_0}(z_1, l) = \begin{cases} \left[\left(1 - \frac{l}{c_1}\right) C_x^{z_0}(\Phi) \right] + \left[\frac{l}{c_1} C_x^{z_0}(z_1) \right], & l < c_1 \\ C_x^{z_0}(z_1), & \text{otherwise.} \end{cases} \quad (2.15)$$

We can extend this approach for solving for the non-backlogged interferer to multiple such interferers. Assume, node x is the receiver, node z_0 is the sender, and a set of nodes

³ $C_x^{z_0}(\{z_1\}, 1.0)$ is written as $C_x^{z_0}(z_1)$.

$Z = \{z_1, \dots, z_n\}$ are the interfering nodes. Assume, the nodes in set Z have normalized offered loads $L = \{l_1, \dots, l_n\}$, respectively. Let us consider the interferer, z_i , with the smallest load, such that its demand can be satisfied. The fraction l_i/c_i can be seen as the fraction of time when all the nodes have backlogged traffic. Thus,

$$C_x^{z_0}(Z, L) = \left[\left(1 - \frac{l_i}{c_i}\right) \times C_x^{z_0}(Z - \{z_i\}, L') \right] + \left[\frac{l_i}{c_i} \times C_x^{z_0}(Z) \right]. \quad (2.16)$$

where L' is the residual offered load vector after the load in the fraction of time with saturated conditions with z_i has been satisfied. For z_j , current residual load is l'_j .

$$l'_j = l_j - \frac{l_i}{c_i} \times c_j. \quad (2.17)$$

The above equation can be further reduced by considering the next node with the smallest demand and so on, until we are left with backlogged nodes only.

2.7.2 MODELING UNICAST

Unicast transmission in 802.11 provides reliability using retransmissions when the packet is not delivered successfully, and an ACK is not received from the receiver. When retransmitting a packet, the backoff window is doubled. This is done repeatedly until the ACK is received, or the retry limit has been exceeded. The broadcast model presented in Section 2.4 and Figure 2.3 can be easily extended to handle ACKs and increased backoffs for each retransmission. This would require an extra transition from the XMIT(0) state to the BACKOFF(k') state with a probability equal to collision probability (modeled by $1 - dr$) where k' is the new backoff window, $0 < k' < 2CW_{min}$.

Let us consider a scenario with sender z_0 , receiver x , and interferers Z as before. The analytical approach presented in Section 2.5 needs following modifications to solve the unicast model.

- *Idle time computation* : Due to retransmissions, and multiple backoffs for the transmission of a single packet, the ratio between normalized idle times (i_i) and transmit

times (c_i) does not remain a constant. We can compute idle time by considering all possible subsets Y of the interferer set Z and the collision probability with each of these subsets, when they are active. For each Y , the backoff time evolution is a geometric process with the collision probability as parameter. Thus,

$$i_i = \sum_{Y \in \mathcal{P}(Z)} \frac{DIFS + SIFS + bo(Y)}{(P + H)/W} t_{\{z_0\} \cup Y}, \quad (2.18)$$

where, $bo(Y)$ is the average backoff time spent for transmitting a packet (including retransmissions) from z_0 to x when a subset of interferers Y is active:

$$bo(Y) = \sum_{k=0}^m (1 - dr_x^Y)^k 2^{k-1} CW_{min} \sigma. \quad (2.19)$$

Here, m denotes the retransmission limit for a packet.

- *Consideration of ACK* : We keep equation 2.3 unchanged by considering ACK transmissions as part of a sender's transmission. Thus, in any XMIT slot, a node may be transmitting data, or receiving ACK. ACK packets are small and their impact in causing interference is also small relative to data packets. Also, ACK is transmitted only once per successful packet transmission, while the packet may be retransmitted. Thus, for a single packet, the proportion of time slots occupied by ACK is very small compared to the time slots occupied by data. In the XMIT slots, ACK may impact the deferral probability, and the probability of collision by causing DATA-ACK, or ACK-ACK collisions. Both these probabilities may still be modeled by attributing a small (appropriately computed based on sizes) probability to a XMIT slot being occupied by an ACK transmission. Another simplified model could simply ignore the effect of ACK transmissions in causing interference.

With the above modifications, the link capacity can be computed as in the case of broadcast following the same steps. Note that once the slots of the sender's transmission has been identified, the unicast capacity for those slots is identical to the broadcast capacity. This is because if the probability of packet capture is fixed, it does not matter whether a packet

is being transmitted or retransmitted. The throughput of the link will be the same in both cases, as throughput only depends on the number of unique packets successfully received.

Summarizing, modeling unicast requires modifying the model for idle time computation, and considering the probability of collision and deferral for ACK packets. Even though the inclusion of these in the model makes the model more accurate, it adds an extra complexity for the analytical and simulation-based approaches. The impact of these factors are small because ACK packets are small in general, and the extra idle time is much less than the packet transmission time for large packets. Also, as we argued above, retransmissions do not impact the capacity computation for a link except for the extra idle time. Given this, it is worth debating whether there is much benefit at all from modeling the more complex unicast. It has been shown before in [65, 71] that the interference between unicast transmissions can be well estimated by estimating the interference between broadcast transmissions. We also observed similar behavior in our testbed (not reported here).

2.8 EXPERIMENTAL PROCEDURE

Our experimental testbed consists of 12 Dell Latitude D520 laptops running Linux 2.6.15 kernel. The testbed is located in one floor of a modern office-cum-lab environment. See Figure 3.2 for a network diagram. Each laptop uses a DLink AirPremier DWL-AG660 802.11a/b/g PC card with Atheros AR5212 chipset. The Madwifi driver, Version 0.9.6 [4] is used. The cards are configured in ad hoc mode when used as transmitter, and in monitor mode, when used as receiver. Thus, measurements of dr and rss values are done in the monitor mode. rss is in the prism monitoring header which is obtained whenever a packet is captured when the card is in monitor mode. The value reported by Atheros cards is the gain dB relative to the noise floor. In particular, the card reports the value $10 \log_{10}(\frac{S+I}{N})$, where S is the received signal power and I is the aggregate interference power, N is a fixed noise floor (fixed at -95dBm). According to the above representation, any external interference

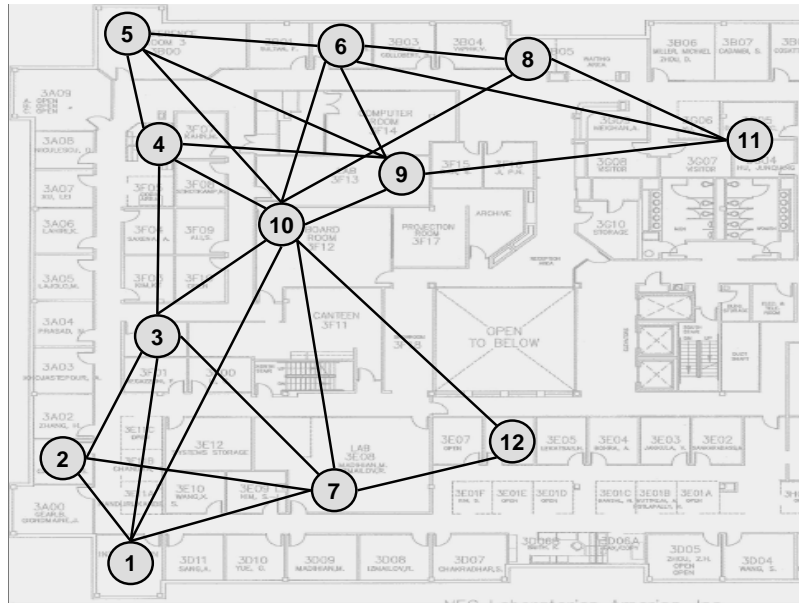


Figure 2.4: Locations of the nodes on the floor map and links with more than 90% delivery ratio. Width of the map is 60m.

will influence the measured r_{ss} value between two links.⁴ To alleviate this problem, we have done all our experiments in the night in a relatively ‘quiet’ environment so that interference I from other 802.11 networks could be considered zero. Thus the measured r_{ss} is simply a dB gain over a fixed noise floor and can be easily converted to power (mW or dBm) to use in the formulation in Section 2.5. Instead of inventing new notations, we will be using the term r_{ss} everywhere. In the experiments it is in dB, in the analysis it is in dBm or mW.

All experiments reported here are done for 802.11b. We also did a similar set of validations for 802.11a and had very similar experience. We choose to present 802.11b results here as it gives longer range links and has a rich set of interferences in our testbed. All experiments are done at the lowest phy-layer rate (1 Mbps) and with large (1400 bytes) packet sizes. We have verified that profile for one packet size can be used for other packet sizes. Profiles also seem quite independent of the choice of channels. However, profiling

⁴In other cards, for example, Prism2-based [8], it may be possible to measure the external interference as noise.

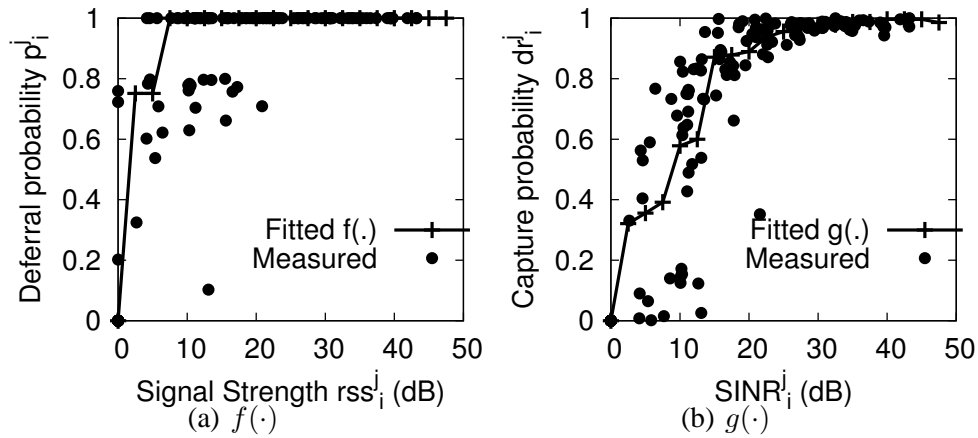


Figure 2.5: Profile for functions $f(\cdot)$, probability of deferral, and $g(\cdot)$, probability of capture.

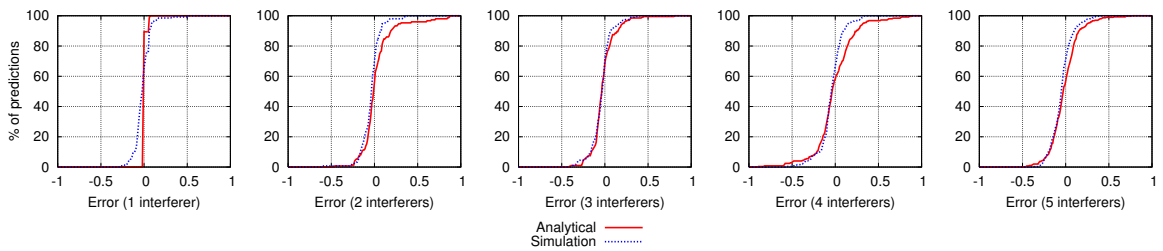


Figure 2.6: CDF of error between the estimated and measured transmission capacity of senders, c_i for node z_i .

needs to be done for each possible data rate. Needless to say, different card models must be profiled separately.

2.8.1 PROFILING EXPERIMENTS

We do a set of measurements to create the profiles $f(\cdot)$ and $g(\cdot)$, which form the inputs to the 802.11 MAC model. Recall that function $f(\cdot)$ models the probability of deferral in terms of the received signal strength (equation 3.2), while function $g(\cdot)$ models the probability of packet capture in terms of SINR on the link (equation 3.3). To create the profiles, we use two laptops (as described above), say z_i and z_j , and place them at different random

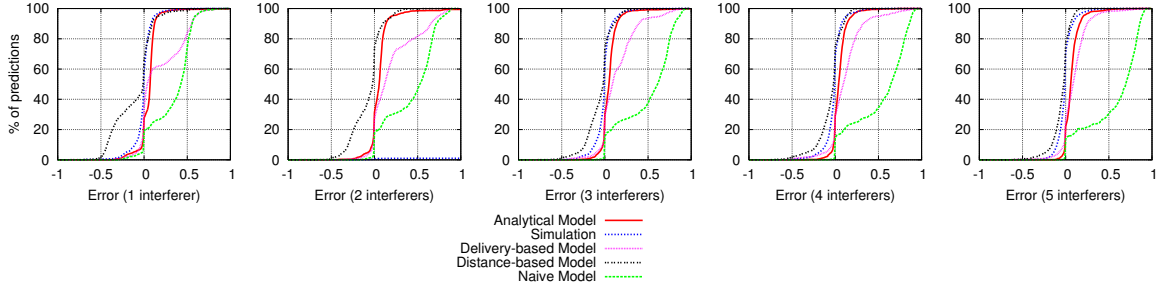


Figure 2.7: CDF of error between the estimated and measured throughput capacity on links, $C_x^{z_i}(Z - z_i)$ for link from z_i to x .

locations to create a large number of samples of average rss_i^j and rss_j^i values⁵ and then relate these samples to measured values of p_i^j and p_j^i . To do this, c_i and c_j are measured when both of the nodes have saturation UDP broadcast traffic, and then equation 2.13 is used to compute $p_i^j = \frac{1-(1+\alpha)c_i}{c_j}$, $p_j^i = \frac{1-(1+\alpha)c_j}{c_i}$. Each $\langle p, rss \rangle$ pair thus obtained is plotted in Figure 2.5(a). A large number of such points are obtained by repeating the process for different random locations of z_i and z_j , which gives different samples of link quality and sender-side interference.

To create the profile for $g(\cdot)$, we use similar random experiments using two nodes. (In fact both these experiments are done together to save effort). In this case, the $SINR_i^j$ is determined from equation 3.4 as (rss_i^j/noise) , as there is no interference. The delivery ratio dr_i^j is directly measured. As before, a plot is created (Figure 2.5(b)) relating capture probability and SINR.

The figures show the measured values as scatterplot and also the fitted curves. The curves for the desired functions are fitted using a linear interpolation of average values in buckets of 2dB each. An interesting observation in the graphs is that the profile for packet capture probability is shifted to the right when compared to the profile for deferral

⁵All averages are long term averages. Some methods of collecting stable average statistics for 802.11 are described in [65, 71]. We follow very similar techniques.

probability. This is expected, as the threshold for deferral is lower than the threshold for successful packet delivery.

2.8.2 MODEL EVALUATION AND VALIDATION EXPERIMENTS

This part of the work concentrates on the target network – the 12-node testbed described before. Average rss values for all link pairs in the network are collected. Here, each node takes turn to transmit UDP broadcast packets and every other node measures the average rss values. Again, this process is similar to measurements reported in [65, 71]. This takes $O(N)$ steps for an N node network. The profiles generated in the previous section and the values collected are used to seed the 802.11 model described in Section 2.4. Both analytical approach and direct simulation can now be used to solve the model to determine the throughput capacity of any given link. We have written a simulator in C which implements the Markov Chain based 802.11 model. We use a slot size of $10\mu s$ in 802.11b, which is small enough such that all protocol states span integer number of slots. We will later see in Section 2.8.3 that there is an accuracy vs. computation time tradeoff between these two methods.

For validation, we perform direct measurements on the testbed to evaluate link capacities and then compare them with those estimated by the model. In each validation experiment, n nodes are chosen from the testbed as transmitters while the remaining $12 - n$ nodes act as receivers. Each transmitter then broadcasts packets as fast as possible (to model saturated traffic) for 60 seconds. At the end of this time period, the throughput on each one of the $n(12 - n)$ links is measured by counting the number of packets received from each sender. For each such link, there are $n - 1$ interferers. We also measure the transmission capacity (number of packets *actually transmitted* in the air per second) for each transmitter. This quantity is reported by the card to the Madwifi driver.

We have performed validation experiments with up to 5 interferers. When $n = 2$, it is a single interferer scenario. Here, we have measured all possible combinations of such

scenarios, which require 66 experiments, and provide data for 132 transmitters, and 1320 links. When $3 \leq n \leq 6$, we randomly pick 50 configurations of n transmitters each, which results in data for $50n$ transmitters, and $50n(12 - n)$ links. Thus, overall we have performed 266 sets of experiments resulting in 7820 data points in the plots to be presented next.

Figure 3.6 shows the CDF of the absolute error (modeled – measured) in transmission capacity for both solution approaches – analytical and direct simulation. We specify capacity as a fraction of the channel bit rate. Note that the model performs quite well for fewer interferers, increasingly losing accuracy with more interferers, where the approximations used in the modeling and measurement errors start mattering more. Also, note that simulation provides better accuracy relative to the analytical method. This is expected due to the approximations used in the analytical method.

Exactly similarly, we present the absolute error between estimated and measured link throughput capacities in Figure 3.7. Once again note the excellent accuracy. It may appear here that the accuracy is more than for transmission capacities in Figure 3.6. This appearance is due to the fact that throughput capacities are smaller than transmission capacities; thus absolute errors are also smaller. The horizontal scale of both the plots are the same. The summary statistics for the errors will be presented momentarily in the following subsection.

COMPARISON WITH SIMPLER MODELS

It is instructive to compare our model with simpler models that one would use in absence of approaches such as ours. We use three simple models for comparison - (i) *naive model* (also used in [71]), where the link delivery ratio on a link is used as an estimate of link capacity; (ii) *delivery-based model*, where sender-side interference is modeled by assuming that the normalized transmission capacity of the sender is $1/(1 + \text{no. of neighbors})^6$ and then multiplying this number with the link delivery ratio; (iii) *distance-based model*, fashioned after the protocol interference model [34]). Here, transmit range, interference and carrier

⁶Here, a node is a neighbor if it has a link with at least 90% delivery ratio.

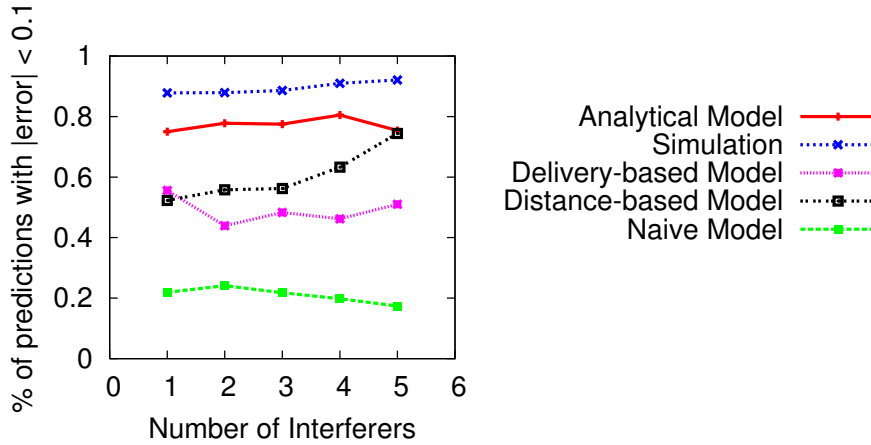


Figure 2.8: Summary error statistics for different models for different numbers of interferers.

sensing ranges⁷ are first determined based on a set of independent measurements in the same environment. If two links have a receiver in interference range of the other sender, or have the senders in each others sensing range, then they are said to be in conflict. The normalized capacity of a link in this model is $1/(1 + \text{number of conflicting links})$. The capacity is 0 for non-existent links (i.e., sender and receiver are outside transmit range).

We compare these models with our analytical and simulation approaches. The CDFs for errors for these models are also plotted in Figure 3.7. Note that the naive model overestimates capacity a lot, as it ignores interference. The delivery-based model also overestimates significantly as it does not have any way to model the receiver-side interference. This is very apparent from the plots with small number of interferers. On the other hand, the distance-based model underestimates significantly. This is likely because of conservative range estimates and the mistaken assumption that the ranges are isotropic. For larger number of interferers, it appears that errors are going down for the delivery-based and distance-based models. This is an illusion as the capacities are also smaller with larger number of interferers and thus absolute values of errors are also smaller.

⁷90% probability for respective events are considered for estimating ranges.

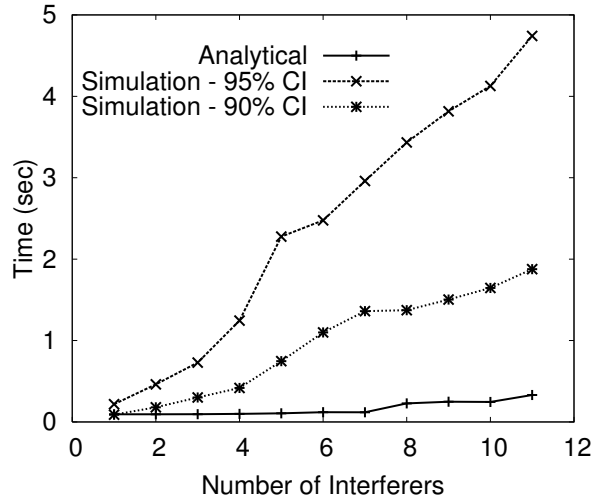


Figure 2.9: Computation time for analytical and simulation approaches with increasing number of interferers.

Figure 2.8 summarizes the errors in estimating link capacities for all the models in terms of percentage of predictions with absolute error within 10%. This data directly comes from the CDF presented in Figure 3.7. Note again that the simulation-based solution is the most accurate, estimating capacities of links more than 90% of the times with an error that is within 10% of the channel bit rate. This goes down to about 80% of the times in the analytical approach. The simpler models typically perform much worse, though distance based model becomes competitive for larger number of interferers. Going back to the example in Section 6.1 note that both delivery and distance-based models do a very poor job in modeling the ‘flow-in-the-middle’ scenario (see Table 1).

2.8.3 COMPUTATION TIME

Recall the discussion on the complexity of solving the equations for the analytical solution from Section 2.6. Long computation time will limit the applicability of our approach. We thus need to analyze the computation time issues.

For the plots presented above, we have used Mathematica [3] to solve the equations. It uses the Newton’s method [50] for solution, which in turn uses a method of linear approx-

imation. Newton's method requires a set of good starting values for rapid convergence. We have seeded the variables with the values from the delivery-based model. Note that this model is computationally very cheap. We have set the maximum number of iterations to 1000. We have also specified the stopping criteria, such that the iterations stop when an accuracy of 10% has been achieved.

For simulations, we have used the *batch means method* to ensure that simulations converge to a target level of confidence. Our target for the above validation plots has been 95% confidence interval of batch means being less than 5% of the overall mean statistics. Figure 2.9 shows the computation time for the analytical approach and the simulation approach. For simulations, we now also add the times for a less accurate simulation (90% confidence interval less than 5% of mean). This demonstrates a tradeoff between accuracy and computation time. Simulations are almost an order of magnitude slower than the analytical approach. The computation times are reported for a Dell PC with a 3 GHz Pentium processor with 4GB of RAM, running Linux. For the purpose of this plot, we have evaluated computation times up to 12 transmitters (i.e., 11 interferers) in our 12 node testbed. Note that the time to solve the analytical approach increases very slowly, and is approximately 0.35 sec even with 11 interferers. The trend indicates that with a powerful computer, the analytical solution approach should be useful even for on-line decision making for resource scheduling, at least at a coarse time scale (second or sub-second). For example, for applications such as coarse-grain channel assignment, admission control, centralized routing, etc. 0.1-1 second computation time is easily affordable. Studying the computational issues further is on our future research agenda.

2.8.4 VALIDATION FOR NON-BACKLOGGED INTERFERERS

In this section we present some validation results to demonstrate that our model extends to the case when the interferer is not backlogged. We show the capacity of a link in presence of an interferer for three cases – when the interferer causes sender-side interference, when

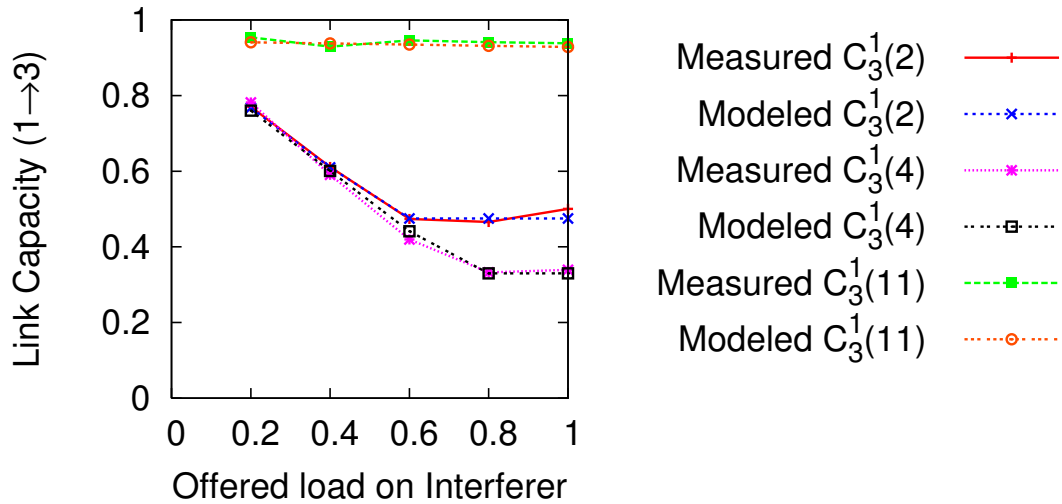


Figure 2.10: Capacity of a link (1,3) in presence of one non-backlogged interferer. Interferer 2 contends for channel with 1; 4 causes collisions, while 11 does not effect the link.

the interferer causes receiver-side interference, and when the interferer does not affect the link at all. To do this, we pick one link in the testbed and choose three suitable nodes as interferers to validate these three cases. In each case we determine the capacity of the link in question from our model and compare it with the measured capacity. Figure 2.10 shows the accuracy of our prediction using analytical modeling in each case.

2.9 CONCLUSIONS

In this chapter, we have addressed the challenging problem of modeling link capacities in a real, deployed 802.11 network. This is a departure from the existing methods of analytical or simulation-based modeling that often make unrealistic assumptions. Our model is based on the realistic physical interference model that drives a discrete time Markov chain-based model of 802.11 behavior. The physical interference model is profiled using measurements and is seeded again by measurements on the target network to be evaluated. The methods we proposed are practical – (i) The profiled measurements can be kept in a library and reused. (ii) The measurements on the target network are simple and take $O(N)$ steps. (iii) The analytical solution time is of “sub-second” scale opening up a lot of applications that

use course-grain decision making, such as overlay MAC scheduling, routing, admission control and channel assignment. Our future work will explore some of these applications using the proposed model.

While we have used a single channel, single packet size, single data rate and single interface card model in our work here, this is not a limitation. Profiling can be done for all these parameters separately. Some additional modeling can indeed help in profiling effort. For example, profiling for one size can possibly be extrapolated for other sizes. In principle, the modeling approach is able to handle heterogenous systems, where different nodes use different parameters, so long as cards with all such parameter settings have been profiled for. The harder problem is handling dynamically changing parameters, for example, auto rate control in 802.11. In this case, the rate control algorithm must be modeled as a part of our approach. Also, our approach is general enough such that extensions of 802.11 (e.g., 802.11e) can be modeled using a similar Markov model, though more states probably will make the solutions more compute intensive.

CHAPTER 3

MEASUREMENT-BASED APPROACHES FOR ACCURATE SIMULATIONS

In this chapter, we address the issue of unrealistic simulations of wireless networks using a measurement-based approach. The idea is to use empirical modeling using measurement data as a mechanism to model physical layer behavior, as in Chapter 2. We demonstrate the power of this approach for 802.11-based networks using ns2, a packet-level network simulator. We build the models for deferral, reception, and signal propagation using measurements from a real network.

3.1 INTRODUCTION

Simulation-based modeling is a useful tool for evaluating performance of network protocols. Simulations served the networking community well for wired networking regime. However, simulations for wireless networks have often been questioned [54, 12], primarily due to the lack of realistic lower layer models. However, the research community has not yet practiced serious validation exercises for wireless network simulators barring minor exceptions [60]. Our goal in this work is to revisit the issue of unrealistic simulation models of wireless networks for the lower layers, and address the problem using a new approach that uses measurement-based modeling.

Network simulators widely used in wireless networking literature such as ns-2 [7], qualnet [6], opnet [5] etc. implement the network protocol layers in the same fashion as in a real system. The upper layer implementations (such as transport and network) are fairly accurate. This is because they are implemented in software in a real system. This makes it easier to model them in the simulation software. This is also true for MAC-layer

models as detailed specs and firmware implementations are available to a serious simulation modeler. However, the wireless physical layer has been hard to model. While theoretical models do exist, they make assumptions on the propagation environment and the interface characteristics and use various model parameters (e.g., path loss exponent) that are hard to instantiate. Also, often such models work at a much finer timescale (at the bit or symbol level, e.g.) while popular network simulators operate at a packet-level time scale. Making the timescale finer may cause a serious slowdown of the simulator eliminating the scalability benefit – one possible reason why such attempts have not been seriously pursued yet. On the other hand, research has shown that physical layer modeling can make serious impact on the upper layer protocol performance [79] thereby making realistic modeling all the more important.

Our goal here is to propose measurement-based approaches to model the physical layer of protocol stack so that not only popular packet level simulators can still be used, but also the simulation accuracy is vastly improved. The approach is not simulator specific, but we have used ns-2 because of its popularity. Similarly, our work is not MAC/radio specific, but we focus on 802.11 because of its ubiquity. We identify three components that comprehensively capture the physical layer behavior in an 802.11-based network. They are (i) signal propagation model, (ii) carrier sensing model on the sender side, and (iii) packet reception model on the receiver side. We propose measurement-based approaches to model the above three components. The idea is to use measurements to preserve realism where analytical models are inadequate.

We validate the accuracy of the measurement-based approaches vis-a-vis direct experimentation on a 12-node 802.11-based indoor mesh network testbed. Our general conclusion is that the technique is very accurate when measurement data from an actual testbed is available. When complete testbed is not available for measurements, measured data from a limited set of nodes can also be used for modeling using the proposed approach while providing high level accuracy compared to existing simulations approaches. Our

hope is that this study will encourage the wireless networking research community to use measurement-based techniques for simulation studies. Wide adoption will also lead to reuse of measurement-based models making the approach very cost-effective in terms of effort.

3.2 RELATED WORK

In [54], the authors describe unrealistic assumptions often made in wireless network simulators. They also develop a simulator that they validate against real experiments; however they report experiments related to propagation modeling only. Several emulation approaches are described to validate wireless ad hoc network simulations in [41]. However, here comparisons against real networks are not reported. In a recent comprehensive article [12] the authors survey many questionable practices for simulating mobile ad hoc networks. They note inadequate modeling of protocols and lack of validations as two major issues. They also note other issues such as improper documentation, or lack of statistical validity that are not explored in our work. In [60], a validation approach has been developed using direct execution simulators for ad hoc networks.

In [35], the effect of detail in wireless network simulations and how they influence the conclusions are studied. In [79], a careful study is done using different simulators that shows how the details in physical layer modeling can impact upper layer protocol performance in a simulator. Physical layer emulations [42] and various hybrid approaches [86] have recently been promoted to impart realism to modeling studies. However, they are quite complex, require significant amount of hardware and are yet to be widely adopted.

The measurement approaches discussed in this chapter have similarities with several recent works, such as [71, 17, 65, 24, 49] for 802.11 networks and [75] for Berkeley mote-based networks. These papers emphasize the significance of using measurements over analytical modeling. Some of these papers also promote using just pairwise signal strength measurements between nodes to model interference and its impact. We utilize these ideas in our work in the context of creating an accurate and realistic wireless network simulator.

3.3 APPROACH

The physical layer components in an 802.11 network simulator can be classified into three broad categories – (i) *radio propagation model*, (ii) *deferral or carrier sense model* on the sender side, and (iii) *packet reception model* on the receiver side. We describe our approach to handle them below.

3.3.1 PROPAGATION

Typically, wireless network simulators assume a generic propagation model, such as free space model or two-ray ground reflection model coupled with a shadowing model [70] as in ns-2. Naturally, such a generic model may not be appropriate for the propagation environment to be evaluated. Further, parameters of such models (e.g., path loss exponents) still need to be instantiated. Our approach here is as follows.

- a. *If a testbed is available*, we perform direct measurement on the testbed to determine propagation behavior. Here, the receiver simply measures the received signal strength (RSS) and no real modeling is performed. This requires only $O(N)$ measurements for an N node network. Each node can transmit a beacon and every other node simply measures the RSS. Commodity 802.11 interfaces allow such measurements.¹
- b. *If a testbed is not available (but a pair of network nodes are available)*, we model the propagation behavior using an empirical, measurement-based approach in the environment being considered. This is not unlike early work in cellular communications that gave rise to popular empirically derived models such as Okumura-Hata models [70]. A similar modeling approach has also been considered in outdoor 802.11-based networks with reasonable accuracy [17].

¹Note that there are subtleties here that commodity cards allow RSS measurements only when the packet is received correctly. Prior measurement studies indicated that impact of this is relatively minor [71].

3.3.2 DEFERRAL AND RECEPTION

Carrier sensing in 802.11 cards is implemented using a channel acquisition module, which determines whether the channel is idle for transmission. This is modeled in simulators by using a carrier sense threshold, and a received signal with higher power than this threshold makes the channel busy. It has been observed [65] that carrier sensing between a pair of nodes is not deterministic, and in practice, if a pair of nodes attempt to transmit simultaneously, the probability that one node defers due to other may be a value somewhere between 0 and 1.

Modeling the packet reception is harder. This depends on signal to interference plus noise ratio or SINR, where signal is the received signal power and interference is the aggregate of the interference powers received at the receiver. Interference is simply signal transmitted by any node other than the designated transmitter. Fundamentally, SINR affects the bit-error rate (BER) in a received packet [70]. The SINR vs. BER relationship typically depends on receiver design and modulation used. BER ultimately affects PER (packet-error rate) depending on the coding used. Note again the probabilistic nature of packet reception. Usually, there is a sharp fall in BER (and hence PER) with increasing SINR. Thus, often simulators simplify this by assuming a simple two-step function to model SINR vs. PER relationship. This essentially translates to the so-called *capture threshold*, signifying an SINR threshold needed for successful packet reception. Even when modulation/coding specific SINR vs. PER relationship can be used (the best case), it is unclear whether a universal theoretically based model would suffice for any interface.

3.3.3 MODELING STRATEGY

Direct measurements are possible for modeling the propagation behavior in Section 2.1 (using $O(N)$ measurements). However, similar direct measurements are not possible for modeling deferral or packet reception behavior, *even when a testbed is available*. The reason is that all possible subsets of transmitting nodes must be considered, requiring an

	Direct Measurement	Measurement-based Model	Theoretical Model	Theoretical Model (Simple)
Propagation	V4	V3		V2 V1
Reception		V4 V3	V2	V1
Deferral		V4 V3		V2 V1

Figure 3.1: Versions of the simulators considered and the models used by them.

exponential number of measurement steps. This requires us to take an empirical modeling approach that still only uses $O(N)$ measurement steps and the rest is done via modeling. The modeling part assumes that only aggregate interference power is important to determine deferral or reception, and not individual interference powers or number of interferers. Note that this assumption should be true in theory. We have indeed performed limited amount validations to test this out (reported in the next section).

We develop several versions of the ns2 simulator, *only differing in the physical layer implementation*. To describe the simulators better, let us categorize the propagation, deferral and packet reception modeling in the simulators in 4 categories. See Figure 3.1. We name the simulator versions V1 to V4, with increasing complexity. V3 and V4 replace the entire physical layer by our measurement-based model. The difference in V3 and V4 is that in V4, direct RSS measurements are used to model propagation (note (a) in Section 2.1); while in V3, a model is used for propagation that is derived from measurements (note (b) in Section 2.1).

V1 and V2 use simpler models. V1 is very similar to the default ns-2 simulator. Here, the propagation model is a free space propagation model, reception is based on a SINR threshold,² and deferral is based on a carrier sense threshold. These thresholds are tuned

²The default ns2 has an even simpler reception model, where it simply compares signal with one interferer only at a time. V1 makes it somewhat more realistic by using a true SINR computation.

using measurement data as a guide. V2 differs from V1 in that it uses a somewhat more sophisticated model for packet reception based on the theoretically derived PER vs. SINR curves [8].

3.4 MEASUREMENT-BASED MODELS

In this section, we present the measurement-based models we use for the simulators. All measurements were done on our experimental testbed consisting of 12 Dell Latitude D520 laptops running Linux 2.6.15 kernel. The testbed is located in one floor of an office-cum-lab environment. See Figure 3.2 for a network diagram. Each laptop uses a DLink AirPremier DWL-AG660 802.11/a/b/g PC card with Atheros AR5212 chipset. The Madwifi driver, Version 0.9.6 [4] is used. The cards are configured in ad hoc mode when used as transmitter, and in monitor mode, when used as receiver. RSS measurements use the appropriate field in the prism monitoring header which is obtained whenever a packet is captured when the card is in monitor mode. The value reported by Atheros cards is $10 \log_{10}(S + I/N)$, where S is the signal strength and I is the interference, N is fixed at -95dBm (noise floor). All experiments reported here are done for 802.11b. We also did similar set of validations for 802.11a and have very similar experience. But we choose to present 802.11b results here as it gives longer range links and has a rich set of interferences in our testbed. The experiments are done in nights when interference from external 802.11 networks is expected to be minimal. All experiments are done at the lowest PHY-layer rate (1 Mbps) and with large (1400 bytes) packet sizes.

3.4.1 MODELING PROPAGATION

Radio propagation in indoor environment is a complex phenomena. There are three main factors that play a role in determining the received signal power – path loss, shadowing and multipath fading [70]. At a high level, path loss describes the exponential decay of signal power with distance, with the exponent depending on the propagation environment.

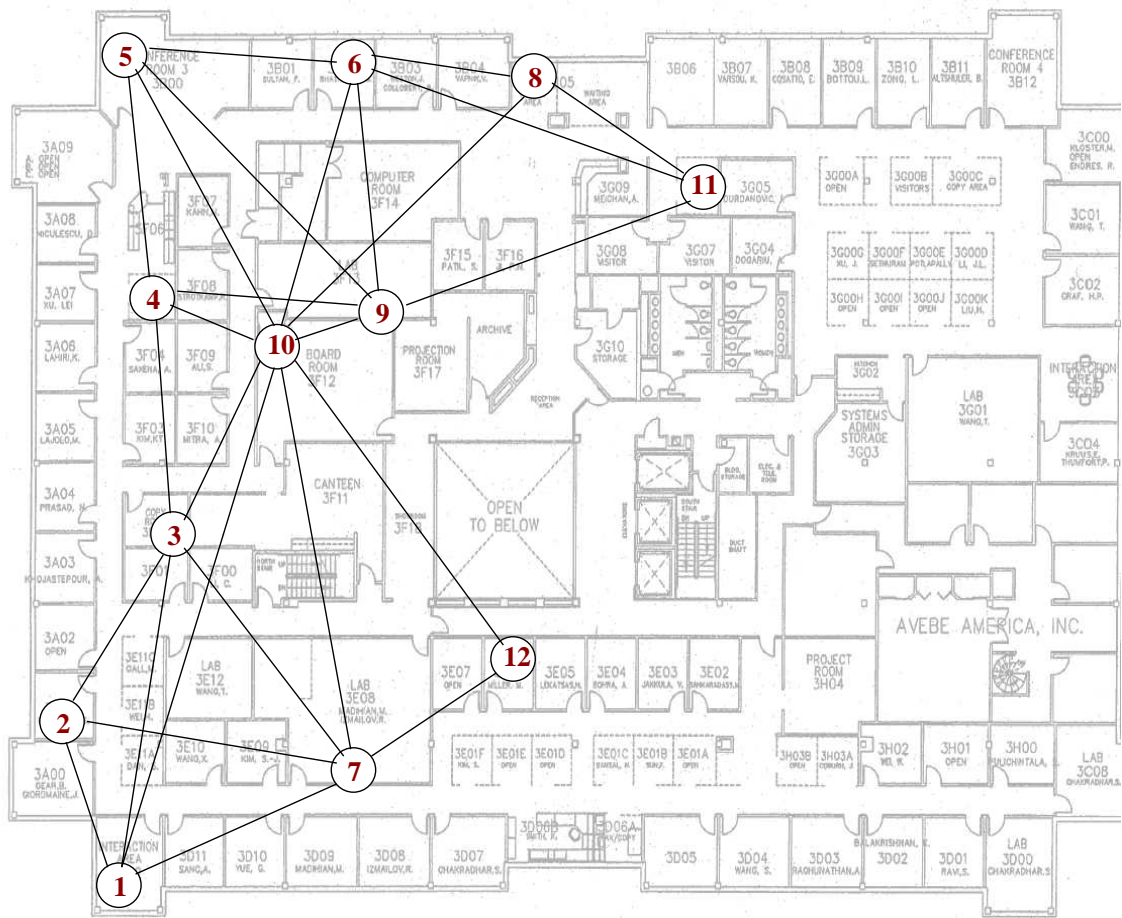


Figure 3.2: Locations of the nodes on the floor map and links with more than 90% delivery ratio. Width of the map is 60m.

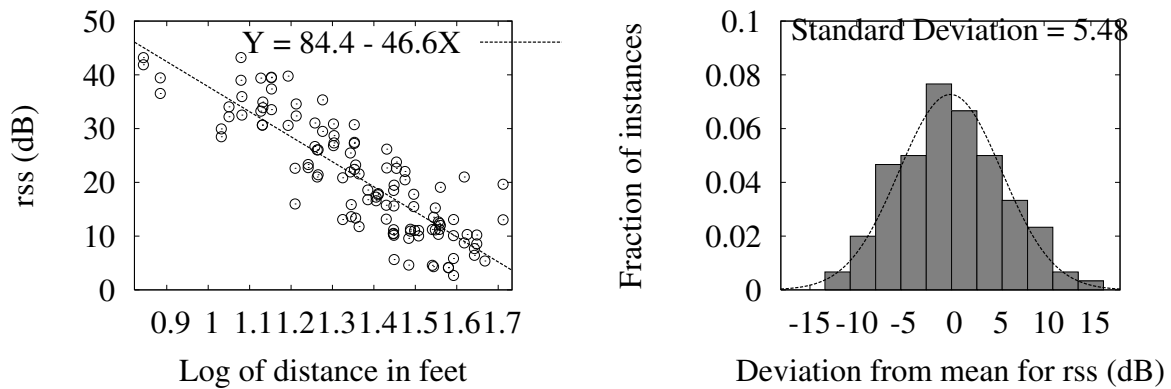


Figure 3.3: (a) Measurement data showing received signal strength vs. distance and also the least-square fit. (b) Empirical estimation of the shadowing model.

Shadowing describes random variation of path loss in similar propagation environment, commonly modeled by a Normal distribution in dB (log-normal shadowing). Following a similar modeling work [17] we ignore multipath fading due to its modeling complexity and impact only in small time and spatial scales.

Combining path loss and log-normal shadowing, we have

$$P_{dB}(d) = P_{dB}(d_0) - 10\alpha \log_{10} \left(\frac{d}{d_0} \right) + X_\epsilon, \quad (3.1)$$

where $P_{dB}(d)$ is the received signal power at distance d , d_0 is a reference distance where power measurement is already available, α is the path loss exponent, and X_ϵ is a Normal random variable in dB having a standard deviation of ϵ dB and zero mean. The path loss exponent α is 2 in free space, but is higher in a cluttered environment.

We use an empirical method to estimate α and ϵ from measurement data following similar work in [17]. We collect average RSS values for each pair of nodes in the testbed from 132 separate measurements (12 transmitters \times 11 receivers) and use least square linear regression to find the path loss exponent for our testbed environment. Figure 3.3(a) shows the scatterplot, and the fitted line, which gives the path loss exponent α as 4.66. Similarly, ϵ is estimated by fitting a Normal distribution for the error values in the above regression. See Figure 3.3(b). We get $\epsilon = 5.48$. We use this model in the simulators V3. Note that if a complete testbed is not available, but only a couple of nodes are available, we can still create this model by performing a large number of RSS measurements by placing just two nodes in different random locations in the test environment.

3.4.2 MODELING DEFERRAL

The first step is to create an empirical relationship for the probability of deferral between two nodes based on received signal strengths. We express this relationship as a function $f(\cdot)$, such that $p_i^j = f(rss_i^j)$, where p_i^j is the (deferral) probability that node i defers to the transmission of node j and rss_i^j denotes the measured values of average signal strength of packets transmitted from node j and received at node i . We determine function $f(\cdot)$

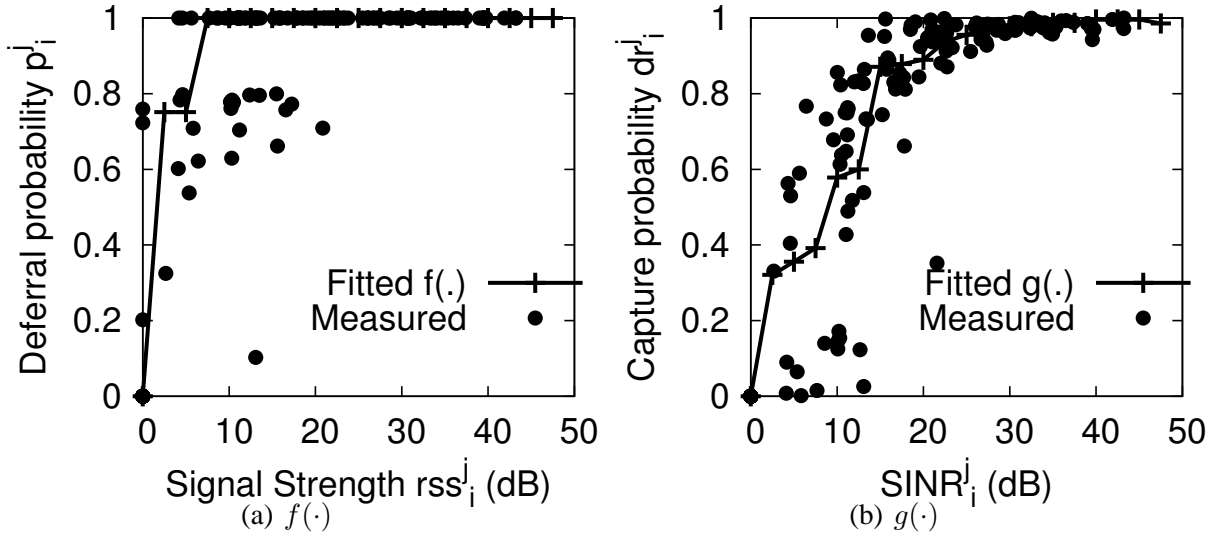


Figure 3.4: Determining (a) deferral and (b) packet reception probabilities.

simply by taking two nodes and positioning them in many random locations in the test environment, and then directly measuring the RSS values between them as well as the deferral probability.

The deferral probability is measured as follows. Both nodes attempt to broadcast UDP packets as fast as possible. Thus, they always have backlogged traffic. We measure the transmit rate (rate at which a node is transmitting packets on the air) of each node. We also measure transmit rate when the node is transmitting alone. The ratio of these two rates gives the deferral probability p . A large number of such measurements $\langle p, rssi \rangle$ are taken and are shown in the scatterplot of Figure 3.4(a). $f(\cdot)$ is estimated as the linear interpolation of average values of p for small buckets of $rssi$ values. Further, it is assumed that deferral probability p depends only on the sum of $rssi$ values if multiple transmitters are present. Thus,

$$p_i^Y = f \left(\sum_{j \in Y} rssi_i^j \right), \quad (3.2)$$

where Y denotes a set of active transmitters.

3.4.3 MODELING PACKET RECEPTION

A similar approach is taken for modeling the packet reception behavior. Define *delivery ratio* dr_i^j from node j to node i as the fraction of packets received by i that are transmitted by j in the absence of any other interfering transmitter. Let us define $dr_i^j(Y)$ as the delivery ratio from j to i in presence of the set of interferers Y . Our first task is to model dr_i^j as $dr_i^j = g(rss_i^j/\text{noise})$. This simply relates packet reception (capture) probability to SNR, the ratio of the received signal strength and noise. Here rss_i^j denotes the average signal strength of packets received from j to i in absence of interference.

Once the function $g(\cdot)$ has been modeled, $dr_i^j(Y)$ can be expressed as follows:

$$dr_i^j(Y) = g(SINR_i^j(Y)), \quad (3.3)$$

where,

$$SINR_i^j(Y) = \frac{rss_i^j}{\sum_{k \in Y} rss_i^k + \text{noise}}. \quad (3.4)$$

As in the case of equation 3.2, the above equation also requires only pairwise measured rss values in the deployed network.

A set of experiments as before is devised to empirically model $g(\cdot)$. Two nodes are placed in many random locations. One of them transmits broadcast UDP packets and the other receives. The average dr and rss values are recorded at the receiver. The scatterplot in Figure 3.4(b) shows the experimentally obtained values. The function $g(\cdot)$ is obtained via interpolation as before. As stated before, these results are for the lowest PHY-layer rate (1 Mbps), and thus the $g(\cdot)$ function is specific to this data rate. Similar experiments must be done at all data rates to get the rate specific $g(\cdot)$ functions.

Note that the empirical technique above measures SINR without any interferer (thus, actually SNR) with an assumed noise floor (-95dBm). We have also validated that indeed when one or more interferer is added, the function $g(\cdot)$ estimated above still holds. Figure 3.5 shows the experimentally obtained delivery ratio vs. SINR scatterplot in the presence of 1 interferer. Note the similarity of this plot with Figure 3.4(b). This provides

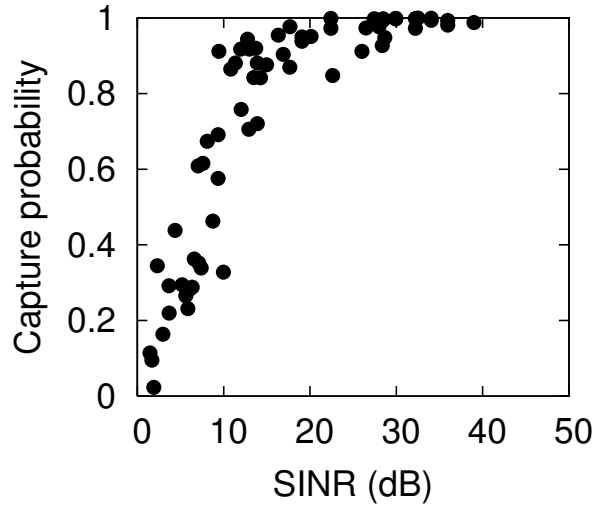


Figure 3.5: Capture probability versus SINR in presence of one interferer.

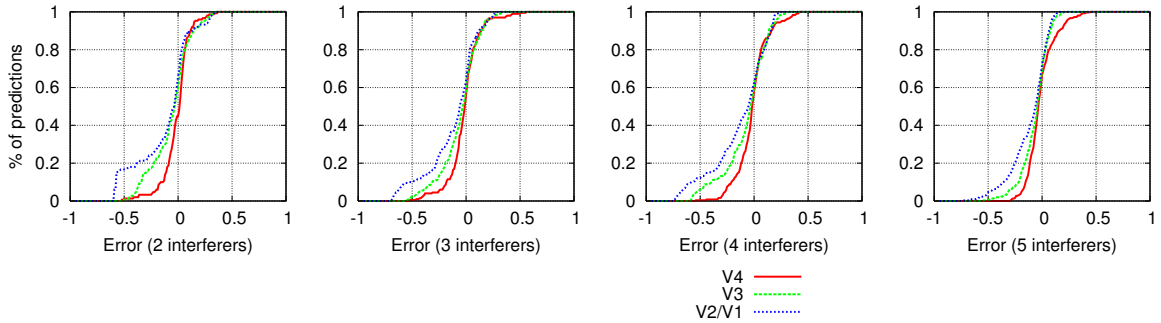


Figure 3.6: CDF of error between the estimated and measured transmission capacity of senders.

credence to our approach that function $g(\cdot)$ can be modeled using measurements without any interferer, and thus requires only $O(N)$ measurement steps.

3.5 EVALUATION

We evaluate the accuracy of the simulators on the target network – the 12-node mesh testbed described before. Average RSS (r_{ss}) and delivery ratio (dr) values for all link pairs in the network are collected. Here, each node takes turn to transmit UDP broadcast packets and every other node measures the average r_{ss} and dr values. This process is similar to

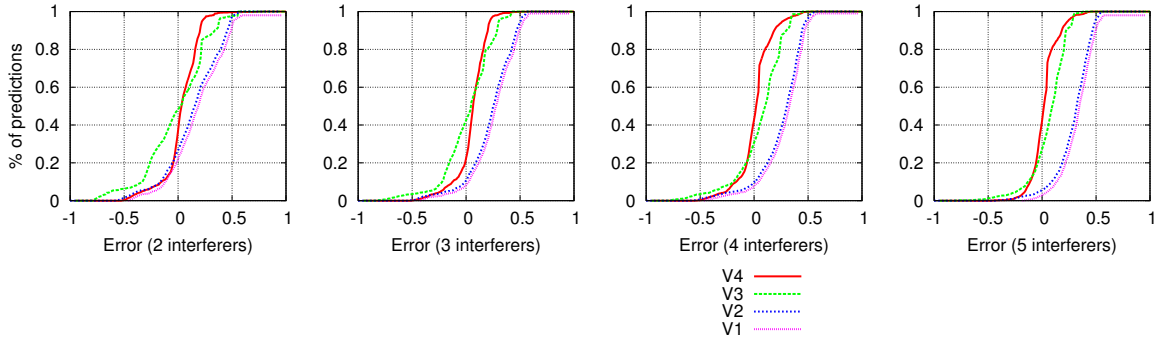


Figure 3.7: CDF of the error between the estimated and measured throughput capacity on links.

measurements reported in [65, 71]. This takes $O(N)$ steps for an N node network. The rss measurements are used to seed simulator V4, while the rss and dr measurements are used to create the deferral and reception model for simulators V4 and V3.

For validation, we perform direct measurements on the testbed to evaluate link capacities and then compare them with those estimated by the various versions of the simulators. In each validation experiment, n nodes are chosen from the testbed as transmitters while the remaining $12 - n$ nodes act as receivers. Each transmitter then broadcasts packets as fast as possible (to model saturated traffic) for 60 seconds. At the end of this time period, the throughput on each one of the $n(12 - n)$ links is measured by counting the number of packets received from each sender. For each such link, there are $n - 1$ interferers. We also measure the transmission capacity (number of packets *actually transmitted* in the air per second) for each transmitter. This quantity is reported by the card to the Madwifi driver.

We have performed validation experiments with up to 5 interferers. When $n = 2$, it is a single interferer scenario. Here, we have measured all possible combinations of such scenarios, which require 66 experiments, and provide data for 132 transmitters, and 1320 links. When $3 \leq n \leq 6$, we randomly pick 50 random sets of n transmitters each, which results in data for $50n$ transmitters, and $50n(12 - n)$ links. Overall, we have performed 266 sets of experiments resulting in 7820 data points in the plots to be presented next.

Figure 3.6 shows the CDF of the absolute error (i.e., estimated – measured) in the sender side transmission capacity for the various simulators. We present capacity normalized to the channel capacity. Since V1 and V2 use the same deferral and propagation model, the transmission capacity of these two simulators are identical. Note that V4 is quite accurate – the error is within 10% of the channel capacity 85% of the times. V3 is less accurate than V4 (the error is within 15% of capacity 85% of the times), because V3 uses a model for propagation rather than using direct measurement. V1 and V2 *underestimate* the transmission capacity significantly, likely because they model a weaker path loss. This results in more deferral and lower transmission capacity.

Exactly similarly, we present the absolute error between estimated and measured link throughput capacities at the receiver side in Figure 3.7. Once again, note the excellent accuracy for V4 followed by V3. The 85 percentile error for V4 and V3 is 10% and 15% of capacity, respectively. Note again V2 and V1 provide very poor estimation, *overestimating* the capacity this time. In the case for V2 and V1, the throughput capacity is almost the same as the transmission capacity as collisions rarely happen because of almost perfect deferral. In reality, however, many more packets are actually transmitted, but many of them actually lead to collisions leading to much poorer received throughput.

The take-home message from these results is that careful measurement-based modeling can be successfully used to develop accurate simulators (V4 and V3). When measurements are not used, even when best possible strategies are used in the simulation models (e.g., V2), the errors are very high. For example, for estimating throughput capacity, for 85% of the scenarios, the error in V2 increases to 50% of the channel capacity.

3.6 CONCLUSIONS

In this chapter, we have demonstrated that empirical modeling of the physical layer is necessary in building more accurate wireless network simulators. We have specifically focused on 802.11 and developed two versions of the popular ns-2 simulator that model the wireless

physical layer with different levels of fidelity. In both versions, the deferral and reception model are built using measurements. For the propagation modeling, one version (V4) uses direct measurements and the other (V3) uses a modeling approach. In validation experiments over a 12-node mesh testbed, both these versions were found to be reasonably accurate (85 percentile errors about 10% of capacity). Simulation errors in more traditional simulation models were found to be unacceptably high (85 percentile error within about 50% of capacity). Our future work will focus on improving the error margins and validating our simulators with unicast traffic, with relayed traffic, etc. The eventual goal is to make such simulators using various measurement data sets available to the research community for evaluating protocol performance.

CHAPTER 4

ESTIMATING INTERFERENCE USING PASSIVE MONITORING

In this chapter, we present an approach to estimate the interference between nodes and links in a live wireless network by passive monitoring of wireless traffic. Unlike Chapter 2 and Chapter 3, this does not require any controlled experiments or injection of traffic in the network. Our approach requires deploying multiple sniffers across the network to capture wireless traffic traces. These traces are then analyzed to infer the interference relations between nodes and links. We model the 802.11 MAC as a Hidden Markov Model (HMM), and use a machine learning approach to learn the state transition probabilities in this model using the observed trace. This in turn helps us to deduce the interference relationships.

4.1 INTRODUCTION

Interference between radio links limits the capacity of a wireless network. The Holy Grail of research thus has been the understanding and modeling of interference and its impact on network capacity, ranging from very abstract modeling exercises [34] to more practical analysis and evaluation [49, 67]. Needless to say that the practice has been mostly targeted for WiFi (IEEE 802.11 Standard and its derivatives) networks, as it is by far the most predominant wireless access network for end users.

While a lot of work has been done, WiFi network installations are yet to gain from these results. Interference is a serious concern in WiFi because the 802.11b/g band – popular because of its good propagation characteristics relative to the 802.11a band – provides only three orthogonal channels limiting the scope for channel diversity. While one might expect that dense AP deployments – as it is getting common nowadays – would provide for

association diversity that would reduce interference, it is not unusual in practice to find multiple APs within the range operating on the same channel. Poor WiFi network performance in congested scenarios has become well-documented measurement literature [40, 72].

Our goal in this work is to model and understand the prevalent wireless interference in a real WiFi network installation. From a practical standpoint, we need to do this in the most unobtrusive fashion possible, in particular (i) *without installing any monitoring software on the network nodes*, and (ii) using a *completely passive technique*. The need for (i) comes from a matter of practicality. Many APs are often closed devices, and clients may not be always accessible for monitoring software installations. The need for (ii) is more obvious. Active measurements impact (and are impacted by) network traffic. Our approach thus requires the use of a distributed set of ‘sniffers’ that capture and record wireless frame traces. The traces are to be analyzed later for understanding interference relations. While this requires additional hardware for measurement and analysis, we view this as a third-party solution. Such independent third-party solutions for wireless monitoring are not uncommon in industry [1, 2]. The research world has also provided similar approaches. See, e.g., DAIR [13, 14], Jigsaw [21] and Wit [61]. While these approaches provide many monitoring solutions, they still do not provide fundamental understanding of interference relations between network nodes and links.

Our approach can be used as a toolbox to understand the interference properties in an arbitrary WiFi network. This can in turn help the system managers to perform capacity planning and perform appropriate radio resource management, such as use of channels, transmit powers or directional antennas.

4.1.1 APPROACH

Existing measurement-based modeling studies [49, 67] for modeling and analysis of interference in WiFi networks require elaborate measurement exercises that include (a) profiling of the behavior of the radio interfaces to learn the carrier sense and packet capture behav-

iors, and (b) RSS (received signal strength) measurements on the network nodes to learn link-wise RSS values. None of these are very practical in the setting we consider. These will require profiling every type of interface used in the network, installing RSS measurement code in all nodes, and measuring RSS specifically in quiet periods.

Thus, we take a fundamentally different approach. The distributed set of ‘sniffers’ collect traffic traces from the live network. These sniffers do not transmit any packets making the method completely unobtrusive. The traffic traces are later collated, merged and analyzed off-line to determine which link pairs interfere in the network. Merging of traffic traces is an important problem by itself. Here, we benefit from existing work [84, 85, 61, 21] that developed merging techniques with distributed sniffers.¹ Then, a machine learning approach using the *Hidden Markov Model (HMM)* [68] is used to analyze the merged traces to infer interference relationships. Since the approach is passive, it is only dependent on the sufficiency of the available network traffic for the interference analysis. The challenge in this case is to make accurate estimates even in presence of little traffic, and traffic of unknown and arbitrary nature. This is important as all network APs may not be heavily used all the time. There are indeed many other issues related to the location of the sniffers, fidelity of the merged traces, etc. that will impact the accuracy of the technique to a varying degree.

4.2 RELATED WORK

4.2.1 ANALYZING INTERFERENCE

Interference in an 802.11 wireless network can be readily measured by putting saturated traffic on two nodes or links simultaneously and measuring the aggregate throughput. The decrease in throughput due to interference from the other transmission indicates the amount of interference. This approach ordinarily needs $O(n^4)$ measurements for an n node network. However, the work in [65] shows this can be done with only $O(n^2)$ measurements.

¹These techniques also infer and add the packets that are missing from the merged trace.

A more sophisticated approach does not perform direct measurements as above, but uses certain modeling steps to reduce the number of measurements to $O(n)$. The idea here is to measure RSS on each link using broadcast beacons. A profiling study describing the deferral and packet capture behavior of the radio interface along with the RSS measurements help forming a model of the physical layer behavior of the wireless interface. A MAC layer model is added to the physical layer model to form a complete model that can estimate interference between active links and link capacities in presence of interfering traffic. This approach is powerful enough to model realistic physical interference. Different variations of this basic approach have been presented in [71, 49, 67]. This method is still unrealistic in live networks as the RSS measurements need a quiet, interference-free environment. Also, the profiling study must be available.

In addition to the above, there are various sundry works on evaluating interference characteristics in an 802.11 network. For example, in [38], authors investigate the impact of carrier sensing. In [18] the authors develop a model for the physical layer capture. In [24], the authors show that pairwise interference modeling is often not accurate and multiple interferers must be accounted for.

4.2.2 USING DISTRIBUTED SNIFFERS

In contrast to the above methods requiring active measurements, we use passive monitoring of a live network using distributed sniffers. Previous studies have used distributed sniffers to conduct a range of measurements over live networks to learn various properties such as congestion [40], protocol behavior in a hotspot setting [72]. Bahl *et al* also has used such an approach in DAIR for troubleshooting [13] and security [14].

While earlier studies were conducted by analyzing individual traces, Yeo *et al* are the first to provide a technique to merge individual traces to create a unified view of the network activity [84, 85]. This unified trace, created using common references of beacons across traces, provides more opportunities to analyze the link level characteristics of a wireless

network. Cheng *et al* apply this technique for a large scale sniffer deployment to create a system called Jigsaw [21], which they use to perform fault diagnosis across multiple network layers in the network [20]. Mahajan *et al* develop a system called Wit, where they advance the technique of Yeo *et al* of merging traces by proposing an inference engine to guess any missing packets [61]. In our work, we employ the same technique to merge individual traces into a unified trace. However, unlike the above studies which focus on understanding MAC level behavior, anomaly or fault detection, our focus is on learning the interference in the network.

A recent work by Schulman *et al* questions the fidelity of such traces generated by multiple sniffers [73]. They argue that in a high load scenario, a large number of packets are lost and the timestamps of the packets may not be accurate due to clock drifts. Thus, the unified trace depicts an incomplete picture of network activity, and any inference based on that may be inaccurate. Our technique relies on having sufficient information rather than complete information, and can work even on incomplete traces. More is discussed about this aspect in Section 4.3.2.

4.3 OVERALL APPROACH

4.3.1 PROBLEM STATEMENT

In 802.11, links can interfere either at the sender side or at the receiver side or both. On the sender side, the interference is because of deferral due to carrier sense. On the receive side, it is because of packet collisions that require packet retransmission. In both cases, the sender additionally has to go through a backoff period, when the medium must be sensed idle.² The net effect of the interference is reduction of throughput capacity of the link.

For modeling convenience, we consider interference between link pairs only. Due to the additive nature of the received power, a given link in reality interferes with a set of other

²We are assuming that the reader has an overall idea of the 802.11 MAC protocol. Specific details will be brought up as necessary.

links (so called ‘physical interference’) [34]. This is because a single transmission may not generate enough power to cause deferral or collision for the given link, however, multiple such transmissions may still cause deferral or collision. However, pairwise consideration can still bring up a useful picture of interference. Also, in reality, multiple concurrent transmissions may actually be rare [61]. Thus, learning more elaborate higher order interference relationships may not be very useful in practice. We do note that this simplification is not fundamental to our basic technique. The technique can be extended, albeit with higher computational cost, to physical interference.

In wireless networks, interference is hardly deterministic. It is rather probabilistic because of the inherent fluctuation of the signal power due to fading effects and probabilistic dependency of error rates with SINR (signal to interference plus noise ratio). It is thus best to characterize the interference between two links as non-binary, using a number between 0 and 1. Prior measurement and modeling studies have elaborated on this aspect [65, 49]. Thus, our goal is to estimate via passive monitoring the non-binary, pairwise interference between any two network links, in terms of *probability of interference*. For every link pair, the probability of interference is given by:

$$p_d + (1 - p_d)p_c, \quad (4.1)$$

where p_d is the ‘probability of deferral’ between the senders, and p_c is the ‘probability of collision’ at the receivers if both senders transmit together. A previously proposed measurable metric also captures this probability. The metric is Link Interference Ratio (LIR) proposed in [65].

4.3.2 DISCUSSIONS

The major challenge of using passive monitoring is that one can identify whether two nodes or links interfere only if they both have packets to transmit at the same time. Obviously, the observed behavior of two links that otherwise would interfere, but never transmit together in practice, is no different from the case when the links do not interfere. Thus, our approach

is based on the conjecture that if we observe live network traffic for long enough period, such instances will arise where simultaneous transmissions are attempted in the network for each link pair. Thus, interference between all link pairs can be estimated. Our goal is to (i) identify such instances, and (ii) infer the interference behavior during such instances. There are several challenges here that we discuss in the following.

GENERATING UNIFIED TRACE

Traces are collected by deploying several sniffers in the network for each channel to be monitored. The exact location of the sniffers is not important. The idea is to simply have enough sniffers at strategic locations so that a large percentage of frames that were transmitted by every node could be captured by at least one sniffer. Having a large number of sniffers alleviates the problem of positioning them optimally which is a complex problem by itself.

The individual traces from the sniffers are merged to produce a single complete trace with a common time base that will be analyzed. We use the technique proposed in recent literature by Yeo *et. al.* [84] to merge the traces. The basic idea is to look for beacons common to multiple sniffers and synchronize the packet timestamps in accordance with the timestamps of these beacons, so that the final merged trace has a uniform time base.

It has been argued recently that such unified traces may suffer from two major problems – possibility of missing packets due to collisions or packet losses at sniffers, and timing errors due to clock synchronization errors [73]. These problems may render the unified trace incomplete and incorrect, thus jeopardizing its applicability for network analysis. For the first issue, a technique of inferring missing packets has been suggested in [61] that can be used to complete the trace to a large extent. Even if the trace is incomplete, if it carries the same statistical relationships as a complete trace would, then our method should still be effective. For the issue of timing problems, [73] shows that the drift between AP clock and sniffer clock is significantly large even within a single beacon interval of 50ms.

Inaccuracy in timestamps can significantly affect our method, as will be apparent later. However, we expect that each sniffer would have a large number of unique APs it can hear beacons from. The frequency of occurrence of such common beacons between traces would be much higher than once every beacon interval, and so the packet timestamps will be synchronized at much smaller time scales. This should reduce the timing errors as the clock would be adjusted before the clock drift becomes too large.

Regardless, it is indeed a challenging problem to create a reasonably complete and accurate unified trace for analysis. In [73], a metric has been proposed to measure the quality of a unified trace in terms of its completeness. It can aid our method as it is possible to choose only parts of the trace for analysis that has a high score for this metric.

LIGHTLY LOADED NETWORK

Typical hotspots and WLANs are usually lightly loaded on average. Note that several measurement-based papers [40, 72, 39] that have highlighted performance artifacts in 802.11, have specifically chosen highly congested periods (e.g., conference and meetings) to perform the study. However, a monitoring framework cannot depend on the use of such selective periods. For example, it is appropriate to monitor a departmental WLAN during regular usage, so that interference information can be inferred, that can help design a better resource management strategy for periods when it could be congested, for example. This is in contrast to waiting until a heavy usage period occurs, like a large meeting, to learn the interference behavior. Short lived TCP flows (for web access, for example) are typical in WLANs. There could be only few instances when flows are simultaneously active for a given link pair. It is important for the inference mechanism to be robust even when a small number of such samples are available in the collected trace.

UNKNOWN LOAD

The monitoring infrastructure cannot look at the packet queues of the transmitters and does not know when a packet captured in trace was indeed ready for transmission. Interference modeling is fundamentally hard if the offered load is not known. To see this, assume that frames from two senders alternate in the merged trace during an observation period, and no two frames overlap in time. This could indicate that the senders interfere. However, it is also possible that they do not interfere and just happen to transmit in an alternate fashion following a specific packet arrival pattern from the upper layer. Analysis of inter-packet times, however, can provide certain confidence – a strategy we will utilize. For example, if the inter-packet times are such that they could be produced by backoffs, this increases the confidence that the two transmitters indeed interfere and are carrying saturated loads for the period of observation. But this requires accurate timing analysis.

On the other hand, simpler methods are possible if saturated periods can be correctly identified. For example, one can use a moving time window on the merged trace and look for window positions where two transmitting nodes share the available bandwidth within the window to the same extent that two saturated interfering senders would. If such instances are found, then the two nodes can be declared interfering. However, choosing the correct window size is a difficult problem. A large window will rarely get saturated, while a small window will contain too few frames to provide enough statistical confidence.

USE OF STRAIGHTFORWARD HEURISTICS

Straightforward heuristics have limited ability in inferring interference from packet traces. The argument in the previous subsection points out one such issue, as offered load is typically unknown and searching for saturated portions in the trace can be hard. Similar other heuristics are hard to design as well. For example, two packet transmissions overlapping in time may indicate that the two respective senders do not interfere. However, concluding that these senders are non-interfering from few such instances may be inaccurate. This is

because the reason of this packet overlap may be due to backoff intervals counting down to zero at the same instance. Another reason could be that the interference between such senders is probabilistic. Thus, sufficient statistics is needed to develop an accurate estimate.

4.3.3 APPROACH

Thus, to determine interference relationships in the network links, one needs a rigorous statistical modeling approach, instead of relying on heuristic-based trace analysis. In the next section we develop such a rigorous approach based on the well-known Hidden Markov Model (HMM) [68]. The basic idea is to model the sender-side of the interacting node pairs in the network via a Markov process based on the MAC layer operation of 802.11. The parameters of this process (essentially the state transition probabilities) depend on their interference relationship (specifically, deferral probability). These parameters are determined from the observed trace using standard methods. These in turn estimate the deferral probabilities.

It turns out that rigorous modeling is only needed to determine the sender-side interference. The receiver-side interference results in collisions that are easily detectable, as they result in retransmissions.³ Retransmitted packets are identified by the set ‘retransmit bit’ in the frame header. A retransmitted frame, say R , can be correlated back to the prior frame, say P , that has not been received correctly. Any frame S from a different sender overlapping with P must be the cause of collision. If no such P exists, the packet loss is due to wireless channel errors rather than collisions. Because of the probabilistic nature of packet capture, sufficient statistics need to be built up to determine receiver-side interference. This is because frames like S and P even when overlapping, may not result in a collision sometimes. Thus, the fraction of times they collide would determine the probability of receivers-side interference.

³For unicast transmissions only. But unicasts are much more frequent relative to broadcasts in a real network packet trace.

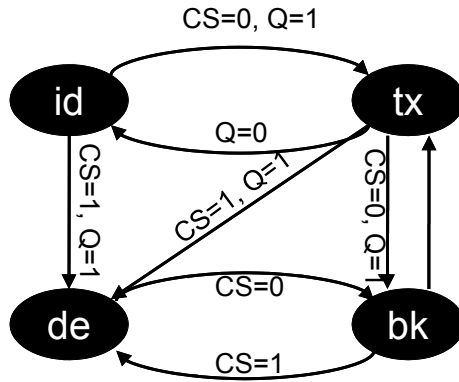


Figure 4.1: State transition diagram for a single sender. $CS = 0$ ($CS=1$) means that the carrier is sensed idle (busy). $Q = 0$ ($Q = 1$) means that the interface packet queue is empty (non-empty).

Prior measurements have, however, shown that collisions are rare even in congested environments [61], indicating perfect or overly conservative carrier sensing in most WiFi hardware as well as lack of synchronized transmissions. Our own experience has also been similar. Since receiver-side interference is easier to detect and also rare, we discuss only the sender-side interference in the rest of this chapter.

4.4 HIDDEN MARKOV MODEL FOR 802.11 MAC INTERACTIONS

A hidden Markov model (HMM) [68] consists of a system modeled as a Markov process with unknown parameters, where the states of the Markov process are not directly visible, but some observation symbols influenced by the states are visible. There are standard methods [68, 25, 15] to learn the unknown parameters (such as the state transition probabilities of the Markov process) using the observation symbols. HMMs have been used in various machine learning fields such as pattern, speech and handwriting recognition. We

will be using the HMM approach for inferring sender-side interference relations between pairs of senders in an 802.11 network.

4.4.1 MARKOV CHAIN

The 802.11 MAC protocol can be modeled as a Markov chain for each sender [16, 49]. An 802.11 sender, say X , resides in one of the following four states - ‘idle,’ ‘backoff,’ ‘defer,’ and ‘transmit.’ In the idle state, the sender does not have any packet to transmit (interface packet queue empty). In all other states the sender has at least one packet to transmit. In the backoff state, the sender is backing off, waiting for its backoff countdown timer to expire. In the defer state, the sender is sensing carrier to be busy and it is thus ‘deferring’ to another transmission. In this state, the backoff timer, if already started, is frozen. In the transmit state, the sender is actually transmitting a frame. These four states captures the essence of the 802.11 MAC protocol. We are intentionally ignoring DIFS to keep the chain description simple.

Let us call the 4 states id , bk , de , and tx , respectively for brevity. At a high level, the 802.11 MAC works as follows. The sender remains in the id state until it has a packet to transmit. When it has a packet to transmit, it senses carrier. If carrier is idle, it enters the tx state. See Figure 4.1. If carrier is busy, it enters the de state. It comes out of de when carrier is turned idle. It then goes to the bk state, chooses a random backoff interval, and then goes to the tx state once the countdown of the backoff timer is complete. If the sender senses carrier busy while in the bk state, it must defer the transmission. It then goes into the de state, from which it comes back to the bk state once the carrier is idle again. The backoff countdown timer is frozen while in the de state. Thus, the sender completes the remaining backoff time when it comes back to the bk state.

After the transmission completes in the tx state, the sender goes back to the id state, if it has no other packet to transmit. Otherwise, it goes back to the bk state after choosing another random backoff interval. The state transition probabilities between bk and de

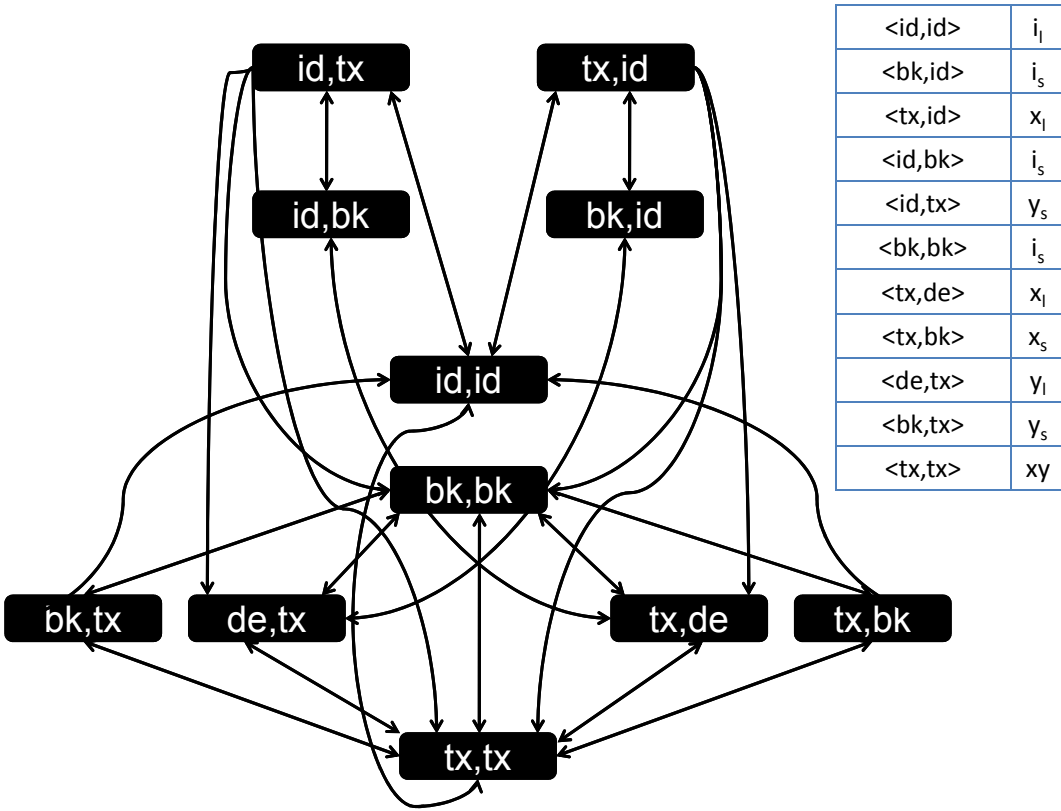


Figure 4.2: Markov chain modeling the combined MAC Layer behavior of two nodes (sender side only). Note that some arrows are bidirectional.

depend on the state of other nodes (i.e., transmitting or not) in the network, and the deferral probabilities between the sender and these nodes. Similar argument applies for the transition probability from id to de and tx , and transition probability from tx to de and bk .

Since the transmissions from other nodes impact the state transitions for a given node, a combined Markov model needs to be considered to get a complete picture of the network behavior. Here, each state is a tuple consisting of states of individual nodes. Such a Markov chain would lead to a state space explosion with exponential number of states, and would thus be intractable. Since our focus in this work is on determining the pairwise interference relationships, we can restrict ourselves to the consideration of a combined Markov chain for only a pair of nodes, say X and Y . Each state in this Markov chain is a 2-tuple consisting of the states of X and Y . For example, the state where X transmits and Y defers would

be $\langle tx, de \rangle$. There could be 16 possible states in theory. However, 5 of them are not legal (e.g., $\langle de, de \rangle$, $\langle de, bk \rangle$ etc.⁴), leaving 11 possible states. See Figure 4.2 for the combined Markov Process.

In this Markov chain, the state transition probability between certain states depends on deferral probabilities between X and Y . For example, from state $\langle bk, bk \rangle$ to state $\langle tx, de \rangle$ or $\langle tx, bk \rangle$ would depend on deferral probability of Y with respect to X . To see this, assume that Y carrier senses X perfectly. Then when X moves from bk to tx state (i.e., starts transmitting as soon as the backoff interval is over), Y must also move from bk from de as it defers to X 's transmission by freezing its backoff countdown timer. If instead Y never carrier senses X , it will remain in the bk state.

Note again that this combined Markov chain is specified for a node pair only, as we are interested in pair-wise interference. This chain can be repeated for all pairs to determine the sender-side interference between all node pairs. When considering a particular pair, we filter out the packets of just the two senders for analysis, and ignore the other packets. This may cause an active node to appear idle for certain periods of time if the node defers for a third node's transmission. While this may result in our method missing out on an opportunity to interpret the interaction between the particular pair as interfering or non-interfering, it is important to note that this does not create any incorrect interpretation. Recent studies [61] show that instances of 3 or more nodes simultaneously being active is much less than instances of only a pair of nodes being active. Thus, we should get enough instances of just a pair of nodes being active in a long trace. An alternate but computationally intensive method could try to identify portions of the trace where only the senders in a node pair being considered are active. Another important observation is that the Markov Process assumes that both nodes strictly follow the 802.11 protocol. Any MAC layer misbehavior may cause the Markov chain to change and may induce errors in our analysis.

⁴Note that this Markov chain assumes only two nodes X and Y interact. Thus, for example, the state $\langle de, de \rangle$ is not possible as both nodes cannot defer at the same time.

4.4.2 OBSERVATION SYMBOLS

As we do not know the interference relation yet, the state transition probabilities of the combined Markov chain is unknown. Also, the states of this Markov chain are not directly visible in the packet trace. We thus need to map each state in this Markov chain to an observation symbol obtained from the trace that can be used to learn the state transition probabilities. There are four possible observation symbols in the trace depending on whether X or Y transmits:

i : neither X , nor Y transmitting.

x : X transmitting.

y : Y transmitting.

xy : both X and Y transmitting.

Each state in the Markov chain can be mapped to one of the four symbols above. This mapping is not unique as more than one state can map to the same observation symbol. For example, both states $\langle id, id \rangle$ and $\langle bk, bk \rangle$ map to the symbol i . Similarly, both $\langle bk, tx \rangle$ and $\langle de, tx \rangle$ map to symbol y . The difficulty here is that backoff cannot be distinguished from defer or idle periods. This ambiguity can be reduced by using a heuristic that exploits the time duration of various observation symbols. This is elaborated below.

A backoff interval in 802.11 comes from a random process and can last for integral number of slots ($20 \mu s$ in 802.11b). Also, the maximum backoff interval is bounded (31 slots in the first backoff stage⁵). While not impossible, it is very unlikely that a defer or idle period will be within this bounded interval and also last for exactly an integral number of slots.

⁵Only the first backoff stage is relevant, as we are only concerned about the sender-side interference here. In any case, the backoff stage can be identified by observing the number of retransmissions in the merged trace.

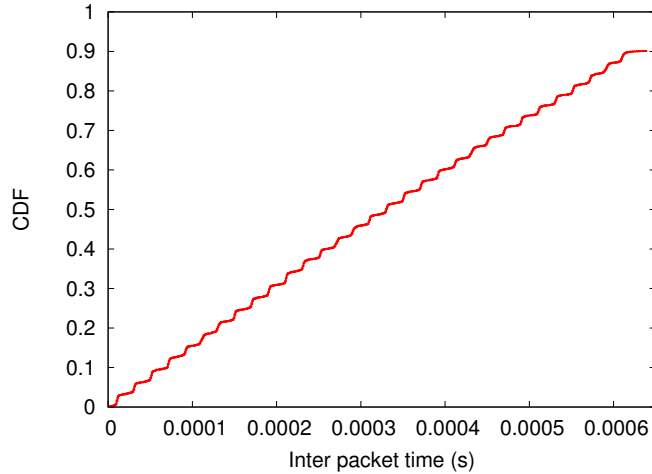


Figure 4.3: CDF of observed inter-frame times in a recorded trace for a single saturated sender.

This strategy to distinguish between backoff and idle/defer periods requires highly accurate clocks (within few microseconds). Without any specialized technique the experimentally observed accuracy is not sufficient. See Figure 4.3 for a CDF of observed inter-frame times in a recorded trace for a single saturated sender. Ideally, one would expect a perfect staircase pattern (vertical risers and horizontal steps) with 32 steps. While 32 steps are visible, the pattern is wavy due to clock errors.

We thus use a weaker heuristic in this work that does not require strong clock accuracy. We assume that defer/idle periods are always longer than 31 slots and backoffs are always equal or shorter. This, however, introduces errors when airtime of an 802.11 frame is less than 31 slots ($620 \mu\text{s}$ for 802.11b⁶). This also introduces errors for very small idle times. With these sources of error, the results in the next section provide only a lower bound on the accuracy obtainable by the base technique. In our future work, we will explore possibilities of using accurate timing information to remove these sources of inaccuracies.

With the above weaker heuristic, each observation symbol can be of two types. The symbol i can be either i_s or i_l , corresponding to short (≤ 31 slots) and long (> 31 slots)

⁶This means TCP packets with payload less than 400 bytes in 802.11b to give the reader an idea.

respectively. According to the heuristic, i_s is most likely output by $\langle bk, bk \rangle$ state, while i_l is most likely output by $\langle id, id \rangle$ state, for example. Similarly, the symbols x and y can be either x_s and x_l , and y_s and y_l , respectively. Figure 4.2 shows the observation symbols for each state.

Each packet in the merged packet trace consists of a timestamp for when the packet was received at the sniffer, the id of the sender, size of the packet, and the rate at which it was transmitted. This information is parsed to obtain the sequence of above observation symbols from the trace. Based on this sequence, we use the following technique to learn the state transition probabilities of the Markov chain, that in turn will provide the probability of interference between the senders.

4.4.3 FORMAL SPECIFICATION AND LEARNING

We now provide the complete formal specification of the HMM using standard notations [68]. The HMM consists of the following:

- Set S of N states, where $N = 11$. S is given by:

$$S = \{S_i\} = \{\langle id, id \rangle, \langle bk, id \rangle, \langle tx, id \rangle, \langle id, bk \rangle, \langle id, tx \rangle, \langle bk, bk \rangle, \langle tx, de \rangle, \langle tx, bk \rangle, \langle de, tx \rangle, \langle bk, tx \rangle, \langle tx, tx \rangle\}.$$
- Set V of M observation symbols, where $M = 7$. V is given by: $V = \{i_s, i_l, x_s, x_l, y_s, y_l, xy\}$.
- Matrix A of state transition probabilities, indicated by $A = [a_{ij}]$, where a_{ij} is the transition probability from state S_i to S_j . This matrix is unknown at the outset and will be determined. Note that some state transitions are invalid and such a_{ij} is set to 0. Such transitions are absent in Figure 4.2.
- Matrix B of observation symbol probabilities, indicated by $B = [b_{jk}]$, where b_{jk} is the probability that the observation symbol is v_k for state S_j . In our case, observation

symbols are deterministic for each state. But they are not unique. The mapping from states to symbols are shown in a table within Figure 4.2.

- Vector π of the initial state distribution, indicated by $\pi = [\pi_i]$, where π_i is the probability of initial state being S_i . We use $\pi_i = 1/N$ for all $i, 1 \leq i \leq N$.

The above defines the HMM, $\lambda = (A, B, \pi)$. The packet trace provides the observation sequence $O = O_1, O_2, \dots, O_T$, where each observation $O_i \in V$, and T is the number of observations in the sequence.

Given the above HMM λ and the observation sequence O , we wish to learn the model parameters $\lambda = (A, B, \pi)$ that maximize $P(O|\lambda)$. This is a difficult problem, and there is no optimal algorithm for it. We can, however, use the expectation-modification (EM) algorithm, which is an iterative method to determine λ , such that $P(O|\lambda)$ is locally maximized. The EM algorithm alternates between an expectation (E) step, which computes the model parameters most likely to produce the observation, and a modification (M) step, which computes the maximum likelihood of model parameters across multiple E steps [25]. We use the Baum-Welch method, which is a type of EM algorithm, based on the forward-backward algorithm developed by Baum *et. al.* [15]. The method insures that in every estimation step, we find a model which is more likely to produce the observation. Thus, if we estimate the parameters of the model λ to get $\bar{\lambda}$, then $P(O|\lambda) \geq P(O|\bar{\lambda})$.

While using the Baum-Welch method, we do not readjust the parameters B and π in the model λ . We initialize the state transition probabilities such that equal probability is assigned to all the outgoing valid transitions from each state. This ensures that there is no initial bias in the model towards interfering or non-interfering pair of nodes. This aids in quick convergence of the method. We also need to use the scaling technique in the procedure [59]. This is needed as we deal with very long sequences of observations. and continued multiplications of certain fractions creates problems with numeric accuracies.

4.4.4 LEARNING DEFERRAL PROBABILITY

Transitions into any state with a defer component (i.e., states such as $\langle de, * \rangle$ and $\langle *, de \rangle$) indicate interference. Thus, our task is to evaluate the total probability of transition into such states. Let us denote the set of these states as D , where $D \subset S$. Let us denote by P the set of states that have transitions into the states in D , according to the state transition diagram in Figure 4.2. If $\Pi = [\Pi_i]$ is the stationary (steady state) distribution of the states, then the deferral probability is the conditional probability that the chain enters a state in D given that the chain is in a state in P . Thus, the deferral probability is given by,

$$\sum_{\forall i.s.t.S_i \in P} \sum_{\forall j.s.t.S_j \in D} a_{ij} \Pi_i.$$

Once the transition probabilities $A = [a_{ij}]$ are learnt, $\Pi = [\Pi_i]$ can be determined as $\Pi = \lim_{n \rightarrow \infty} \pi A^n$. The convergence is guaranteed as A is a stochastic matrix. The above expression to compute deferral probability assumes a symmetric link between a node pair. Links may be asymmetric in reality, and the above expression can be easily modified to consider a one-directional deferral probability. However, in this work, we evaluate only the bi-directional, and hence symmetric deferral probability.

4.5 EVALUATION

We evaluate the effectiveness of the HMM-based approach by experiments and simulations. The first set of experiments serves as micro-benchmarking. Two senders and broadcast traffic are used to specifically evaluate the sender-side interference using carefully controlled load. The degree of interference is varied by repositioning the senders. The second set of experiments are used to study the real-life behavior in our departmental WLAN. Here, realistic TCP download traffic was used to drive the experiments. More elaborate, network-scale evaluation is done using an ‘extended’ ns-2 simulator [7]. This extension is meant to impart realism in the physical layer and interface behavior in ns-2. This extension is based on our recent work and has been validated by real testbed experiments [44].

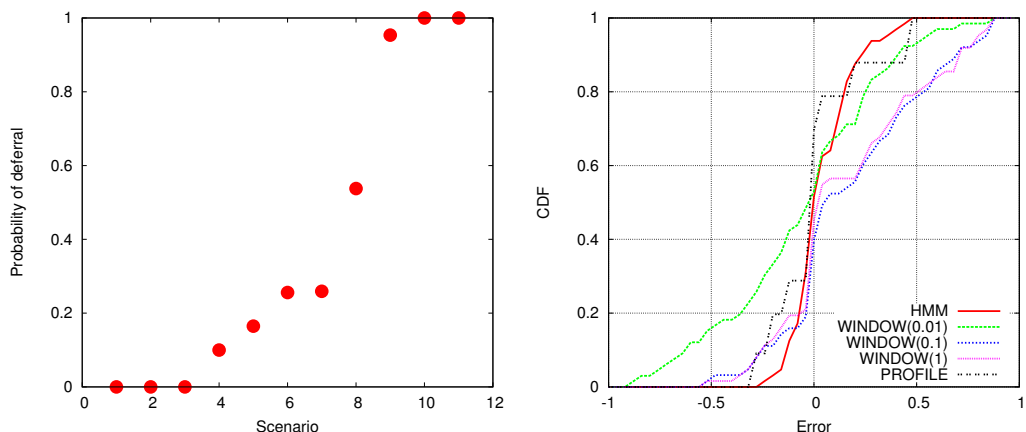
4.5.1 COMPARISON POINTS

For performance comparison, we use two other methods to infer interference.

Profile based method (PROFILE) This technique is specifically based on [71, 49]. This involves understanding the relation between the received signal strength (RSS) and the probability of deferral. This is done by using a pair of nodes to collect a lot of measurements for the above two variables and then creating a profile for the specific interface card used. This needs to be repeated for all different cards used in a network. Once the profile for a specific card is known, the probability of deferral between two nodes can be obtained by measuring the average RSS values between them and doing a lookup on the profile. Note that this technique is based on active measurements and is thus expected to be quite accurate. We use this technique as a benchmark.

Moving window based method (WINDOW(t)) This technique is an example of a simple heuristically based approach that may not perform well without extensive parameter tuning. See the discussions in Section 4.3.2 in this regard. This technique involves using a moving time window of size t seconds to scan the combined packet trace, such that we consider only the packets in the window at a time. For each window position, we try to infer if the nodes interfere or they do not by analyzing their throughputs during the window (see below). Finally, we use the ratio of the number of window instances where the nodes interfere and the number of window instances where they do not to obtain the probability of deferral. Specifically, we use the following technique:

- Throughput of each node in the window being considered is calculated. If the throughput is equal to or greater than half the capacity at the bit-rate of the sender for both nodes, then the window is considered saturated, otherwise the window is considered unsaturated.



(a) Measured probability of deferral for different scenarios. (b) CDF of error in estimating probability of deferral.

Figure 4.4: Combined performance results for 11 chosen scenarios for two node experiments.

- Interference can be inferred only for saturated time windows. A saturated time window is marked non-interfering if the throughput of at least one of the senders is equal to or greater than half the capacity. It is marked interfering otherwise.
- Probability of deferral is the fraction of saturated time windows that are marked interfering.

4.5.2 MICRO-BENCHMARKING WITH TWO NODES

We use a two-sender, two-sniffer scenario here. Each sniffer is co-located with a sender to guarantee that all frames are received. In fact, we use just two machines are used for these experiments, each with two 802.11 radios, where one radio acts as the sender, the other acts as the sniffer.⁷

⁷We use a desktop machine running Linux with two PCI based 802.11 cards, and a laptop running Linux with a miniPCI and a PC based 802.11 card. All the cards are based on Atheros chipsets and we run the popular MadWiFi [4] driver on both the machines.

For the experiments, all the four radios are put on the same channel. The choice of channel is immaterial. The sender radio is configured in ‘ad hoc’ mode. All experiments are done for 802.11b and by setting the PHY-layer data rate to 11Mbps.

We keep one machine fixed at one location, and relocate the other to various locations in the building to create a range of interference scenarios where the two senders either interfere or do not interfere, or interfere partially. For each scenario, we perform the following measurements. First, we measure the actual probability of deferral between the nodes using the method described in [65]. We let each sender broadcast 1400 byte UDP packets as fast as it can in isolation for a minute, and measure their throughputs in isolation. We then let them broadcast together as fast as they can, and measure their throughputs again. The ratio of the sum of throughputs when the senders broadcast together to the sum of throughputs when the senders broadcast in isolation is defined as BIR , or the broadcast interference ratio. The ‘measured’ probability of deferral is estimated as $1/BIR - 1$.

We also measure the RSS values at each sender when the other sender broadcasts in isolation. This is again done for each scenario. This is used to estimate the probability of deferral using the *PROFILE* method described above. The interface card profiles have been independently done using a method similar to [49].

Now, we do a series of experiments to capture live network traffic so that *HMM* and *WINDOW(t)* methods can be applied. We generate traffic in the following fashion for each scenario. The senders broadcast 1400 byte UDP packets simultaneously for one minute. The offered load is varied from 0.1 Mbps to 6 Mbps in 10 steps. The inter-packet times are chosen from a Poisson distribution. The PHY-layer bit rate is chosen to be 11 Mbps; thus, 6 Mbps for each node means saturated load. Meanwhile, each sniffer captures all the packets it hears in that duration. The packet trace from each sniffer is merged using the techniques described earlier, and this combined trace is used to estimate the probability of deferral using the *HMM* and the *WINDOW(t)* methods. The later is repeated for three different window sizes ($t= 0.01s, 0.1s, 1s$).

We make such measurements for 11 different locations of the laptop, creating 11 different scenarios. The distribution of the measured probability of deferral at different locations is presented in Figure 4.4(a). For each scenario, 10 different values of offered load are used between 0.1 Mbps and 6 Mbps, thus creating 110 measurements for *HMM* and the *WINDOW(t)* methods, and 11 measurements (one for each scenario only) for the *PROFILE* method. The distribution (CDF) of errors (‘estimated’ – ‘measured’ probability of deferral) is plotted for all three methods in Figure 4.4(b). Note that the *HMM* approach is quite competitive with the *PROFILE* method (actually it is slightly better overall for the particular distribution of deferral probabilities). The root mean square error (RMSE) values are 0.165 and 0.208 for *HMM* and *PROFILE*, respectively. The RMSE values for *WINDOW(t)* methods is 0.385, 0.408, and 0.402 for $t = 0.01s$, $0.1s$, and $1s$ respectively. We have noted before, however, that the *PROFILE* method is impractical for analyzing live network traffic and it also requires access to the network nodes.

Overall, *HMM* is quite competitive with *PROFILE*, but requires only passive measurements. The experience with the window-based method is quite variable. It is also quite sensitive to choice of window size.

4.5.3 EXPERIMENTS ON DEPARTMENTAL WLAN

These experiments are done on a departmental WLAN with 7 APs. The WLAN is spread over two floors of a building. See Figure 4.5 for a layout. Two laptops are used as clients. They fetch several large files sequentially via HTTP download for about 20 mins. This simulates real network traffic that are sniffed using 9 sniffers (Soekris [74] single board computers with 802.11 miniPCI cards with Atheros chipset and with external USB flash memory to store packet traces). The sniffers are deployed based on convenience, i.e., near a power outlet and in the rooms that we have regular access to etc. But an attempt was made to keep them as close to the APs as possible. The client laptops are moved around among 7 possible locations, using one location pair at a time. (16 location pairs are tested.) The

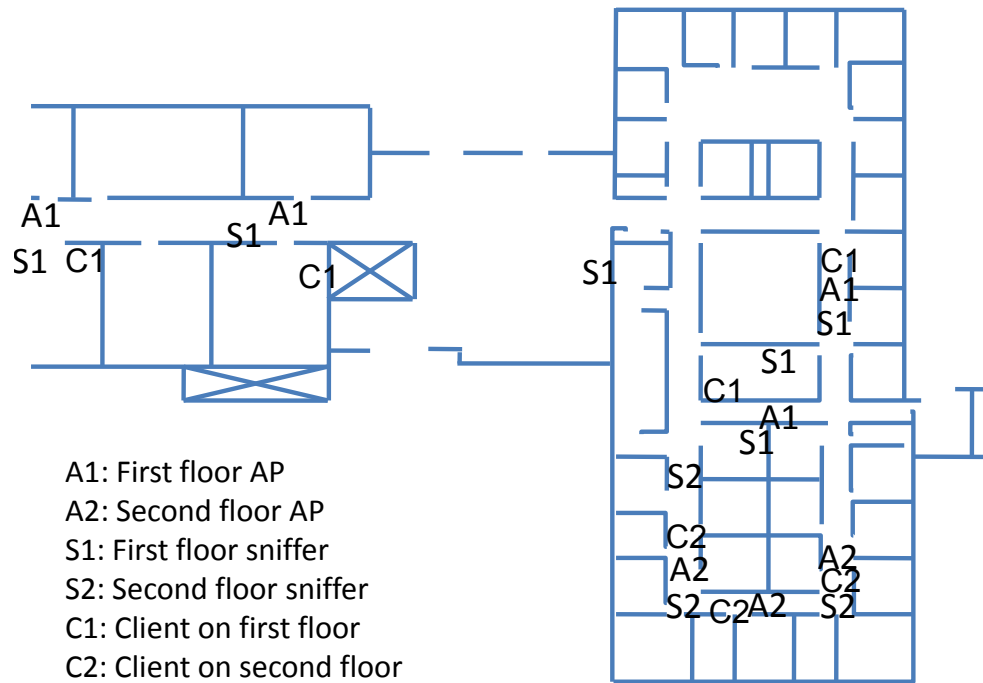


Figure 4.5: Locations of APs, sniffers and clients shown projected on 2D. The nodes are distributed actually over two floors – APs on ceilings, sniffers on floors and clients on floors or on tables.

Figure 4.5 shows all nodes projected on the layout of the 1st floor for brevity. However, the nodes are actually distributed in a 3D space over two floors.

For each location pair, the two laptops associate with two different APs and then simultaneously perform the HTTP download as mentioned before. Unlike the micro-benchmarking experiments, the default auto-rate control with 802.11b is used. Also, the 802.11 frames are now unicast with ACK. RTS/CTS are switched off. For each case, the probability of interference between the pair of download links (AP to client) is ‘estimated’ using equation 4.1. First the probability of deferral (p_d) is estimated using the HMM-based method using the merged sniffed traffic traces from all sniffers. Second, the probability of collisions (p_c) are estimated by observing the retransmissions for overlapped packets as

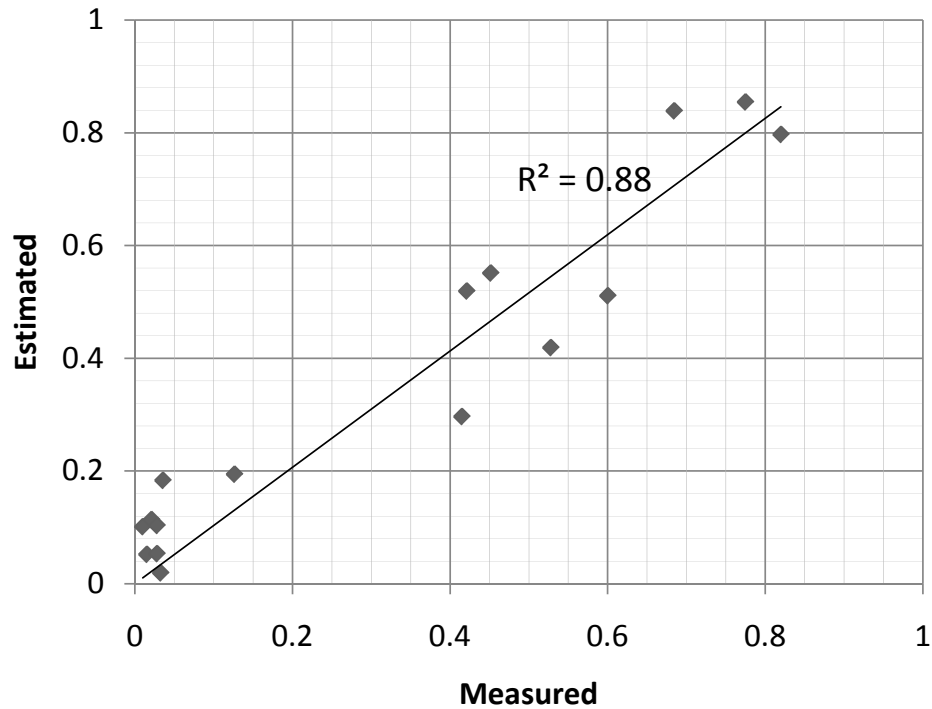


Figure 4.6: Estimated and measured probabilities of deferral for the 16 test cases with the departmental WLAN.

described in Section 4.1.1. However, in all cases retransmissions were rare, typically less than 1% of frames were retransmitted. This is consistent with prior experimental observations [61]. Thus, p_c could be safely ignored with p_d alone determining the probability of interference.

For validation, p_d is ‘measured’ via the BIR method described in the previous subsection. For these measurements, simultaneous saturated UDP traffics on the downlinks are used for about 2 mins. The validation results are shown in Figure 4.6 as a scatterplot. Note the high degree of predictability of the estimation in this real-life experiment. The straight line is the least square fit with the condition that that the line passes through 0. Note that it is very close to the $y = x$ line. The R^2 value for this line is 0.88 showing a good fit.

A careful reader will notice a slight bias at the low end of the deferral probabilities. The HMM method consistently overestimates deferral probability, when the probability is very small. We have also observed this in our micro-benchmarking though it does not show up in the CDF plots. The reason for this is the heuristic we used in our modeling (Section 4.4.2) that defer/idle periods are always assumed longer than 31 slots. When there is little interference, often idle periods could be shorter than backoffs. If they are misclassified as backoffs, the possibility of misclassifying some idle states as defer increases. As discussed in Section 4.4.2, a stronger heuristic using more accurate clocks could address this issue.

We have only reported experiments in a WLAN environment. However, there is nothing specific in the technique related to one-hop network, as the technique estimates interference between link pairs. Thus, it is as applicable to a multihop network as it is for a one-hop network.

4.5.4 SIMULATIONS

To evaluate the performance of HMM technique in a realistic network setting (beyond two node experiments that we did in the previous section) we make use of simulations. Simulations let us create arbitrary topologies and hence interference conditions easily. However, one problem with using simulations is that the physical layer (including interface behavior for carrier sense and packet capture) implementation is often idealized or unrealistic. In our prior work [46] we addressed this issue by extending the popular ns2 simulator with realistic measurement-based models. The models have been validated by experiments showing excellent accuracy. Thus, we are confident that the packet traces generated by the extended simulator is reflective of packet traces obtained in a real network experiments, except missing packets and clock synchronization errors are absent in simulations.

For the sake of completeness, we note here that in [46] the enhancements are done specifically in the following physical layer components – (i) radio propagation model, (ii)

deferral or carrier sense model, and (iii) packet reception model. For (i), models are derived from real measurements in a testbed. For (ii), a measurement-based profile of a testbed is created where every value of RSS is mapped to a deferral probability. Erstwhile, the deferral model in ns-2 consisted of having a carrier sense threshold, which in presence of a constant signal strength would always produce binary interference. For (iii), a similar measurement-based profile is used at the receiver-side to model the packet capture probability with respect to the SINR. These profiles make the interference relations between links non-binary.

We consider 3 scenarios, where 20 nodes are uniformly and randomly distributed in a $200\text{m} \times 200\text{m}$ area, a $150\text{m} \times 150\text{m}$ area, and a $100\text{m} \times 100\text{m}$. These 3 scenarios produce different topologies ranging from sparse to dense. We generate traffic by creating one-hop TCP flows on random feasible links. A link is considered feasible if the delivery ratio on the link is greater than 50%. The difference between the start times of successive flows comes from the Poisson distribution. The duration of each flow also comes from the Poisson distribution. We vary the ratio of the duration and the inter-arrival time of flows to change the load on the network.

We run the simulation long enough to give the opportunity to all pairs of nodes and links to be potentially active together. As the traffic is random, this may not happen for some pairs. For the results presented here (See Figure 4.7), the simulations were run for 180s, the average duration of each flow was 5s, and the average inter-arrival time between flows was varied from 2.5s to 1s, such that the average load in the network varies from 2 to 5 flows.

In Figure 4.7 the results are shown in rows, one row for each network density – low ($200\text{m} \times 200\text{m}$), medium ($200\text{m} \times 200\text{m}$) and high ($200\text{m} \times 200\text{m}$) density. The first plot in each row shows the ‘measured’ probability of deferral for node pairs. The next two plots show the CDF of error performances of *HMM* and *WINDOW(t)* methods similar to the previous experiments. The lightest and heaviest loads used are shown separately as we now

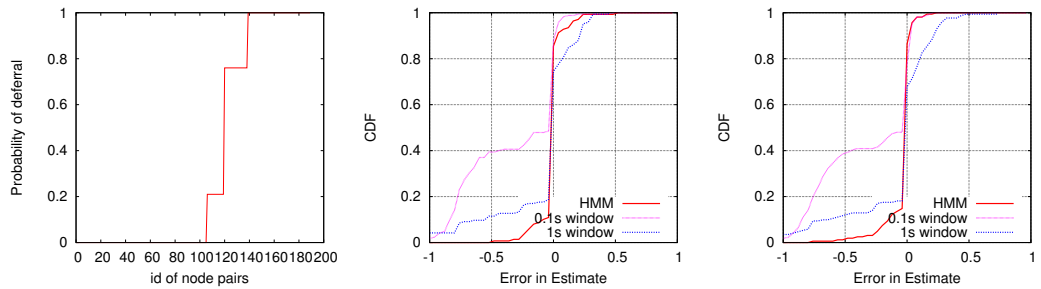
know that the load impacts the performances of these methods. The *PROFILE* approach is not shown here as it would be perfectly accurate in the simulator (as the simulator's deferral model itself uses the same profile model). From the plots note that HMM performs significantly better than the window-based method. Average RMSE value for the HMM method is about 0.09, while the average RMSE value for the window based method is about 0.5. As expected, the accuracy of HMM is better with more interference (dense network). Also, heavier traffic load works slightly better than lighter traffic. Overall, HMM performs within 10% error at least 90% of the times.

4.6 CONCLUSIONS

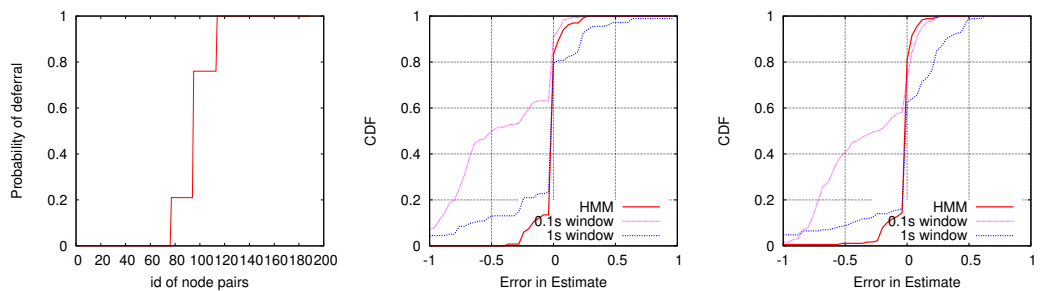
We have investigated a novel machine learning approach to estimate interference in a WiFi network. The technique uses a merged packet trace collected via distributed sniffing. It then recreates the MAC layer interactions between network nodes via a machine learning approach using the Hidden Markov Model. This is finally helpful in inferring interference relationships. The advantage is that the proposed technique is purely passive and thus can work with a live network without any access to the network elements. We have demonstrated via experiments and simulations that the HMM-based technique can provide accuracy similar to existing methods that uses profiling and active measurements directly on network nodes. Simpler, heuristically based passive methods have been shown to perform very poorly.

While the HMM-based technique is able to estimate non-binary interference relations, one shortcoming at this point is that it can infer only pairwise interference and not physical interference. However, this is more of a limitation of our current study and not of the basic approach. The Markov model can be extended from node pairs to node triplets, node quadruplets, and so on. In each step, the number of states increases making the learning process computationally more intensive. However, for all practical scenarios, we do not expect this needs to be extended beyond a handful of nodes at a time. This is because it is

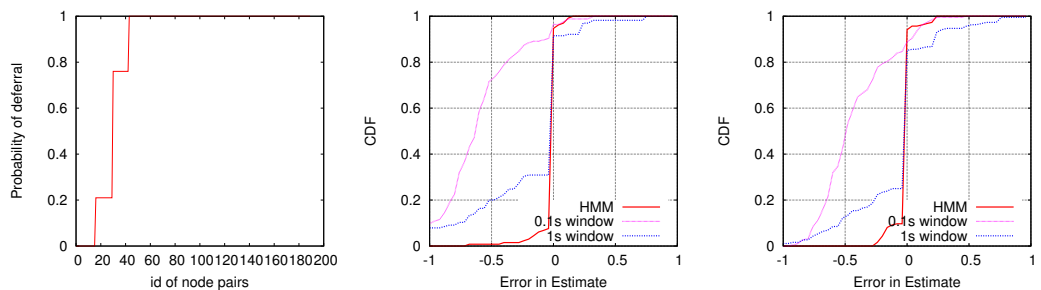
unlikely that a large group of nodes will actually transmit together in a live network. Our future work will extend our approach to physical interference, and will also perform more performance studies with live networks.



(a) Distribution of deferral (low density) (b) CDF of estimation error (light traffic, low density) (c) CDF of estimation error (heavy traffic, low density)



(d) Distribution of deferral (low density) (e) CDF of estimation error (light traffic, medium density) (f) CDF of estimation error (heavy traffic, medium density)



(g) Distribution of deferral (low density) (h) CDF of estimation error (light traffic, high density) (i) CDF of estimation error (heavy traffic, high density)

Figure 4.7: Performance results for 20-node scenarios with the extended ns-2 simulation. Distribution of probabilities of deferral for node pairs and CDF of interference estimates for low and high traffic are shown for three network densities.

CHAPTER 5

VOIP ON WIRELESS MESH NETWORKS

In the previous chapters, we presented various methods to model interference, and its impact on the network capacity. In Chapters 2 and 3, we built methods to estimate the capacity of a link in a 802.11-based network in presence of multiple interferers, first by analytical methods, and then using measurement-based simulators. In Chapter 4, we presented a machine learning approach to estimate interference between links. We now present an application of having a capacity model.

We study the problem of supporting VoIP calls in a wireless mesh network. Specifically, we propose solutions for call admission control (CAC) and route selection for VoIP calls. Call admission decisions must evaluate how the capacity of the mesh network is utilized by the existing calls. We address this issue via a measurement-based modeling effort to model mutual interference between wireless links. The modeling approach evaluates whether capacity constraints (or, required QoS metrics) will be satisfied if a new call is admitted with a given route.

5.1 INTRODUCTION

Voice over IP (VoIP) applications have seen tremendous growth in the recent years. This has led to the emergence of VoIP applications, e.g., Skype, and service providers, e.g., Vonage, Packet8, etc. Recently, with the advent of dual interface cellphones with WiFi interfaces and ubiquitous availability of wireless LANs, the VoIP over WLAN is gaining popularity. In typical scenarios, the footprint of each Access Point (AP) in a WLAN is limited to 250 meters outdoors, and up to 100 meters indoors. For providing wide area coverage

such as in a shopping mall or campus area, the deployment and maintenance of this wired backplane required for connecting a large number of APs becomes a fairly arduous task. This is where the emerging *wireless mesh networks* [10] can be useful. Mesh networks add routing functionalities to the APs of WLAN, thus eliminating the wired backplane, making them easier to deploy. Because of their potential wide-spread use, it is of paramount importance to study methods to implement VoIP services on wireless mesh networks.

SUPPORTING VOIP OVER MESHES

In this chapter, we focus on quality of service (QoS) provisioning issues for supporting VoIP over mesh networks. Specifically, we address two related questions: a) How can we maintain the QoS of VoIP calls over a mesh network and b) How can we improve the capacity of the mesh network in terms of the number of VoIP calls that can be supported? We answer the above questions by solving the *call admission control* (CAC) and *route selection* problems for VoIP calls in the mesh network.

The role of CAC is to determine whether to accept or reject an incoming VoIP call based on the available capacity of the mesh network. CAC is a necessary component of a VoIP service in order to maintain QoS of the ongoing calls while ensuring that calls are not rejected when network capacity is available to accommodate the call. Accuracy of the CAC depends upon how well the mesh network capacity is inferred. This is inherently difficult because of wireless interference: two wireless links in the vicinity interfere to some extent. Interference also leads to MAC protocol inefficiency: two interfering links when active simultaneously often provides less aggregate throughput than when only one of them is active. A result of all these is that any new VoIP call can reduce throughputs (and hence QoS) of many existing calls even without directly sharing any link with them in the chosen route. Thus, call admission decisions must somehow model wireless interferences accurately and must be able to predict the available capacity. This is a hard problem.

Further, routing decisions are closely coupled with admission control. For efficient route selection, one not only has to look for a feasible route (i.e., one that has enough available capacity), but also one must ensure that the choice of routes still leaves enough residual capacity to be able to admit future calls for a given calling pattern statistics. Because of wireless interference, checking for feasibility itself can be computationally intensive. This is because there are exponentially many paths between a source-destination pair, and because of wireless interference, each one of them must be checked in its entirety for feasibility. Thus, practical and effective heuristics are desired for the route selection problem.

CONTRIBUTIONS

With this background we make the following contributions in this work.

- A. In order to develop an effective call admission decision, we develop a *measurement-based* capacity utilization model for 802.11-based mesh network. This model provides the current view of every node's available capacity that needs to be met when admitting a new VoIP call. Using a 802.11a-based testbed, we experimentally demonstrate the effectiveness of this modeling approach in making call admission decisions.
- B. We address the problem of finding a feasible route for a VoIP call while meeting any capacity constraint. We develop a polynomial time solution that can always find a feasible route if one exists. We show via simulations that by discovering feasible routes, we can increase the call acceptance rate by 20% compared to the traditional shortest path routing using hop count metric.
- C. Finally, we demonstrate that if a distribution of calling pattern is known, it is possible to find routes that can improve the VoIP call acceptance rate. We present an algorithm that creates routes by avoiding critical links and results in increasing the acceptance of future calls. We show via simulations that using this routing approach, we can achieve up to 40% increase in call acceptance rate.

5.2 RELATED WORK

Studies focussing specifically on VoIP over 802.11 have considered the delay and loss characteristics under the PCF and DCF modes [81, 36, 82]. A recent work on VoIP over WLAN [33] presents analytical studies on the number of calls that can be supported in a single hop WLAN. The study reports that increasing the payload per frame increases the number of supported calls. Various methods for improving the performance of VoIP in wireless mesh networks have been proposed in [63, 30]. These methods include using path diversity and packet aggregation. Our work addresses a more challenging problem of determining the capacity of a call along a path in a multi-hop mesh network.

Several models have been proposed to compute the capacity of a wireless network. Bianchi proposed a model for determining the capacity of 802.11 in a single cell [16] which has been followed by other similar models as in [80]. In multi-hop wireless mesh networks with given interference and traffic models, the works in [37, 53, 57] formulate a linear program to solve the joint routing and scheduling problem to maximize the capacity of the network. The work in [77] addresses capacity issues specifically for VoIP calls, but it assumes a TDMA based MAC layer. All these works assume some form of scheduling at the MAC layer that is not available with 802.11. An analytical model to compute the end-to-end throughput capacity of a multi-hop path in 802.11-based network has been proposed in [31]. The capacity models proposed above use a somewhat abstract and idealized model of interference, that assumes that interference is binary and is between link pairs only. Further, interference is assumed to be based on physical distances between the transmitters and receivers, simplified radio propagation models, idealized transmitter and receiver characteristics, and so on. In practice, interference is a complex phenomenon as demonstrated in experimental studies in [65, 71], where practical, measurement-based methods are promoted to estimate the interference between 802.11 links.

For multihop wireless networks, several modifications to on-demand routing protocols have been proposed to support QoS for real-time applications [58, 83, 19, 78]. In spirit,

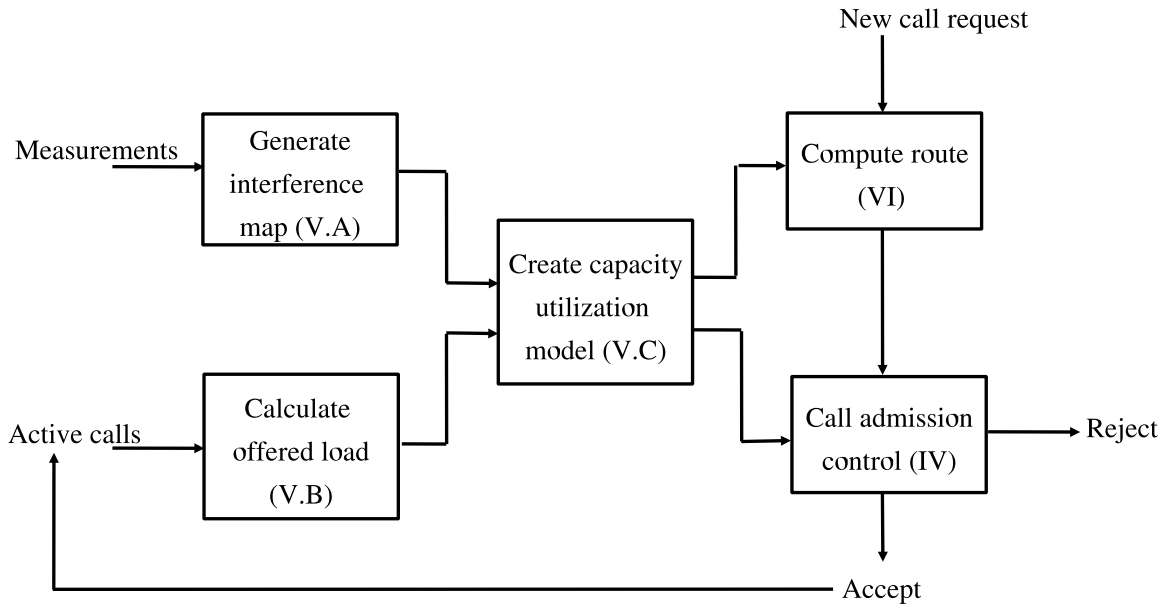


Figure 5.1: Architectural block diagram for the approaches developed in this chapter. The number in the parantheses in each block indicates the section where it is discussed.

these techniques propose or modify an on-demand routing protocol to support QoS. These techniques cannot guarantee that a feasible path will be found if one exists as the proposed protocols perform only neighborhood checks to verify capacity constraints. Furthermore, the above on-demand protocols require exchange of multihop messages to find the route and result in significant call set up time.

5.3 ARCHITECTURE OVERVIEW

In a typical mesh network deployment for supporting VoIP services, a person can make VoIP calls using WiFi enabled phones. Any “internal call” (calls made between clients inside the network), or “external call” (calls made to or from clients outside the network) goes through a central Session Initiation Protocol (SIP) server. The SIP server authorizes the calls and resolves IP addresses and phone numbers, and then the route is established between the clients.

Figure 5.1 shows various system components in our architecture. An interference map is created based on the measurements reported by each mesh node. The interference map models the interference for the given mesh topology. A list of active calls along with their respective routes is maintained. From the list, the current offered load at each node is obtained. The capacity utilization model is constructed from per-node traffic load along with the interference map. This model is updated every time a call arrives or departs, or a new measurement report is received. Using this capacity utilization model, a route for the new arriving call is computed, and the call admission decision is made depending on whether a feasible route is found.

Upon call arrival, the call setup must be done within a few seconds. The call setup consists of SIP authorization, route computation, call admission control, and the actual route setup. SIP authorization and route setup are not studied in this work.

The entire system apart from the node-based measurements is deployed on a central server. Such a *centralized* architecture has several benefits: a) a central server can easily interact with the SIP server, b) minimum functionality at mesh nodes paves the way for deployment of commodity hardware and software platforms that is also easy to upgrade. Many commercial WLAN and mesh network platforms [27, 62] already have a central manager node where the CAC and routing software can be deployed. The centralized view is also consistent with one of the deployment scenarios currently being standardized by the CAPWAP working group [64].

5.4 VOIP CALL ADMISSION CONTROL

A good call admission control design must look at a VoIP specific QoS metric and understand the effect of the network performance parameters such as delay and loss on this QoS metric. A capacity model can then be built based on this understanding.

VoIP QoS Measure: For G.729a encoder, VoIP sends 50 packets per second of 20 bytes each. The *R*-factor, or *R*-score, proposed in [22] is a popularly used QoS metric for VoIP

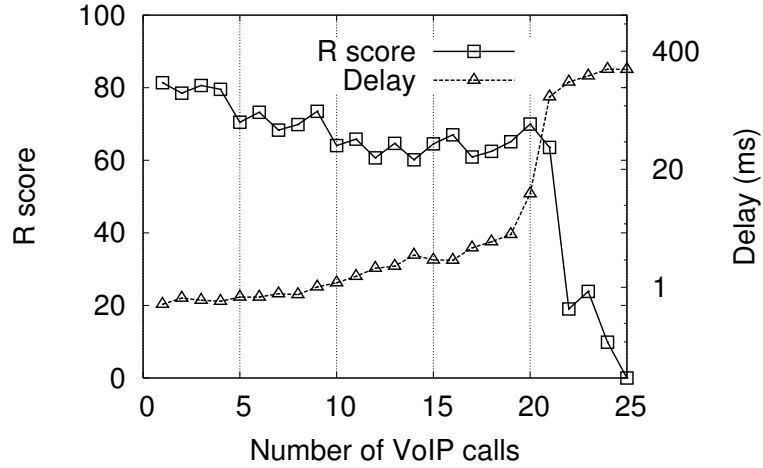


Figure 5.2: On a 2-hop path, the graph shows how R score depends on capacity. Once the capacity threshold is reached delay increases due to queuing delay and R score goes down.

calls. R -score takes into account one-way delay, loss rate, and the type of the encoder. For example, for the G.729a encoder [29],

$$R = 94.2 - 0.024d - 0.11(d - 177.3)H(d - 177.3) - 11 - 40 \log(1 + 10e),$$

where

- $d = 25 + d_{jitter_buffer} + d_{network}$ is the total one-way delay in ms comprising of 25 ms voice encoder delay, delay in the de-jitter buffer (50ms), and network delay;
- $e = e_{network} + (1 - e_{network})e_{jitter}$ is the total loss rate including network and jitter losses;
- $H(x) = 1$ if $x \geq 0$, else 0. R -score should be larger than 70 for acceptable call quality.

Meeting the target VoIP QoS: The quality of a VoIP call is sensitive to delay and loss. The exact dependence is non-linear as given by the R -score formulation above. In order to

maintain a good call quality ($R \geq 70$), the one-way delay should be less than 200ms and the packet loss rate along the path should be less than 5%. Packet loss rate can be reduced by choosing a path consisting of links with a high delivery ratio. Packet transmission delays at each hop is typically within a few millisecond. However, queueing delays can add up.

From elementary queueing theory, the average queueing delay increases with load, but really becomes large when the average load reaches close to the capacity. We demonstrate this in connection with VoIP and mesh networks using an experiment with a 2-hop segment of a 802.11a-based mesh testbed (testbed described later in Section VII). The 802.11a links are operated at 6 Mbps (and thus in theory each link individually can support 42 calls using calculations of [33]). In the experiments we keep adding VoIP calls to this 2-hop segment and record average R -score and total one-way delay. See Figure 5.2. Notice that at the point the queueing delay starts increasing abruptly (around 20 calls), R -score also falls rapidly from around 60-70.¹ This experiment demonstrates that the 2-hop segment can support a maximum of about 20 calls,² a limit admission control protocols must understand a priori.

Capacity utilization: Based on the above observation, we conclude that in order to meet the QoS for a given set of active calls, the load on each node in the mesh network must meet a capacity bound. In order to ensure the above condition, we determine the capacity utilization at every node for a given set of active calls. Specifically, we solve the following problem: *For a mesh network of n nodes, modeled as a graph $G = (V, E)$, and a set of paths for k active calls, $P = \{p_1, p_2, \dots, p_k\}$, find the normalized capacity utilization $c_i, 0 < c_i < 1$, for each node i in the network.* Here, a path is defined as a sequence of nodes. Normalized capacity utilization of a node is the total bits/sec traffic transmitted, received or heard by the node (i.e., the total busy time for the radio medium as perceived by the node), normalized to the nominal link capacity.

¹We relax the acceptable R -score limit to 60 to account for unavoidable errors caused due to random losses on the wireless link.

²Ideally, we should get half of the theoretical capacity of a single link, which is 42 calls.

Call admission decision: The call admission controller is invoked once route computation is attempted for a new call. The route computation is described in the section VI. The route must ensure that the capacity constraint at all the nodes in the network is satisfied (i.e., $c_i < 1$) with this new call admitted on this route, i.e., the route is *feasible*. If no such route is found, the call is rejected. If a feasible route is found, the call is added to the set of active calls and the capacity utilization is recomputed for future use.

5.5 MODELING CAPACITY UTILIZATION

Capacity utilization is modeled by first measuring the amount of interference between nodes and creating an interference map. The individual VoIP call's contribution to any node's capacity utilization can be inferred from the interference map.

5.5.1 INTERFERENCE MAP

Following our recent work on measurement-based interference modeling [48], we characterize interference between a node pair in terms of the *carrier sense factor* or *csf*. For two nodes x and y , csf_x^y (carrier sense factor of “ x ” with respect to “ y ”) is defined as the ratio of the actual transmission rate of x when both x and y attempt to transmit at the maximum possible rate, to the transmission rate of x , when x transmits alone at its maximum possible rate. csf_x^y thus denotes the “normalized transmission rate” for x in presence of y .

Typically, csf_x^y takes values between 0.5 and 1. A value of 0.5 implies that x and y are perfectly within carrier sensing range of each other. On the other hand, a value of 1 implies that x and y cannot hear each other. *csf* between a pair of nodes can be indirectly estimated by using the correlation with the delivery ratio of the link y to x , as well as the signal strength and noise of received packets on the link y to x [71, 48]. This requires just $O(n)$ measurements in a mesh network, where each node takes turn in broadcasting at the maximum possible rate, and the other nodes measure the required parameters, i.e., delivery ratio and signal strength from received packets. *csf* can also be explicitly measured by

doing pairwise experiments for all pairs of nodes in the network. In each experiment, two nodes broadcast at the maximum possible rate, and the number of packets sent out can be measured at each node to get the csf values. This, however, requires $O(n^2)$ measurements. This strategy is somewhat similar to techniques described in [65].

csf estimates (or measurements) between all node pairs define the interference map for the network.

5.5.2 CURRENT OFFERED LOAD

The offered load (l) at each node is normalized with respect to the link capacity of the node. As pointed out before, the maximum number of calls that can be supported on an 802.11a link at 6Mbps is 42. Thus, for a single two-way call on a link, the offered load on each node is $1/84$. For a two-hop call on path A-B-C, the offered load on A and C is $1/84$, while the offered load on the middle node is $2/84$, because it has to forward traffic in both directions. Thus, for any call with a given path, the offered load on the source and destination of the call is $1/84$, while at the intermediate nodes, it is $2/84$. For each node i , the offered load due to all active calls can be added to compute the total offered load l_i , $0 < l_i < 1$.

5.5.3 CAPACITY UTILIZATION AT NODES

The normalized capacity utilization at any node has been defined in the previous section. For brevity, we may not always use the term “normalized.” Note that if capacity utilization of a node is c_i , it means that the unutilized (or residual) capacity of the node is $1 - c_i$. Capacity utilization can be modeled by the following factors.

Actual traffic load on the node: One of the components of the capacity utilization of a node is the actual traffic the node is transmitting. Actual traffic load (t_i) at a node (i) is greater than the offered load (l_i). It is equal to the offered load plus the extra traffic the node has to transmit due to retransmissions caused by packet collisions. Packets will collide at the receiver if the receiver has another transmitter in its carrier sensing range

that is outside the carrier sensing range of the transmitter (hidden terminal phenomenon).³ Thus, a node j is a hidden terminal for i , if for the receiver k , $csf_i^j = 1$ and $csf_k^j < 1$. The fraction of j 's traffic reaching k is $2 \cdot (1 - csf_k^j)$. Let the amount of traffic sent by i to k be denoted as l_{ik} , such that $\sum_k l_{ik} = l_i$. Since the packet transmissions are uniformly distributed over time, the probability of collision of a packet at the receiver can be approximated as $l_{ik} \cdot l_j \cdot 2 \cdot (1 - csf_k^j)$. This amount of traffic must be retransmitted by i . The retransmitted packets may collide again with a smaller probability. So, if we approximate the number of MAC layer retransmissions to just one, and add the extra traffic generated due to all such hidden terminals (j) and all the neighbors k at i , we get the expression, $t_i = \sum_k l_{ik} \cdot (1 + \sum_j l_j \cdot 2 \cdot (1 - csf_k^j))$, if $csf_i^j = 1$ and $csf_k^j < 1$.

Amount of traffic overheard: The amount of traffic the node can overhear is also included in the capacity utilization of the node. This is because the node is unable to transmit during that time, and hence the capacity is utilized for that moment. A node i can hear the traffic of all the nodes j , where $csf_i^j < 1$. If $csf_i^j = 0.5$, the node can listen to all the packets from j . When $0.5 < csf_i^j < 1$, the amount of traffic the node can listen to is $2 \cdot (1 - csf_i^j)$. Thus the amount of traffic overheard at node i , say o_i , can be approximated as $o_i = \sum_{j \neq i} t_j \cdot 2 \cdot (1 - csf_i^j)$.

Consideration of residual capacity: The residual capacity is given by $1 - t_i - o_i$. The residual "usable" capacity is in fact less than this because of possible collision and retransmissions due to hidden terminals. We model this effect indirectly. Assume that the residual capacity is $1 - c_i$. Thus, the node can generate an extra $1 - c_i$ amount of traffic. The extra traffic generated due to retransmissions of this traffic, say r_i , can be given as $r_i = \text{maximum of } (1 - c_i) \cdot (1 + \sum_j l_j \cdot 2 \cdot (1 - csf_k^j))$ if $csf_i^j = 1$ and $csf_k^j < 1$, over all neighbors k . This is similar to the method in the first step.

³Note that the RTS/CTS is not useful in VoIP. This is because VoIP payloads are small (20 bytes), and relatively RTS/CTS would be a significant overhead.

To determine c_i , we add up all these components and equate it to 1. This gives the equation: $t_i + o_i + r_i = 1$. This equation is solved for c_i to get the capacity utilization at each node in the network.

5.6 ROUTE COMPUTATION

For a new call, if a feasible path is found that meets the capacity constraint for all nodes (i.e., $c_i < 1, \forall i$), we can accept the call and use the path to route the call. A question of routing metric arises if there are more than one feasible paths. Conventional link quality based metrics like ETX [23] is not appropriate in this context. The assumption is that only good links are chosen and the interference map-based approach in Section V has already modeled the effect of interference. Instead our goal here is to choose feasible paths that increase the number of supported calls and minimize future call rejections. We first focus on studying the feasibility aspect.

Typically, a feasible route can be constructed by incrementally including links from the network graph and forming a path that connects the source to the destination (note that we are not trying to optimize any path metric here). Any incremental strategy usually results in a fast polynomial time algorithm for discovering a feasible path. However, such an incremental strategy works only if the following condition is true: *when more than one links are determined to be feasible in isolation to carry a certain amount of traffic, they remain feasible when considered together*. In case of wired networks, the above condition is true. Thus, if links are determined feasible, any path in the subgraph containing only the feasible links is also feasible. However, in wireless networks, the above condition is generally not true.

For call admission, we need a fast heuristic to discover a path. We cannot exhaustively explore all paths and check for feasibility, as there are exponentially many of them. We take a different approach. Instead of checking for feasibility on a link basis, we check for feasibility on a sequence of links (path segment) and then string these path segments

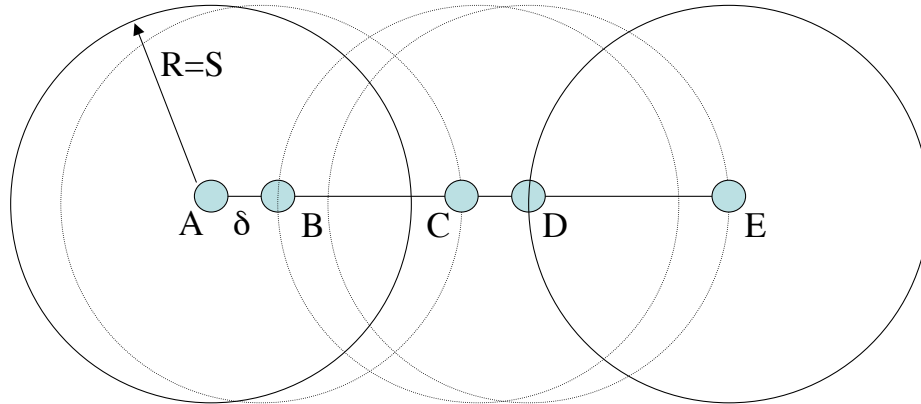


Figure 5.3: Transmission range (R) = carrier sense range (S). In the worst case scenario, two nodes A and E 4 hops apart in a path do not share any node in their carrier sense ranges.

together. A sequence of links is able to capture the capacity utilization of a larger area that reflects the interference region of the intermediate nodes in the sequence. Essentially, our goal is to find the length of the path segments such that if individual path segments are determined to be feasible, so is the path comprising of these path segments. If we consider a unit disk model, we can show that this length depends upon the ratio of carrier sense range (S) to the transmission range (R). We assume that S and R are circular regions around a node, which define the area where a packet transmission by the node can be sensed and received respectively.

We start with the simplest case, where $S = R$. See Figure 5.3. It can be argued using geometry that the nodes such as A and E that are 4 hop away must be at least $2R = 2S$ distance away from each other in the worst case, when δ (distance of A-B) tends to 0. If they are any closer than this, the number of hops will decrease as well.

Nodes A and E that are at least 4 hops apart are guaranteed not to share any node that are within both of their carrier sense ranges. Clearly, this implies that if a path segment $S_1 : (AB, BC, CD)$ is feasible and path segment $S_2 : (BC, CD, DE)$ is feasible, so is $S_1 \cup S_2$. Therefore, if one just considers feasible 3 hop path segments and finds a path using

these segments, one can discover a feasible path. In order to use this approach, we need to mark all infeasible path segments from the original graph. Any path using the unmarked path segments is then feasible. Checking for feasibility of each path segment can be done in $O(k + 1)$ time, where $k + 1$ is the number of nodes in the path segment. The number of such k hop segments in the network can be estimated as $O(nd^k)$, where n is the number of nodes in the network and d the average node degree.

We can similarly show that for the case of carrier sense range twice the transmission range (i.e., $S = 2R$), the hop-wise length of the path segments to be considered is 7. We note that with higher length, the marking of the path segments can become a slower process as the number of k -hop path segments grows as $O(nd^k)$. However, it still remains polynomial time.

For fast computation of route, we consider only 2 hop path segments (i.e., edge pairs) as a heuristic in our evaluations. The penalty we pay for this simplicity is that, occasionally routing may determine routes that are actually infeasible. However the call admission controller determines the infeasibility of such routes and rejects them. In experiments (as reported in the next section), we have found that the chance of finding infeasible routes using this 2 hop technique is negligible. We next present the algorithm for computing feasible paths using this approach.

5.6.1 EDGE-PAIR ALGORITHM

From the given original graph $G = (V, E)$, we construct an edge graph $G_E = (V_E, E_E)$ where an edge in G is represented as a unique node in G_E . There is an edge between two nodes (x, y) in G_E only if (x, y) represents a feasible edge pair in the original graph G . For example, if $(a \rightarrow b), (b \rightarrow c)$ are two edges in G forming a feasible edge pair, then the corresponding nodes $x : (a \rightarrow b)$ and $y : (b \rightarrow c)$ in G_E have an edge between them.

In order to find a feasible route between nodes a and b in G , we consider the node set X and Y in G_E such that $x \in X, y \in Y$ represent edges incident from a and to b in G

```

Feasible_Route(G, a, b)
/* First compute the edge graph containing only feasible edge pairs. */
Compute  $G_E$ , where  $V[G_E] = E[G]$  and
 $E[G_E] = \{((a, b), (x, y)) : (a, b) \in E[G],$ 
 $(x, y) \in E[G], b = x, \text{ and}$ 
 $Check\_Feasibility((a, b), (x, y)) = TRUE\}$ .
Find set  $X \subset V[G_E]$  where  $x \in X$  is edge incident from  $a$ .
Find set  $Y \subset V[G_E]$  where  $y \in Y$  is edge incident on  $b$ .
Add node  $s$  to  $V[G_E]$  and edges  $s$  to  $X$  in  $E[G_E]$ .
Add node  $d$  to  $V[G_E]$  and edges  $Y$  to  $d$  in  $E[G_E]$ .
Find shortest path  $P$  from  $s$  to  $d$  in  $G_E$ .
Convert  $P = \{s, (a, b_1), (b_1, b_2) \dots (b_j, b), d\}$  to
 $P' = \{a, b_1, b_2, \dots b\}$ .
Return  $P'$ .

```

Figure 5.4: Fast heuristic algorithm to find a feasible path in the graph G .

respectively. The set of paths P from $x \in X$ to $y \in Y$ for all x, y forms the feasible path set from a to b . The algorithm is presented in Figure 5.4.

In the above algorithm, the feasibility of an edge pair is determined by using the technique described for CAC in Section IV and V. The current offered load on the nodes in the edge pair is increased assuming the edge pair will lie on a path of an incoming call. The increase in offered load depends on the position of the edge pair in an end-to-end path, because end points generate traffic only in one direction and do not relay traffic. The capacity utilization at all nodes is then recomputed and if the capacity constraint at any node is violated, the edge pair is marked infeasible.

In order to select a path in G_E , we add two virtual nodes s and d to V_E with edges from s to X and edges from Y to d . We compute shortest path from s to d in G_E . This gives us the *shortest feasible path* from a to b in original graph G . In order to choose less loaded paths, we can also assign a weight to each link that is, for example, proportional to the current load along that link and use this metric to compute the shortest path. In the evaluation section, we refer to this extension as *max residual feasible path*.

Note that the above algorithm can be extended to 3 hop or longer path segments. For example, in case of 3 hops, we need to first create G_E with 2 hop segments or edge pairs. From G_E , we construct G_E^2 (a graph with edge pairs as vertices and links between feasible edge triplets as edges) by repeating the same process as used to get G_E from G . This technique is quite efficient, because, before considering the feasibility of a 3 hop path segment, we check the feasibility of 2 hop segments and thus reduce the number of 3 hop segments to be checked.

5.6.2 ROUTING USING CALL STATISTICS

So far, we have restricted our attention to finding feasible paths efficiently and focused less on which of the many feasible paths that might exist should be selected for routing the incoming call. We indeed have provided two simple methods for selection – shortest feasible path and max residual feasible path. However, a potentially better approach for path selection could be to allow more calls to be supported in future. Such an approach is important to VoIP service providers that are interested in supporting as large a call volume as possible while maintaining call quality. The exact sequence of future call arrivals may be unknown; however, an approach can be designed simply based on long-term call statistics, specified in terms of the probability $p(a, b)$ of a mesh node pair (a, b) to be the source and destination of a new call. Such statistics may be available to the service providers collected via long term measurements. Hot-spot nature of certain mesh routers or regions of mesh networks can generate quite skewed distributions that can be exploited in this approach.

A similar idea called *Minimum interference routing algorithm (MIRA)* [52] has been proposed for traffic engineering work in wireline networks. The basic principle behind MIRA is to define a notion of criticality for a given link and select a route that best avoids critical links. For a given source a and destination b , a link is critical if it belongs to the min-cut [28] between a and b . The level of criticality is determined by $p(a, b)$.

Loosely based on MIRA, we propose a route computation algorithm for VoIP calls over mesh network using call statistics. A weight is assigned to each link based on link criticality. The notion of criticality is explained below. Weights are defined such that a route computation becomes as simple as finding a shortest path on the weighted graph after the feasibility has been ascertained. In order to capture the interference properties in a wireless mesh network, we initialize all link weights to zero, and then develop the following weight assignment rule when a new call arrives between nodes s and d .

- *Assign weights to links based on their criticality* – In the first phase, the set of critical links for each node pair except (s, d) is determined. In wired networks, a critical link for a node pair is one which belongs to any one of the min-cuts for that node pair. All the critical links for a node pair can be found by running the Ford-Fulkerson max-flow algorithm [28] just once. This constitutes a critical link set for a node pair (a, b) and is denoted as $C(a, b)$. Since we have a wireless medium, any link which interferes with the critical link should also be a critical link, because adding traffic on that link reduces the maximum flow between the node pair as well. So, we add all the links which have any node which interferes with any of the nodes in the critical links, ($csf < 1$), to the critical link set $C(a, b)$.

Then at this stage, the weight of each link l is given as

$$w_0(l) = \sum_{(a,b):(a,b) \neq (s,d), l \in C(a,b)} p(a, b).$$

- *Add capacity utilization constraint* – In the next phase, the weight of each link calculated in the first phase is multiplied with the capacity utilization at the link. Capacity utilization of a link is the maximum of the capacity utilization at the nodes of the link. The revised weights are

$$w_1(l) = w_0(l) \times \max(c_u, c_v), \text{ where } l = (u, v).$$

- *Make weights non zero* – In the final phase, all links which still have a zero weight are assigned a very small weight, ϵ , such that this weight is not significant enough

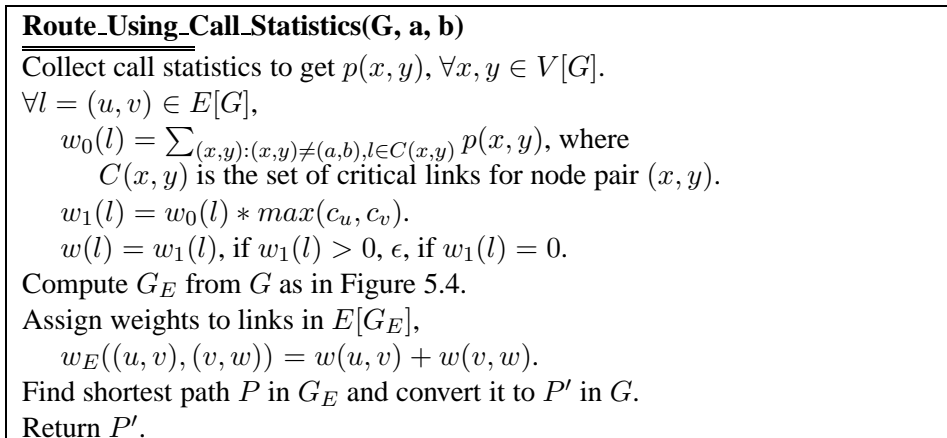


Figure 5.5: Algorithm for routing using call statistics in the network graph G for a call between nodes a and b .

to make the weight of the link comparable to a critical link, or a link with some capacity utilization. A non zero weight is required because the path weight is the sum of link weights, and smaller paths are desired. We chose the value of 0.001 for our experiments. The final link weights are

$$w(l) = w_1(l), \text{ if } w_1(l) > 0; \text{ otherwise } w(l) = \epsilon.$$

With the above link weight assignment, we compute the shortest path on the edge graph (G_E) proposed in the previous subsection. Formal description of the algorithm is given in Figure 5.5.

5.7 PERFORMANCE EVALUATION

Here, we present the results of the evaluation of the capacity utilization model and routing. The capacity utilization model and call admission decisions are evaluated on an experimental testbed. Routing is evaluated on the ns-2 simulator [26], as this evaluation requires a large number of nodes.

5.7.1 EXPERIMENTAL TESTBED

We evaluate our capacity model using a testbed consisting of six identical Dell laptops running Linux 2.6.15, and using Atheros 802.11a/b/g cards and madwifi driver [4]. The laptops are located at different locations in one floor of the NEC Labs building (150' X 120') to create various topologies. 802.11a is used at 6 Mbps for all experiments. Using 802.11a provides us with shorter links so that interesting topologies can be created within a small area. As indicated before, each such link can support 42 VoIP calls. For each topology, an initial experiment is run to generate the interference map by measuring *csf* values, and to find the delivery ratio on each link of the network graph. Static routes are setup between nodes using only those links which have a close to 100% delivery ratio. Calls are generated as a Poisson process with a mean rate of λ calls/sec. The average duration of a call is exponentially distributed with the mean rate μ sec. Calls originate between random source-destination pairs. Since there is no waiting time for the calls, the system can be modeled like an $M/M/\infty$ queue, and the average number of calls in the system at any time is given by λ/μ . In our experiments, we fix λ to 0.2 calls/sec, and vary μ to increase the load in the system. Also, we check the *R*-score of all the active calls for 2 sec intervals and record it for later analysis.

We use three different mesh topologies for the experiments, as shown in Figure 5.6. The solid lines indicate the good links which are used in routing VoIP calls. “Topology 1” is a linear chain where every node can hear nodes only one-hop away. Thus, a node 2-hop away would be a hidden terminal for a node. “Topology 2” is a dense mesh network, while “Topology 3” is a sparse mesh network.

5.7.2 EVALUATION OF CAPACITY UTILIZATION MODEL

We compare our model of creating the capacity utilization graph with a naive model that simply reserves capacity based on the traffic the node generates and the traffic it receives or overhears. In the experiment, we use “Topology 1” (linear chain) that measures the number

Path length	#Calls predicted by naive capacity model	#Calls predicted by our capacity model	Actual #calls supported
1	42	42	42
2	21	19	18
3	17	14	14
4	14	11	10
5	14	10	9

Table 5.1: Number of calls supported in a linear network

of calls that can be supported on a path as we increase the path length from 1 to 5. We then predict the number of calls that can be supported using our capacity model as well as the naive capacity model. Table 5.1 shows that our model estimates the number of calls that can be supported much better. The supported number of calls is separately determined by observing beyond how many calls the average R -score drops below 70. As the path length increases, the naive model keeps overestimating the capacity because it does not consider collisions due to hidden terminals. Our model is much more accurate and predicts the capacity within 10% of the obtained capacity.

5.7.3 EVALUATION OF CALL ADMISSION CONTROL

We use the random calling patterns as described in subsection *A* above to evaluate the effectiveness of CAC. All three topologies are considered. 100 calls are used for each experiment with a) CAC enabled, and b) CAC disabled. We increase the load (λ/μ) on the network from 5 calls to 30 calls. This range includes very low load to very high load (much beyond network capacity). The results for the three topologies are shown in Figure 5.6. For all topologies, median R -scores for all calls and the number of calls that are rejected are plotted against network load (λ/μ).

Note that for all cases, in absence of CAC, the median R -score gets poorer for higher load. However, in the presence of CAC, the R -score is relatively stable independent of load. Also, note that since “Topology 2” is relatively denser, CAC really kicks in at a higher load

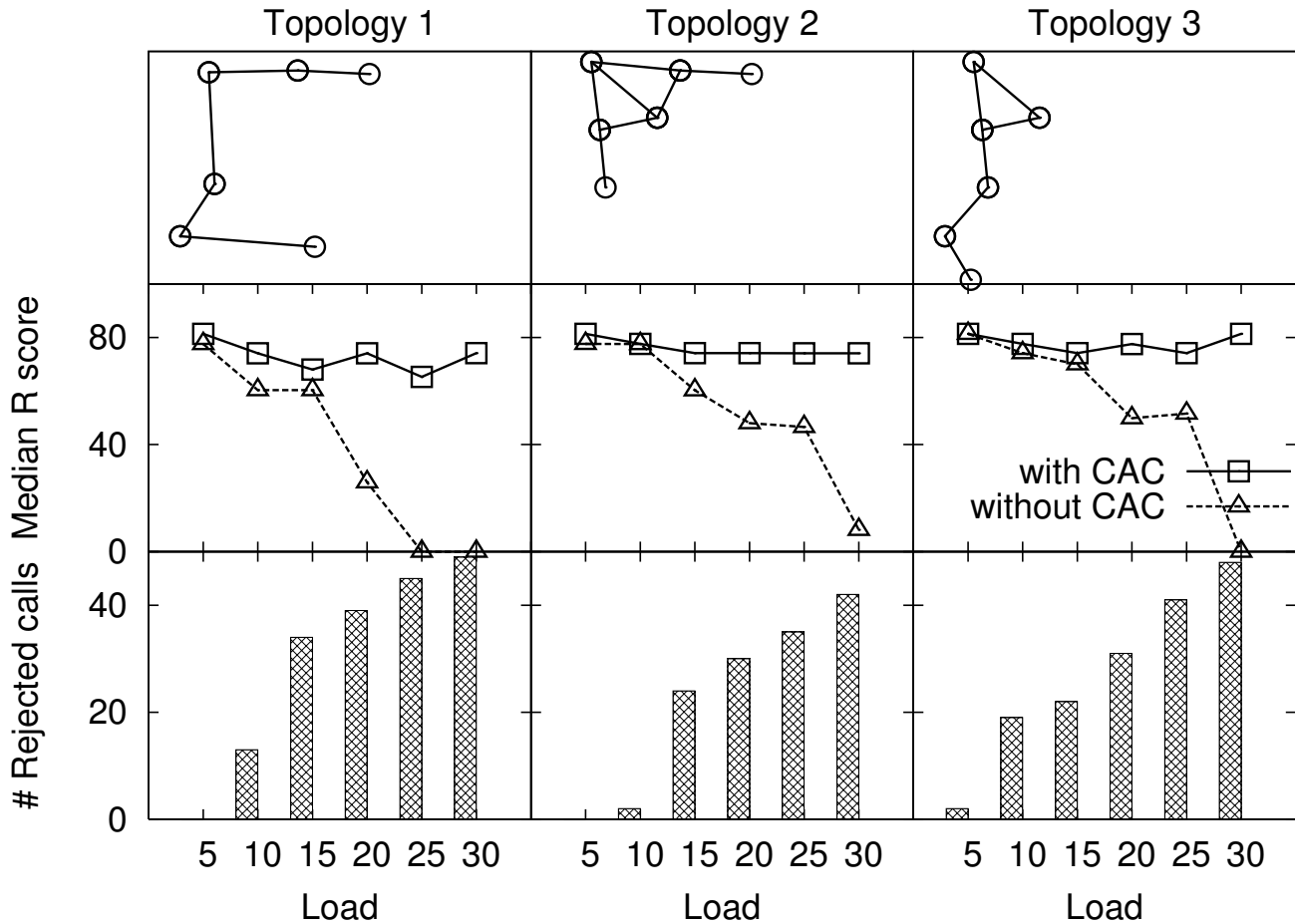


Figure 5.6: Evaluation of CAC for various topologies.

and rejects less number of calls. Also, for any topology CAC does kick pretty much at the same load where we would have R -score degradation without CAC. It does, however, seem that the CAC is slightly aggressive. The reason for this is that for the current model link delivery fractions are assumed to be ideal (100%). While we indeed chose very good links to route packets in these experiments, we still had to cover for less than perfect link qualities by being slightly conservative in accepting calls.

5.7.4 SIMULATION SETUP

The simulation experiments are performed on ns-2 [26] using 802.11b 11Mbps links. The radio propagation model uses the two-ray ground reflection path loss model for the large-

scale propagation model, augmented by a small-scale Ricean fading model [66]. We also patched ns-2 with a realistic packet capture model.

We use two separate topologies for our evaluation. The first is a 13×13 grid in a 4000×4000 square meter area. The radio models are such that the transmission range is about 250 meters and the carrier sense range is set to 550 meters. Thus, every non-boundary node has four neighbors at a distance of 250 meters in each direction. Each vertical or horizontal edge in the grid represents a link and every node can listen upto its two-hop neighbors. The links have no network losses and the *csf* between nodes is either 0.5 or 1. The second topology contains 169 nodes randomly placed in a 2000×2000 square meter area. For call pattern, we consider two scenarios: uniform and skewed. For uniform, the source and destination pair for a call is selected with a fixed probability. For skewed scenario, the source destination pair is chosen based on a weight following the zipf distribution.

A centralized program runs various routing algorithms and determines the routes for the calls. These routes are then fed as static routes in the simulator. If a feasible route is not found, the call is rejected. Calls arrive as a Poisson process $\lambda = 1/6$ calls/sec, and we vary μ to increase the load in the system. Also, we check the *R*- score of all the active calls every 5 seconds, and drop the calls for which the *R* score is less than 70. We run a single long simulation for every scenario, which stops when 2000 calls have been completed.

5.7.5 FEASIBLE ROUTE CALCULATION EVALUATION

We first show that using feasible routes, we can support more calls in the network. Figure 5.7 shows the maximum number of calls that can be supported in the grid as we choose node pairs further away from each other. The metrics used are shortest path (SP) and shortest feasible path (SFP) . We observe that while lesser number of calls can be supported for calls with larger path length, there are more opportunities for finding non-interfering paths, and hence the network can support 10-15% extra calls at saturation.

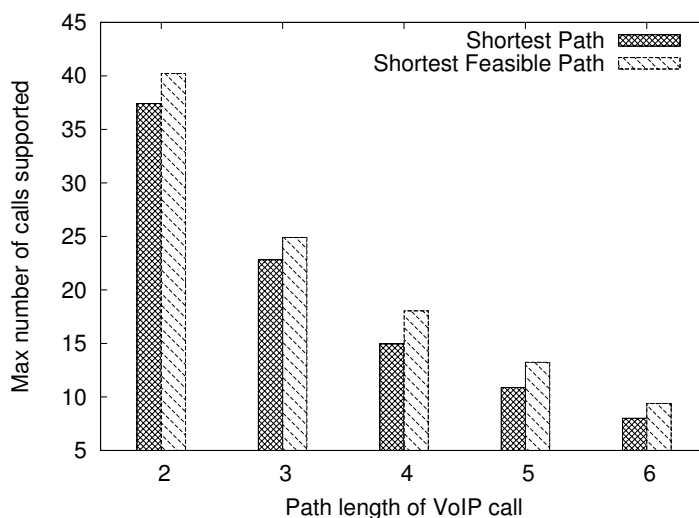


Figure 5.7: Evaluating feasibility improves performance.

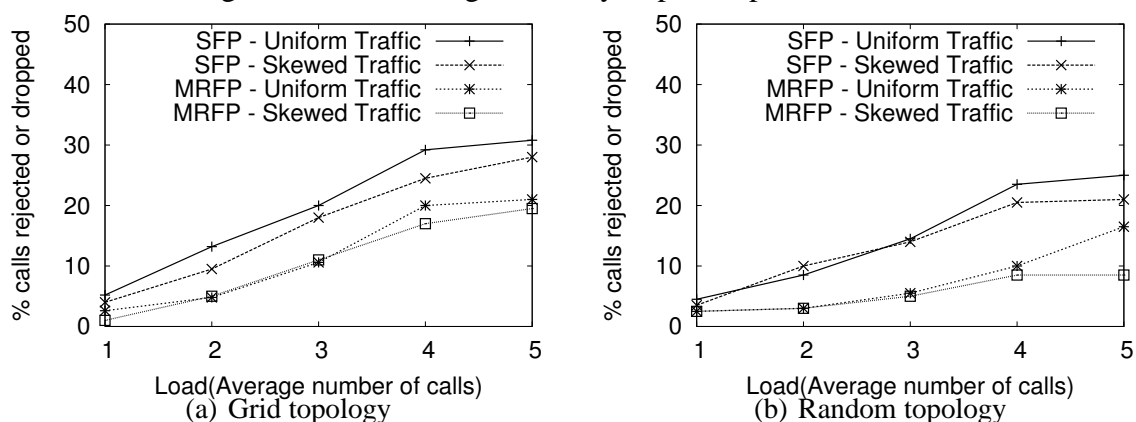


Figure 5.8: Comparison of shortest path (SPF) with max residual feasible path (MRFP) for different load.

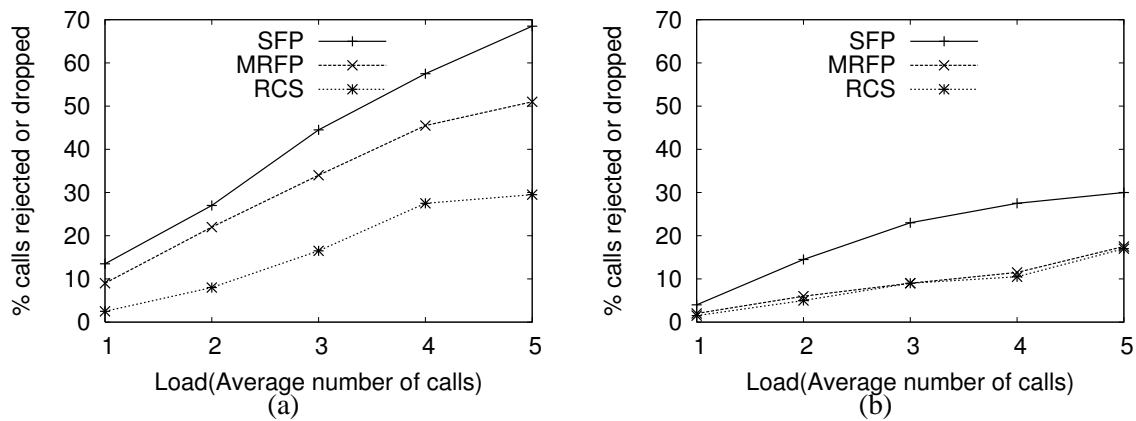


Figure 5.9: Comparison of routing using call statistics (RCS) with shortest feasible path (SFP) and max residual feasible path (MRFP) for a) heavily skewed load (2% nodes are callers), b) lightly skewed load (10% nodes are callers) in Random topology.

Next, we evaluate two routing strategies: a) shortest feasible path (SFP) routing and b) max residual feasible path (MRFP) routing. Figure 5.8(a) and Figure 5.8(b) show the percentage of calls rejected or dropped for each routing scheme in grid and random topologies. For grid topology, drops or rejections by using MRFP reduces by about 30% for large loads when compared to SFP. For random topology, this factor can be upto about 50%.

5.7.6 EVALUATING ROUTING USING CALL STATISTICS

Here, we compare routing using call statistics (RCS) with SFP and MRFP routing. To generate interesting calling patterns where the RCS technique could be beneficial, we assume calls are generated only at hot-spot routers and only a fraction of routers in the network are such hot-spots. This information is provided to the RCS protocol. It is intuitive to see that RCS should perform better when the VoIP traffic is heavily skewed. When the traffic is fairly balanced, all the node pairs have the same weight and if the network topology is also uniform, all the links in the graph get similar weights based on the criticality. RCS degenerates to MRFP routing in such cases.

We present results on a random topology. A grid is not favorable to RCS due to its uniformity. We present results for two cases – 2% and 10% hot-spots, representing a heavily skewed and a lightly skewed traffic pattern shown in Figures 5.9(a) and 5.9(b). As expected, RCS drops lesser number of calls in a heavily skewed traffic pattern, but similar to MRFP routing in lightly skewed traffic.

5.8 CONCLUSIONS

We have addressed two important questions in running VoIP on wireless mesh networks. First, maintaining QoS means that call admission control must be performed. However, without any reasonable model of multihop capacity of the network, the admission decisions cannot be taken. We have shown how a simple, measurement-based model can fairly accurately model the available capacity and thus can guide call admission decisions.

Second, because of the wireless interference, looking for a feasible route to accommodate an incoming call can be computationally hard. We have simplified this issue by introducing the assumption of the knowledge of the ratio of interference and carrier sensing ranges. This ensures that path segments of constant length can be evaluated separately to determine feasibility in polynomial time. We have also introduced routing metrics such as max residual feasible path and new strategies like routing using call statistics. Both improve performance significantly compared to naive methods.

Our modeling work is general enough that it can be extended for newer architectures such as directional antenna or multi-radio/multi-channel system – something that we will address in our future work. We have not explicitly accounted for data traffic in our evaluation, as our methods can always be used to set aside some amount of capacity for data traffic.

CHAPTER 6

CONCLUSIONS

In this dissertation, we have proposed several methods to model interference and its impact on capacity. We have relied on measurements rather than using analytical or theoretical methods, which makes our models realistic and practical to be used for deployed Wi-Fi networks. We demonstrate the application of such modeling using the example of supporting VoIP calls over a wireless mesh network. Specifically, we have made the following contributions –

- We first develop a link capacity model based on the physical interference model, which uses measurements on the target network. The methods we proposed are practical. The profiling measurements used to model the physical layer can be kept in a library and reused. The measurements on the target network are simple and take $O(N)$ steps. The model solutions using analytical and simulation-based methods compare favorably with experiments done on a real network.
- We then demonstrated that measurements from a real network can be used to accurately model the physical layer of a wireless network simulator. We create two versions of the ns-2 simulator with different levels of fidelity and measurement requirements to model deferral, reception, and propagation behaviors at the physical layer. We show excellent accuracy of this modified simulator when compared with experiments on a real network.
- We propose a novel machine learning approach to estimate interference in a Wi-Fi network. The technique uses a merged packet trace collected via distributed sniffing.

It then recreates the MAC layer interactions between network nodes via a machine learning approach using the Hidden Markov Model. This is finally helpful in inferring pair-wise interference relationships. The advantage of this approach is that it is purely passive and thus can work with a live network without any access to the network elements. We have demonstrated via experiments and simulations that this technique estimates interference with accuracy similar to methods using measurement-based profiling and active measurements directly on the network.

- We finally show an application of the capacity model we have created. We address two important questions in running VoIP on wireless mesh networks. First, maintaining QoS means that call admission control must be performed. However, without any reasonable model of multihop capacity of the network, the admission decisions cannot be taken. We have shown how a simple, measurement-based model can fairly accurately model the available capacity and thus can guide call admission decisions. Second, because of the wireless interference, looking for a feasible route to accommodate an incoming call can be computationally hard. We have simplified this issue by introducing the assumption of the knowledge of the ratio of interference and carrier sensing ranges. This ensures that path segments of constant length can be evaluated separately to determine feasibility in polynomial time. We have also introduced routing metrics such as max residual feasible path and new strategies like routing using call statistics. Both improve performance significantly compared to naive methods.

BIBLIOGRAPHY

- [1] AirMagnet. <http://airmagnet.com>.
- [2] AirPatrol. <http://airpatrolcorp.com>.
- [3] Mathematica 5.2. <http://www.wolfram.com/>.
- [4] Multiband Atheros Driver for WiFi (MADWIFI).
<http://sourceforge.net/projects/madwifi/>.
- [5] OpNet. <http://opnet.com>.
- [6] QualNet. <http://scalable-networks.com>.
- [7] The Network Simulator - ns-2. <http://www.isi.edu/nsnam/ns>.
- [8] HFA3863 Data Sheet: Direct Sequence Spread Spectrum Baseband Processor with Rake Receiver and Equalizer. Intersil Corporation, 2000.
- [9] Daniel Aguayo, John Bicket, Sanjit Biswas, Glenn Judd, and Robert Morris. Link-level measurements from an 802.11b mesh network. *SIGCOMM Comput. Commun. Rev.*, 34(4), 2004.
- [10] Ian F. Akyildiz, Xudong Wang, and Weilin Wang. Wireless mesh networks: a survey. *Computer Networks and ISDN Systems*, 47(4):445–487, 2005.
- [11] Mansoor Alicherry, Randeep Bhatia, and Li (Erran) Li. Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks. In *MobiCom '05*, pages 58–72, 2005.

- [12] TR Andel and A. Yasinsac. On the credibility of manet simulations. *Computer*, 39(7):48–54, 2006.
- [13] Paramvir Bahl, , Jitendra Padhye, Lenin Ravindranath, Manpreet Singh, Alec Wolman, and Brian Zill. Dair: A framework for managing enterprise wireless networks using desktop infrastructure. In *Annual ACM Workshop on Hot Topics in Networks (HotNets)*, November 2005.
- [14] Paramvir Bahl, Ranveer Chandra, Jitendra Padhye, Lenin Ravindranath, Manpreet Singh, Alec Wolman, and Brian Zill. Enhancing the security of corporate wi-fi networks using dair. In *MobiSys '06: Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 1–14, New York, NY, USA, 2006. ACM.
- [15] Leonard E. Baum and J. A. Eagon. An inequality with applications to statistical estimation for probabilistic functions of markov processes and to a model for ecology. *Bull. Amer. Math. Soc.*, 73:360–363, 1967.
- [16] G. Bianchi. Performance analysis of the IEEE 802.11 Distributed Coordination Function. *JSAC*, 2000.
- [17] Joseph Camp, Joshua Robinson, Christopher Steger, and Edward Knightly. Measurement driven deployment of a two-tier urban mesh access network. In *MobiSys 2006*, 2006.
- [18] Hoon Chang, Vishal Misra, and Dan Rubenstein. A General Model and Analysis of Physical Layer Capture in 802.11 Networks. In *Proc. IEEE Infocom*, 2006.
- [19] Lei Chen and Wendi Heinzelman. Qos-aware routing based on bandwidth estimation for mobile ad hoc networks. *IEEE Journal of Selected Areas in Communications*, 2005.

- [20] Yu-Chung Cheng, Mikhail Afanasyev, Patrick Verkaik, Péter Benkő, Jennifer Chiang, Alex C. Snoeren, Stefan Savage, and Geoffrey M. Voelker. Automating cross-layer diagnosis of enterprise wireless networks. *ACM SIGCOMM*, 2007.
- [21] Yu-Chung Cheng, John Bellardo, Péter Benkő, Alex C. Snoeren, Geoffrey M. Voelker, and Stefan Savage. Jigsaw: solving the puzzle of enterprise 802.11 analysis. *ACM SIGCOMM*, 2006.
- [22] R. Cle and J. Rosenbluth. Voice over IP performance monitoring. *ACM Computer Communication Review*, 31(2), April 2001.
- [23] D. De Couto and D. Aguayo and J. Bicket and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proc. of ACM MOBICOM*, 2003.
- [24] S. Das, D. Koutsonikolas, Y. Hu, and D. Peroulis. Characterizing multi-way interference in wireless mesh networks. In *ACM Wintech*, 2005.
- [25] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.
- [26] K. Fall and K. Varadhan. ns notes and documentation. <http://www.isi.edu/nsnam/ns>, Nov 1997.
- [27] Firetide. Wireless instant networks. <http://www.firetide.com>.
- [28] L. R. Ford and D. R. Fulkerson. Flows in networks. *Princeton University Press*, 1962.
- [29] ITU-T Recommendation G.729a. Coding of speech at 8kbit/s using conjugate-structure algebraic-code-excited linear-prediction (SC-ACELP), March 1996.
- [30] Samrat Ganguly, Vishnu Navda, Kyungtae Kim, Anand Kashyap, Dragos Niculescu, Rauf Izmailov, S. Hong, and Samir R Das. Performance Optimizations for Deploying

- VoIP Services in Mesh Networks. *IEEE Journal on Selected Areas in Communications*, 24(11):2147–2158, Nov. 2006.
- [31] Yan Gao, Dah-Ming Chiu, and John C.S. Lui. Determining the end-to-end throughput capacity in multi-hop networks: methodology and applications. *SIGMETRICS Perform. Eval. Rev.*, 34(1):39–50, 2006.
- [32] Michele Garetto, Theodoros Salonidis, and Edward W. Knightly. Modeling per-flow throughput and capturing starvation in csma multi-hop wireless networks. In *Proc. of IEEE INFOCOM*, Barcelona, April 2006.
- [33] S. Garg and M. Kappes. Can I add a VoIP call? In *In Proceedings of IEEE ICC*, 2003.
- [34] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2):388–404, March 2000.
- [35] J. Heidemann, N. Bulusu, J. Elson, C. Intanagonwiwat, K. Lan, Y. Xu, W. Ye, D. Estrin, and R. Govindan. Effects of detail in wireless network simulation. In *Proceedings of the SCS Multiconference on Distributed Simulation*, 2001.
- [36] D. Hole and F. Tobagi. Capacity of an IEEE 802.11b Wireless LAN supporting VoIP. In *In Proceedings of IEEE ICC*, 2004.
- [37] Kamal Jain, Jitendra Padhye, Venkata N. Padmanabhan, and Lili Qiu. Impact of interference on multi-hop wireless network performance. In *MobiCom*, 2003.
- [38] Kyle Jamieson, Bret Hull, Allen K. Miu, and Hari Balakrishnan. Understanding the Real-World Performance of Carrier Sense. In *E-WIND*, Philadelphia, PA, August 2005.
- [39] Amit P. Jardosh, Kimaya Mittal, Krishna N. Ramachandran, Elizabeth M. Belding, and Kevin C. Almeroth. Iqu: practical queue-based user association management for wlans. In *ACM MobiCom*, 2006.

- [40] Amit P. Jardosh, Krishna N. Ramachandran, Kevin C. Almeroth, and Elizabeth M. Belding-Royer. Understanding congestion in ieee 802.11b wireless networks. In *ACM IMC*, 2005.
- [41] D. Johnson. Validation of wireless and mobile network models and simulation. In *Proceedings of the DARPA/NIST Network Simulation Validation Workshop, Fairfax, Virginia, USA*, 1999.
- [42] Glenn Judd and Peter Steenkiste. Repeatable and realistic wireless experimentation through physical emulation. *SIGCOMM Comput. Commun. Rev.*, 34(1):63–68, 2004.
- [43] Glenn Judd and Peter Steenkiste. Understanding link-level 802.11 behavior: Replacing convention with measurement. In *Wicon*, 2007.
- [44] Anand Kashyap, Samir R. Das, and Samrat Ganguly. Measurement-based approaches for accurate simulation of 802.11-based wireless networks. Technical Report. <http://www.wings.cs.sunysb.edu/~anand/papers/simulation.pdf>, 2008.
- [45] Anand Kashyap, Samrat Ganguly, and Samir Das. A measurement-based approach to modeling link capacity in 802.11-based wireless networks. In *ACM MobiCom*, Montreal, September 2007.
- [46] Anand Kashyap, Samrat Ganguly, and Samir Das. Measurement-based approaches for accurate simulation of 802.11-based wireless networks. In *MSWIM*, Vancouver, October 2008.
- [47] Anand Kashyap, Samrat Ganguly, Samir Das, and Suman Banerjee. Voip on wireless meshses: Models, algorithms and evaluation. In *IEEE Infocom*, Anchorage, Alaska, May 2007.

- [48] Anand Kashyap, Samrat Ganguly, and Samir R. Das. Characterizing interference in 802.11-based wireless mesh networks. <http://www.wings.cs.sunysb.edu/~anand/interference.pdf>, 2006.
- [49] Anand Kashyap, Samrat Ganguly, and Samir R. Das. A measurement-based approach to modeling link capacity in 802.11-based wireless networks. In *ACM MobiCom*, 2007.
- [50] C. T. Kelley. Solving nonlinear equations with newton's method. In *Fundamentals of Algorithms, SIAM*, 2003.
- [51] Kyu-Han Kim and Kang G. Shin. On accurate measurement of link quality in multi-hop wireless mesh networks. In *MobiCom '06*, pages 38–49, 2006.
- [52] M. Kodialam and T. V. Lakshman. Minimum Interference Routing with Applications to MPLS Traffic Engineering. In *Proc. of IEEE INFOCOM*, 2000.
- [53] Murali Kodialam and Thyaga Nandagopal. Characterizing achievable rates in multi-hop wireless networks: the joint routing and scheduling problem. In *MobiCom*, pages 42–54, 2003.
- [54] D. Kotz, C. Newport, R.S. Gray, J. Liu, Y. Yuan, and C. Elliott. Experimental evaluation of wireless simulation assumptions. *Proc. ACM MSWiM Symposium*, pages 78–82, 2004.
- [55] Anurag Kumar, Eitan Altman, Daniele Miorandi, and Munish Goyal. New insights from a fixed point analysis of single cell ieee 802.11 wireless LANs. In *Proceedings IEEE Infocom*, 2005.
- [56] V. S. Anil Kumar, Madhav V. Marathe, Srinivasan Parthasarathy, and Aravind Srinivasan. Algorithmic aspects of capacity in wireless networks. *SIGMETRICS Perform. Eval. Rev.*, 33(1):133–144, 2005.

- [57] V. S. Anil Kumar, Madhav V. Marathe, Srinivasan Parthasarathy, and Aravind Srinivasan. Algorithmic aspects of capacity in wireless networks. *SIGMETRICS Perform. Eval. Rev.*, 33(1):133–144, 2005.
- [58] Seoung-Bum Lee, Gahng-Seop Ahn, Xiaowei Zhang, and Andrew T. Capbell. INSIGNIA: an IP-based quality of service framework for mobile ad hoc networks. *J. Parallel Distrib. Comput.*, 60(4):374–406, 2000.
- [59] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi. An introduction to the application of the theory of probabilistic functions of a markov process to automatic speech recognition. *Bell Syst. Tech. J.*, 62(4):1035–1074, 1983.
- [60] J. Liu, Y. Yuan, D.M. Nicol, R.S. Gray, C.C. Newport, D. Kotz, and L.F. Perrone. Simulation validation using direct execution of wireless Ad-Hoc routing protocols. In *Proc. PADS Workshop*, pages 7–16, 2004.
- [61] Ratul Mahajan, Maya Rodrig, David Wetherall, and John Zahorjan. Analyzing the mac-level behavior of wireless networks in the wild. In *ACM SIGCOMM*, 2006.
- [62] Meru Networks. <http://www.merunetworks.com>.
- [63] D Niculescu, S Ganguly, K Kim, and R Izmailov. Performance of VoIP in a 802.11 Wireless Mesh Network. In *Proc. of IEEE INFOCOM 2006*, Barcelona, 2006.
- [64] B. O’Hara, P. Calhoun, and J. Kempf. Configuration and provisioning for wireless access points (CAPWAP). RFC 3990, February 2005.
- [65] J. Padhye, S. Agarwal, V. Padmanabhan, L. Qiu, A. Rao, and B. Zill. Estimation of link interference in static multi-hop wireless networks. In *IMC*, 2005.
- [66] R.J. Punnoose, P.V. Nikitin, and D.D. Stancil. Efficient simulation of ricean fading within a packet simulator. In *Proceedings of IEEE Vehicular Technology Conference (VTC 2000)*, pages 764–767, 2000.

- [67] Lili Qiu, Yin Zhang, Feng Wang, Mi Kyung Han, and Ratul Mahajan. A general model of wireless interference. In *ACM MobiCom*, 2007.
- [68] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Readings in speech recognition*, pages 267–296, 1990.
- [69] Ashish Raniwala, Kartik Gopalan, and T. Chiueh. Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 8(2):50–65, 2004.
- [70] Theodore S. Rappaport. *Wireless Communications: Principles and Practice*. IEEE Press, Piscataway, NJ, USA, 1996.
- [71] Charles Reis, Ratul Mahajan, Maya Rodrig, David Wetherall, and John Zahorjan. Measurement-based models of delivery and interference in static wireless networks. In *SIGCOMM*, 2006.
- [72] Maya Rodrig, Charles Reis, Ratul Mahajan, David Wetherall, and John Zahorjan. Measurement-based characterization of 802.11 in a hotspot setting. In *ACM E-WIND*, 2005.
- [73] Aaraon Schulman, Dave Levin, and Neil Spring. On the fidelity of 802.11 packet traces. In *PAM*, 2008.
- [74] Soekris Engineering. <http://www.soekris.com>.
- [75] D. Son, B. Krishnamachari, and J. Heidemann. Experimental study of concurrent transmission in wireless sensor networks. In *Proc. ACM SenSys*, 2006.
- [76] Dongjin Son, Bhaskar Krishnamachari, and John Heidemann. Experimental study of concurrent transmission in wireless sensor networks. In *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 237–250, New York, NY, USA, 2006. ACM Press.

- [77] S. Sriram, T. Bheemarjuna Reddy, B. S. Manoj, and C. Siva Ram Murthy. On the end-to-end call acceptance and the possibility of deterministic qos guarantees in ad hoc wireless networks. In *MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 169–180, 2005.
- [78] Yuan Sun, Elizabeth Belding-Royer, Xia Gao, and James Kempf. A priority-based distributed call admission protocol for multi-hop wireless ad hoc networks. UCSB Technical Report, 2004.
- [79] Mineo Takai, Jay Martin, and Rajive Bagrodia. Effects of wireless physical layer modeling in mobile ad hoc networks. In *Proc. ACM MobiHoc Symposium*, pages 87–94, 2001.
- [80] Y. C. Tay and K. C. Chua. A capacity analysis for the IEEE 802.11 MAC protocol. *Wirel. Netw.*, 7(2):159–171, 2001.
- [81] M. Veeraraghavan, N. Cocker, and T. Moors. Support of voice services in ieee 802.11 wireless lans. In *In Proceedings of IEEE Infocom*, 2001.
- [82] Hung-Yu Wei, KyungTae Kim, Anand Kashyap, and Samrat Ganguly. On Admission of VoIP Calls over Wireless Mesh Network. In *In Proceedings of IEEE ICC*, 2006.
- [83] Qi Xue and Aura Ganz. Ad hoc qos on-demand routing (aqor) in mobile ad hoc networks. *J. Parallel Distrib. Comput.*, 63(2):154–165, 2003.
- [84] Jihwang Yeo, Moustafa Youssef, and Ashok Agrawala. A framework for wireless lan monitoring and its applications. In *ACM WiSe*, 2004.
- [85] Jihwang Yeo, Moustafa Youssef, Tristan Henderson, and Ashok Agrawala. An accurate technique for measuring the wireless side of wireless networks. In *WiTMeMo*. USENIX Association, 2005.

- [86] J. Zhou, Z. Ji, M. Varshney, Z. Xu, Y. Yang, M. Marina, and R. Bagrodia. WHYNET: a hybrid testbed for large-scale, heterogeneous and adaptive wireless networks. In *Proc. WINTECH*, pages 111–112, 2006.