A General Model of Wireless Interference

Lili Qiu Yin Zhang Feng Wang Mi Kyung Han
University of Texas at Austin
Austin, TX 78712, USA
{lili,yzhang,wangf,hanmi2}@cs.utexas.edu

Ratul Mahajan
Microsoft Research
Redmond, WA 98052, USA
ratul@microsoft.com

ABSTRACT

We develop a general model to estimate the throughput and goodput between arbitrary pairs of nodes in the presence of interference from other nodes in a wireless network. Our model is based on measurements from the underlying network itself and is thus more accurate than abstract models of RF propagation such as those based on distance. The seed measurements are easy to gather, requiring only $O(N)$ measurements in an N-node networks. Compared to existing measurement-based models, our model advances the state of the art in three important ways. First, it goes beyond pairwise interference and models interference among an arbitrary number of senders. Second, it goes beyond broadcast transmissions and models the more common case of unicast transmissions. Third, it goes beyond homogeneous nodes and models the general case of heterogeneous nodes with different traffic demands and different radio characteristics. Using simulations and measurements from two different wireless testbeds, we show that the predictions of our model are accurate in a wide range of scenarios.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Modeling techniques

General Terms

Measurement, Performance

Keywords

Model, Wireless Interference

1. INTRODUCTION

Interference is fundamental to wireless networks. Due to the broadcast nature of the medium, transmissions from one sender interfere with the transmission and reception capabilities of other nodes. Understanding and managing interference is essential to the performance of wireless networks. For instance, it can directly benefit channel assignment [21, 25], transmit power control [12], routing [5, 6], transport protocols [18], and network diagnosis [4].

Unfortunately, the state of the art in estimating the impact of interference is rather primitive. Much of the existing work is based on simple, abstract models of radio propagation (*e.g.*, the interference range is twice the communication range). While such models may predict the asymptotic behavior, they can be highly inaccurate in any given network [14, 1].

MobiCom'07, September 9–14, 2007, Montréal, Québec, Canada.

This has prompted researchers to devise models that are seeded using measurements from the underlying network [1, 24]. These measurements are usually collected in a simple configuration, such as each node sending by itself. They are then used to predict the impact of interference in more complex configurations such as multiple transmitting nodes. This is a promising direction because it makes no assumptions about the nature of radio propagation which has proven difficult to model in real environments.

However, the existing measurement-based models are quite limited. They do not apply to configurations that have more than two senders or two flows, have unicast traffic, or have senders with finite demands. The only way today to predict network behavior under these general configurations is to actually measure it. (Indeed, most experimental research today is forced to adopt this methodology.)

But measurement alone is insufficient because it lacks predictive power and scalability. While it can accurately predict the performance of the measured configuration, it cannot predict performance for other configurations. To optimize network performance, one often needs to predict the performance of many alternative configurations. Since measuring all possible configurations is not feasible, it is necessary to develop a model to estimate network performance under arbitrary configurations (e.g., to perform what-if analysis).

In this paper, we develop a general model of interference in heterogeneous multihop wireless networks with asymmetric link quality and non-binary interference relationships. Our model takes as input traffic demand and received signal strength (RSS) between pairs of nodes, which requires only $O(N)$ measurements in an N-node network. It then estimates the rate at which each sender will transmit and the rate at which each receiver will successfully receive packets.

Compared to existing measurement-based models [1, 24], we advance the state of art in three important ways. First, we go beyond the case of two senders (or flows) and model interference among an arbitrary number of senders. This is challenging due to complex interactions among nodes. For instance, the sending rate of node m depends on those of all other nodes, which in turn depend on the sending rate of m itself. Second, we go beyond broadcast transmissions and model the more common case of unicast transmissions. Unicast transmissions introduce additional complexities due to retransmissions, exponential backoff, possibly asymmetric link qualities, and collisions with not only data packets but also ACK packets. Third, we go beyond the case of infinite traffic demands and model the more realistic case of finite demands. Most real networks have heterogeneous nodes with varying traffic demands.

Our model consists of three major components:

1. *An N-node Markov model for capturing interactions among an arbitrary number of broadcast senders.* The model provides a simple yet accurate approximation to the 802.11 distributed coordination function (DCF). It is more general than previous models (*e.g.*, [2]) and can support multihop wireless networks, unsaturated demands, asymmetric link quality, and non-binary interference relationships.

2. *A receiver model of packet-level loss rates.* In particular, we find that slot-level and packet-level loss rates can be quite different depending on how losses are generated. Hidden terminals can significantly increase the packet-level loss rates well beyond the slot-level loss rates by spreading the lossy time slots across many packets. Based on this observation, our receiver model captures both synchronized and unsynchronized packet-level collision losses.

3. *Unicast sender and receiver models.* We further extend the above broadcast sender and receiver models to capture interactions among unicast transmissions. We develop two major extensions for this purpose. The first extension models the retransmission and exponential backoff at the sender side, and the second extension models data/data, data/ACK, and ACK/ACK collision losses at the receiver side.

We evaluate our model using both extensive simulations and real measurements over two different wireless testbeds. Our results show that the model gives accurate predictions over a wide range of scenarios for both broadcast and unicast traffic, with both saturated and unsaturated demands, and across different number of senders. In simulations, where accurate RF profile is available, our model's root mean square error (RMSE) is less than 0.05 for both throughput and goodput predictions. In the testbeds, where the RF profile is empirically measured and subject to measurement noise and bias due to lost packets, our model's RMSE is less than 0.12. While our model is more general, we find that its accuracy is higher than the state-of-art model that considers the special case of two broadcast senders with infinite demands [24].

Paper organization: The rest of the paper is organized as follows. In Section 2, we review the background of IEEE 802.11. In Section 3, we give an overview of our interference model. We present broadcast models in Section 4, and unicast models in Section 5. In Section 6, we describe how to obtain model inputs. We evaluate our model using simulations in Section 7 and using testbed experiments in Section 8. We discuss the related work in Section 9 and conclude in Section 10.

2. BACKGROUND ON 802.11

The IEEE 802.11 standard [22] specifies two types of coordination functions for stations to access the wireless medium: distributed coordination function (DCF) and point coordination function (PCF). In this paper, we focus on DCF, which is much more widely used than PCF. DCF is based on CSMA/CA. Before transmission, a station first checks to see if the medium is available by using virtual carrier-sensing and physical carrier-sensing. The medium is considered busy if either carrier-sensing indicates so. Virtual carrier-sensing considers the medium to be idle if the Network Allocation Vector (NAV) is zero, otherwise it considers the medium to be busy. Only when NAV is zero, physical carrier-sensing is performed. A station determines the channel to be idle when the total energy received at a node is less than the CCA (clear-channel assessment) threshold. In this case, a station may begin transmission using the following rule. If the medium has been idle for longer than a distributed inter-frame spacing time (DIFS) period, transmission can begin immediately. Otherwise, a station that has data to send first waits for DIFS and then waits for a random backoff interval uniformly chosen between $[0, CW_{\min}]$, where $CW_{\min}$ is the minimum contention window. If at any time during the period above the medium is sensed busy, the station freezes its counter and the countdown resumes only after the medium becomes idle again for DIFS. When the counter decrements to zero, the node transmits the packet. In the case of unicast, if the receiver successfully receives the packet, it waits for a short inter-frame spacing time (SIFS) and then transmits an ACK frame. If the sender does not receive an ACK, it doubles its contention window to reduce its access rate. When the contention window reaches its maximum value, it stays at that value until a transmission succeeds, in which case the contention window is reset to $CW_{\min}$.

Model inputs: measured	
R_{mn}	RSS from node m to n
B_n	Background interference at n
d_{mn}	Traffic demand from m to n
Model inputs: radio-dependant constants	
β_m	CCA threshold of m
γ_n	Radio sensitivity of n
δ_n	SINR threshold of n
W_n	Thermal noise of n
Model outputs	
t_{mn}	Normalized throughput: rate of traffic sent by m to n
g_{mn}	Normalized goodput: rate of traffic received by n from m
L_{mn}	Packet loss rate from m to n
Other variables	
S_i	Subset of nodes that are transmitting in state i
π_i	Probability that the network is in state i
M	Matrix of transition probabilities among states
$C(m\|S_i)$	Probability that channel is clear at m in state i
$Q(m)$	Probability that m has data to send when backoff counter = 0
$\overline{OH}(m)$	Average overhead from DIFS, SIFS, and ACK at sender m
$\overline{CW}(m)$	Average contention window of m
$T_\mu(m)$	Average packet transmission time for m

Table 1: A summary of key notations.

3. OVERVIEW OF OUR MODEL

Our model takes traffic demands and RF profile as input and outputs the estimated sending and receiving rates for each node. Such a model is a powerful tool for performing what-if analysis and facilitating network optimization and diagnosis. More specifically, consider a network with N nodes. The inputs to the model are: *i*) traffic demand from each sender m to each receiver n, and *ii*) RF profile, which refers to the received signal strength (RSS) between every pair of nodes, denoted as R_{mn}. The outputs are: *i*) normalized throughput t_{mn}, *i.e.*, the fraction of time when m is sending traffic to n (including header overhead and retransmissions), *ii*) normalized goodput g_{mn}, *i.e.*, the fraction of time when n is receiving useful data from m (excluding header overhead and duplicate traffic), and *iii*) the packet loss rate L_{mn}.

In this paper, we focus on *one-hop* traffic demands, which means that traffic is only sent over one hop and not routed further. If n cannot hear from m, its receiving rate is zero. Modeling network performance under one-hop traffic demands is an important and necessary step towards estimating end-to-end throughput over multihop paths, which we plan to investigate in the future.

Our model operates as follow. First, we measure the RF profile of the network by letting each sender broadcast in turn and having the other nodes measure received RSSI values and loss rates. From these measurements, we recover pairwise RSS (R_{mn}) and background interference (B_n) due to external sources other than nodes in the modeled network (Section 6). While we use custom traffic for our experiments, it may be feasible to perform these measurements using normal application traffic.

Then, we apply our *sender model* to estimate the amount of traffic sent by each sender under the given demand and our *receiver model* to estimate the amount of traffic successfully received. Our key contributions lie in the generality and accuracy of the sender and receiver models. They apply to both broadcast and unicast transmissions for an arbitrary number of senders, with and without saturated traffic demands. For saturated broadcast demands, our model can directly estimate throughput and goodput by computing the stationary probabilities of a Markov model. For unicast demands or unsaturated broadcast demands, the transition matrix of the Markov model itself involves additional unknown variables to be estimated. As a result, the stationary probabilities cannot be directly solved. We solve the problem by applying an *iterative* framework, where we first initialize the unknown variables in the transition matrix and then compute stationary probabilities, which are then used to update the transition matrix. Our results show that the iteration framework is effective and converges quickly (within 10 iterations in our evaluation).

We assume the following radio behavior. A transmitter m determines the channel is "clear" when the total energy it receives is below the CCA (clear-channel assessment) threshold, β_m. A receiver n correctly decodes a transmission from a sender m when *i*) its signal strength is at least radio sensitivity, γ_n; and *ii*) the signal to interference-plus-noise ratio (SINR) is at least the SINR threshold, δ_n. We denote the thermal noise experienced by n as W_n. The values of β_m, γ_n, δ_n, and W_n are constant but radio-dependent.

The key notations used in this paper are summarized in Table 1. We explain each term when it is first encountered.

4. BROADCAST TRAFFIC

In this section, we present our model for broadcast traffic. Extensions to handle unicast traffic are presented in the next section.

4.1 Sender Model

The goal of the broadcast sender model is to estimate how much each sender can transmit given traffic demand. The classic Bianchi model [2] and its extensions (*e.g.*, [20]) model the behavior of 802.11 DCF by constructing a discrete Markov chain. To make the model tractable, all packet transmissions are assumed to be synchronized, *i.e.*, there are no partially overlapping transmissions. In a general multihop wireless network, however, partially overlapping transmissions can be common because not all nodes can carrier sense each other. Thus, these models do not directly apply.

We develop a general N-node broadcast sender model based on Markov chains. We present it incrementally. First, we present the model for saturated traffic demands with variable packet sizes. Then, we extend it to handle fixed packet sizes and unsaturated demands in Sections 4.1.1 and 4.1.2. Finally, we describe techniques to enhance the scalability of the model in Section 4.1.3.

At a high level, we construct a Markov chain where each state i represents a set of nodes (denoted by S_i) that are transmitting simultaneously in a time slot. Given N senders, the Markov chain has 2^N possible states (which we prune in Section 4.1.3). We derive the transition matrix M for the Markov chain based on 802.11 DCF and use it to compute the stationary probability π_i of each state. The throughput of node m is then simply $t_m = \sum_{i|m\in S_i} \pi_i$.

Deriving the transition matrix M: In this section, we assume that nodes send variable-length packets with exponential distribution, and that the state transitions of different nodes are independent. We relax these assumptions in Section 4.1.1. Under the independence assumption, we can focus on computing the transition probabilities of an individual node, say m. This involves computing four transition probabilities for every state i: (a) staying in idle mode, $P_{00}(m|S_i)$; (b) entering transmission mode, $P_{01}(m|S_i)$; (c) exiting transmission mode, $P_{10}(m|S_i)$; and (d) staying in transmission mode, $P_{11}(m|S_i)$. The probability $M(i,j)$ for the network to transition from state i to j is:

$$\begin{aligned} M(i,j) \;=\; & \prod\nolimits_{m\in\overline{S_i}\cap\overline{S_j}} P_{00}(m|S_i)\times \\ & \prod\nolimits_{m\in\overline{S_i}\cap S_j} P_{01}(m|S_i)\times \\ & \prod\nolimits_{m\in S_i\cap\overline{S_j}} P_{10}(m|S_i)\times \\ & \prod\nolimits_{m\in S_i\cap S_j} P_{11}(m|S_i), \end{aligned} \tag{1}$$

where $\overline{S}$ denotes the complement of a given set S.

We compute the four per-node probabilities based on 802.11 DCF. A node can begin transmission when the following three conditions are satisfied: *i*) its random backoff counter reaches 0; *ii*) the medium is clear; and *iii*) the node has data to transmit. Therefore:

$$\begin{aligned} P_{01}(m|S_i) &= Pr[\text{medium is clear} \wedge \text{counter} = 0 \wedge m \text{ has data}] \\ &= Pr[\text{medium is clear}] \\ &\quad \times Pr[\text{counter} = 0\,|\,\text{medium is clear}] \\ &\quad \times Pr[m \text{ has data}\,|\,\text{medium is clear} \wedge \text{counter} = 0] \\ &= C(m|S_i) \times \frac{1}{\overline{CW}(m) + \overline{OH}(m)} \times Q(m), \end{aligned} \tag{2}$$

where $C(m|S_i)$ is the probability for m's medium to be clear while in state i, which we will compute below. $\overline{OH}(m)$ (for overhead) denotes the extra clear time slots that node m needs to wait in addition to $\overline{CW}(m)$, the average contention window. For broadcast transmissions, we have $\overline{CW}(m) = \frac{CW_{\min}}{2}$ and $\overline{OH}(m) = \frac{T_{\text{DIFS}}}{T_{\text{slot}}}$, where T_{DIFS} is the DIFS duration and T_{slot} is the duration of a time slot. $Q(m)$ is the probability that m has data to send given that the medium is clear and the backoff counter is zero. For saturated demands, $Q(m) = 1$. We derive $Q(m)$ for unsaturated demands in Section 4.1.2.

For the staying idle probability, we set $P_{00}(m|S_i) = 1 - P_{01}(m|S_i)$.

To compute $P_{10}(m|S_i)$ and $P_{11}(m|S_i)$, assume that both transmission and idle times are exponentially distributed. (We relax this assumption in Section 4.1.1.) Let $T_\mu(m)$ be m's average packet transmission time, computed based on m's packet size and transmission rate, and T_{slot} denote the duration of a time slot. We have:

$$P_{10}(m|S_i) = T_{\text{slot}}/T_\mu(m) \tag{3}$$

$$P_{11}(m|S_i) = 1 - P_{10}(m|S_i) = 1 - T_{\text{slot}}/T_\mu(m) \tag{4}$$

Computing the clear probability $C(m|S_i)$: We have $C(m|S_i) = \Pr\{I_{m|S_i} \le \beta_m\}$, where $I_{m|S_i}$ is the total interference at m in state i and β_m is the CCA threshold. $I_{m|S_i}$ is the sum of constant thermal noise W_m, the background interference B_m, and interference due to data transmissions by nodes in S_i (except for m itself). Thus, $I_{m|S_i} = W_m + B_m + \sum_{s\in S_i\setminus\{m\}} R_{sm}$. To estimate this sum, we model each term as a lognormal random variable — our testbed measurement results (omitted for lack of space) suggest that the lognormal distribution fits the measured RSSI well. The standard approach for analyzing the sum of lognormal random variables is to approximate the sum itself by a lognormal random variable [7, 26]. Following Fenton [7], we find a lognormal random variable that matches the mean and the variance of $I_{m|S_i}$.

Formally, assuming that R_{sm} ($\forall s \in S_i$) and B_m are all independent, we have: $E[I_{m|S_i}] = W_m + \bar{\mathsf{B}}_m + \sum_{s\in S_i\setminus\{m\}} \bar{\mathsf{R}}_{sm}$, and $Var[I_{m|S_i}] = \mathsf{B}^{\text{var}}_m + \sum_{s\in S_i\setminus\{m\}} \mathsf{R}^{\text{var}}_{sm}$. Let e^Z be a lognormal random variable with $Z \sim N(\mu, \sigma^2)$. The first two moments of e^Z are $E[\mathsf{e}^Z] = \mathsf{e}^{\mu+\sigma^2/2}$ and $E[\mathsf{e}^{2Z}] = \mathsf{e}^{2\mu+2\sigma^2}$. Equating the first two moments of e^Z and $I_{m|S_i}$ gives: $\mathsf{e}^{\mu+\sigma^2/2} = E[I_{m|S_i}]$, and $\mathsf{e}^{2\mu+2\sigma^2} = E[I^2_{m|S_i}] = Var[I_{m|S_i}] + (E[I_{m|S_i}])^2$. Therefore, we have:

$$\mu = 2\log E[I_{m|S_i}] - \frac{1}{2}\log E[I^2_{m|S_i}] \tag{5}$$

$$\sigma^2 = \log E[I^2_{m|S_i}] - 2\log E[I_{m|S_i}] \tag{6}$$

We can then approximate $C(m|S_i)$ as:

$$C(m|S_i) \approx \Pr\{\mathsf{e}^Z \le \beta_m\} = \Pr\{Z \le \log\beta_m\} = \Phi\left[\frac{\log\beta_m - \mu}{\sigma}\right],$$

where $\Phi[x] = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} \mathsf{e}^{\frac{-u^2}{2}}\,du$ is the standard normal CDF.

Computing stationary probabilities π_i: Having derived the transition matrix M, we can compute the stationary probabilities π_i by solving the following system of linear equations:

$$\sum_i \pi_i \times M(i,j) = \pi_j \quad (\forall j) \tag{7}$$

$$\sum_i \pi_i = 1 \tag{8}$$

where (7) comes from the property that the stationary probabilities of the current and next states are equal, and (8) normalizes the sum of the stationary probabilities to 1. When M is sparse, π can be efficiently solved in Matlab using *lsqr* [23]. Section 4.1.3 describes how to make M sparse to enhance scalability.

4.1.1 Handling Similar Packet Sizes

The previous section assumes variable packet sizes and independent transition probabilities for various nodes. But when all nodes use similar packet sizes, the independence assumption no longer holds. Specifically, given two nodes within carrier sense range from each other, unless their random backoff counters both reach 0 together, one node will start transmitting earlier and the other node will sense the carrier and defer to the earlier node [22]. As a result, overlapping transmissions from the two nodes will have almost identical start times. With similar packet sizes, such transmissions will also end at similar times. Such synchronization clearly violates the independence assumption.

To handle such scenarios, we construct a *synchronization graph* $\mathcal{G}_{\text{syn}}(S_i)$ for each state i as follows. Two nodes $m, n \in S_i$ are connected in $\mathcal{G}_{\text{syn}}(S_i)$ if and only if $C(m|\{n\}) < 0.1$ and $C(n|\{m\}) < 0.1$, where $C(s|\{t\})$ is the clear probability at node s when node t alone is transmitting. We find all the connected components in $\mathcal{G}_{\text{syn}}(S_i)$ and define each component as a *synchronization group*.

We then make two adjustments to the transition probabilities $M(i,j)$ to account for the synchronization effect. First, if there exist two nodes m and n in the same synchronization group of $\mathcal{G}_{\text{syn}}(S_i)$ such that $m \in S_j$ and $n \in \overline{S_j}$, then $M(i,j) = 0$. This is because all nodes in a synchronization group must exit the transmission mode together. Second, the probability for all nodes in a synchronization group G to exit the transmission mode together is (T_{slot}/T_μ). In contrast, under the independent transmission model, the probability is $\prod_{m \in G}(T_{\text{slot}}/T_\mu(m)) = \left(T_{\text{slot}}/T_\mu\right)^{|G|}$, which is much smaller when $|G| \geq 2$. Here $|G|$ is the number of nodes in G.

4.1.2 Handling Unsaturated Demands

The main challenge in handling unsaturated demands is estimating $Q(m)$, which is the probability that m has data to send when its backoff counter is 0 and the channel is clear at m. With saturated demands, $Q(m)$ has a constant value of 1, but with unsaturated demands it must be computed to ensure that the traffic demands d_m are not exceeded. Computing $Q(m)$ is difficult due to strong interdependency among nodes. Specifically, $Q(m)$ depends on how often the channel is clear at m, which depends on the amount of traffic generated by other nodes and thus their Q values, which in turn depends on the traffic generated by m and $Q(m)$.

We develop an iterative algorithm to compute Q. The algorithm initializes Q to 1 for all senders. In each iteration, the algorithm first derives the transition matrix M based on the old Q values and computes the stationary probabilities π_i and the achieved throughput $t_m^{\text{prev}} = \sum_{i:m \in S_i} \pi_i$. It then updates Q based on their values in the previous iteration. For this, we use the following relationships:

$$\frac{Q^{\text{prev}}(m) \times T_\mu(m)}{Q^{\text{prev}} \times T_\mu(m) + T_{\text{gap}}(m)} = t_m^{\text{prev}} \tag{9}$$

$$\frac{Q(m) \times T_\mu(m)}{Q(m) \times T_\mu(m) + T_{\text{gap}}(m)} \leq d_m \tag{10}$$

$$Q(m) \leq 1 \tag{11}$$

where $Q^{\text{prev}}(m)$ is the value of $Q(m)$ in the previous iteration, $T_{\text{gap}}(m)$ is the average time gap between two consecutive transmissions from m, and $T_\mu(m)$ is the average packet transmission time of m.

Equation (9) captures the relationship between $Q(m)$ and the m's sending rate in the previous iteration. Equation (10) ensures that the total amount of traffic sent by m does not exceed its demand d_m. We then compute $Q^{\text{new}}(m)$ as the largest possible $Q(m)$ value that satisfies all three constraints (9), (10) and (11). This gives:

$$Q^{\text{new}}(m) = \min\left\{1,\ Q^{\text{prev}}(m)\frac{d_m}{1-d_m}\frac{1-t_m^{\text{prev}}}{t_m^{\text{prev}}}\right\}. \tag{12}$$

At this point, we could directly use $Q^{\text{new}}(m)$ as our estimate of $Q(m)$ for the next iteration. For quick convergence, we apply a relaxation procedure that is commonly used in equilibrium computation [15]. Specifically, we set the new $Q(m)$ to be a linear combination of $Q^{\text{new}}(m)$ and $Q^{\text{prev}}(m)$: $Q(m) = \alpha \times Q^{\text{new}}(m) + (1-\alpha) \times Q^{\text{prev}}(m)$. Our evaluation uses $\alpha = 0.9$, though we find that the model converges quickly for a wide range of α.

4.1.3 Enhancing Scalability

The general sender model, as presented earlier, requires 2^N states and $2^N \times 2^N$ state transitions for N senders. To enhance scalability, we use two techniques that prune the states and transitions. First, we prune all those states that involve too many synchronized transmissions, which should occur with low probability. Specifically, given state i and the corresponding S_i, we eliminate i if the number of edges in the corresponding synchronization graph $\mathcal{G}_{\text{syn}}(S_i)$ exceeds a given threshold, which is set to 1 in our evaluation. Second, we prune all those state transitions whose transition probabilities are too low. Specifically, we reset the transition probability $M(i,j)$ to 0 if it falls below a threshold, which is set to 0.001 in our evaluation. For example, under common 802.11 settings, we can eliminate transitions that involve ≥ 2 synchronization groups exiting the transmission mode together. By pruning unlikely transitions, we can reduce the number of non-zero entries in M, thus improving the efficiency of sparse linear solvers such as *lsqr* in computing the stationary probabilities. The combination of these two techniques is highly effective. For example, consider 10 senders in a 5×5 grid topology, where any two direct horizontal, vertical, or diagonal neighbors can hear each other. Without pruning, the transition matrix has 1024 states and more than a million transitions. After pruning, it has only 370 states and 1736 transitions.

4.2 Receiver Model

We now present our receiver model for broadcast traffic. Our goal is to estimate the goodput g_{mn} (*i.e.*, the receiving rate). We have $g_{mn} = \eta \times t_m \times (1 - L_{mn})$, where L_{mn} is the packet loss rate from m to n, and $\eta = \frac{T_{\text{payload}}}{T_{\text{payload}} + T_{\text{header}} + T_{\text{preamble}}}$ represents the fraction of packet transmission time for the payload (excluding header and preamble overhead).

A key challenge in estimating L_{mn} is how to translate slot-level loss rates (derived from our *slot-level* Markov chain) to packet loss rates. Our experiments show that slot-level loss rates (*i.e.*, the fraction of time slots in which loss occurs) can be quite different from packet loss rates. For example, when loss comes from hidden terminals, where senders do not sense each other and cause collisions, a packet is usually corrupted partially. In this case, the packet loss rate can be significantly higher than slot-level loss rate. Consider transmission of 10 packets, which contain altogether 1000 time slots. Even if only around 10% slots (100 slots) are lossy, they can cause a packet loss rate as high as 100% if these lossy slots are distributed across all packets. Below we first analyze the slot-level loss rates and then convert them into packet loss rates.

4.2.1 Estimating Slot-Level Loss Probabilities

We first estimate the slot-level loss rate from broadcast sender m to receiver n in state i ($m \in S_i$). Let $I_{mn|S_i} = W_n + B_n + \sum_{t \in S_i \setminus \{m\}} R_{tn}$ be the total interference at receiver n. Note that we allow $t = n$ because n and m may be transmitting at the same time. At the slot level, a loss occurs whenever the SINR falls below δ_n and/or the RSS falls below γ_n. Let $\ell_{mn|S_i} = \Pr\{\frac{R_{mn}}{I_{mn|S_i}} < \delta_n\}$ be the slot-level loss rate caused by low SINR in state i. Let $\ell^{\text{rss}}_{mn} = \Pr\{R_{mn} < \gamma_n\}$ be the slot-level loss rate caused by low RSS.

Computing $\ell_{mn|S_i}$: Similar to Section 4.1, we approximate $I_{mn|S_i} = W_n + B_n + \sum_{t \in S_i \setminus \{m\}} R_{tn}$ with a single moment-matching lognormal random variable e^Z, where $Z \sim N(\mu, \sigma^2)$. Since the ratio of two independent lognormal random variables R_{mn} and e^Z is also a lognormal random variable, let $\mathrm{e}^{Z'} = \frac{R_{mn}}{\mathrm{e}^Z}$, where $Z' \sim N(\mu', \sigma'^2)$. We have $\mu' = E[\log R_{mn}] - \mu$ and $\sigma'^2 = Var[\log R_{mn}] + \sigma^2$. Thus:

$$\ell_{mn|S_i} = \Pr\left\{\frac{R_{mn}}{I_{mn|S_i}} < \delta_n\right\} \approx \Pr\{\mathrm{e}^{Z'} < \delta_n\} = \Phi\left[\frac{\log \delta_n - \mu'}{\sigma'}\right] \quad (13)$$

Computing $\ell^{\text{rss}}_{mn|S_i}$: There are two ways to compute $\ell^{\text{rss}}_{mn|S_i}$. When the distribution of R_{mn} is known, we can directly compute $\ell^{\text{rss}}_{mn|S_i} = \Pr\{R_{mn} < \gamma_n\}$. In practice, R_{mn} has to be estimated and is subject to estimation error. To minimize error, we observe that when there is only a single sender and no external interference, all losses are due to low RSS. Specifically, $1 - (1 - \ell^{\text{rss}}_{mn|S_i})^{T_\mu(m)/T_{\text{slot}}}$ gives the packet loss rate (assuming independent slot-level loss within a packet). Thus we can directly use the measured packet loss rate under transmissions from a single sender m to estimate $\ell^{\text{rss}}_{mn|S_i}$.

4.2.2 Computing Packet Loss Probabilities L_{mn}

Packet losses can be broadly divided into three categories. First, packet losses can stem from low RSS. Second, packet losses can stem from collision with packets from the same synchronization group. In this case, the fraction of lost packets is close to the fraction of lost slots. Third, packet losses can stem from collisions with asynchronous transmissions (*e.g.*, from hidden terminals). In this case, the packet loss rate can be much higher than the slot-level loss rate. Let L^{rss}_{mn}, L^{syn}_{mn}, and L^{asyn}_{mn} denote the probabilities for these three types of packet losses, respectively. Assuming independence among them, the combined packet loss rate is:

$$L_{mn} = 1 - (1 - L^{\text{rss}}_{mn}) \times (1 - L^{\text{syn}}_{mn}) \times (1 - L^{\text{asyn}}_{mn}) \quad (14)$$

Computing L^{rss}_{mn}: We estimate L^{rss}_{mn} as $1 - (1 - \ell^{\text{rss}}_{mn})^{T_\mu(m)/T_{\text{slot}}}$. Note that when measured packet loss rate from a single sender m is available, we can directly use it as L^{rss}_{mn} without converting it to ℓ^{rss}_{mn}.

Computing L^{syn}_{mn} and L^{asyn}_{mn}: Let $SS(m)$ be the set of all those states whose synchronization graph has at least one edge involving m:

$$SS(m) \stackrel{\triangle}{=} \{i \mid m \in S_i \wedge \mathcal{G}(S_i) \text{ has an edge involving } m\} \quad (15)$$

Let the synchronous and asynchronous slot-level loss rates be:

$$\ell^{\text{syn}}_{mn} = \frac{\sum_{i \in SS(m)} \pi_i \ell_{mn|S_i}}{\sum_{i:\, m \in S_i} \pi_i}, \text{ and } \quad \ell^{\text{asyn}}_{mn} = \frac{\sum_{i \notin SS(m)} \pi_i \ell_{mn|S_i}}{\sum_{i:\, m \in S_i} \pi_i} \quad (16)$$

To estimate L^{syn}_{mn} from ℓ^{syn}_{mn}, we simply set $L^{\text{syn}}_{mn} = \ell^{\text{syn}}_{mn}$.

To estimate L^{asyn}_{mn} from ℓ^{asyn}_{mn}, we assume that packet losses are generated by collision of foreground traffic (from m to n) and background traffic that arrive independently. We further model background traffic as an ON/OFF process with exponentially distributed ON and OFF periods. Let $\overline{T}_{\text{on}}$ and $\overline{T}_{\text{off}}$ denote the average durations of the two periods, respectively.

Under the above assumptions, the slot-level loss rate for the foreground traffic should be equal to the fraction of time that the background traffic is in ON periods. That is:

$$\frac{\overline{T}_{\text{on}}}{\overline{T}_{\text{on}} + \overline{T}_{\text{off}}} = \ell^{\text{asyn}}_{mn} \quad (17)$$

Assuming $\overline{T}_{\text{on}} = T_\mu$, the above equation yields $\overline{T}_{\text{off}} = \frac{1 - \ell^{\text{asyn}}_{mn}}{\ell^{\text{asyn}}_{mn}} T_\mu$.

A packet is successfully received if it starts in a background OFF period and the rest of this background OFF period lasts at least the packet transmission time. With exponentially distributed background OFF periods, we have:

$$1 - L^{\text{asyn}}_{mn} = \frac{\overline{T}_{\text{off}}}{\overline{T}_{\text{on}} + \overline{T}_{\text{off}}} \times \exp\left[-\frac{T_\mu}{\overline{T}_{\text{off}}}\right] \quad (18)$$

$$= (1 - \ell^{\text{asyn}}_{mn}) \times \exp\left[-\frac{\ell^{\text{asyn}}_{mn}}{1 - \ell^{\text{asyn}}_{mn}}\right] \quad (19)$$

where the first term on the right hand side of Equation (18) is the probability that the packet transmission starts in a background OFF period and the second term is the probability that the rest of this background OFF period lasts for at least T_μ.

5. UNICAST TRAFFIC

In this section, we extend our broadcast models to handle unicast traffic. There are two key differences between unicast and broadcast transmissions. On the sender side, the transition matrix M is different under unicast due to additional ACK overhead and exponential backoff. On the receiver side, there are additional losses due to ACKs colliding with both data and other ACKs. We present the sender side extensions followed by the receiver side extensions.

5.1 Extensions to Sender Model

The transition matrix for the sender, in particular, $\overline{CW}(m)$, $\overline{OH}(m)$ and $Q(m)$ in Equation (2) are different for unicast traffic.

Computing $\overline{CW}(m)$: We derive $\overline{CW}(m)$ from packet loss rate L_{mn} across all receivers n as follows. Let $H(L)$ be the average contention window under packet loss rate L, and $RMAX$ be the maximum number of retransmissions. Then we have:

$$H(L) = \sum_{k=0}^{RMAX} \frac{\min\{(CW_{\min}+1)2^k - 1, CW_{\max}\}}{2} L^k \quad (20)$$

A sender may transmit to more than one receiver, each with a different loss rate. We estimate $\overline{CW}(m)$ as the weighted average over all receivers, where the weights are based on the total transmissions to the receivers. For simplicity, we approximate the weight as $\frac{G_{mn} \times d_{mn}}{\sum_r G_{mr} \times d_{mr}}$, where G_{mn} denotes the expected number of transmissions (including the first transmission) for each data packet sent from m to n. Assuming independent packet losses, $G_{mn} = \sum_{k=0}^{RMAX} L^k_{mn}$. Therefore:

$$\overline{CW}(m) = \sum_n \left[H(L_{mn}) \times \frac{G_{mn} \times d_{mn}}{\sum_r G_{mr} \times d_{mr}}\right] \quad (21)$$

Computing $\overline{OH}(m)$: Unicast involves additional overhead from SIFS and ACK. Let T_{SIFS} and T_{ACK} denote the duration for SIFS and ACK. The average overhead for unicast transmissions from m to n is: $\overline{OH}(m,n) = \frac{T_{\text{DIFS}} + T_{\text{SIFS}} + (1 - L_{mn}) T_{\text{ACK}}}{T_{\text{slot}}}$. We then compute $\overline{OH}(m)$ as the the weighted average of $\overline{OH}(m,n)$ over all n:

$$\overline{OH}(m) = \sum_n \left[\overline{OH}(m,n) \times \frac{G_{mn} \times d_{mn}}{\sum_r G_{mr} \times d_{mr}}\right] \quad (22)$$

Computing $Q(m)$: For saturated unicast demands, $Q(m) = 1$. For unsaturated unicast demands, we can update $Q(m)$ in the same way as in Section 4.1.2. The only adjustment we need is to account for retransmission by m. This can be achieved by simply changing the right hand side of Equation (10) from d_m to $\sum_r G_{mr} \times d_{mr}$.

Computing throughput t_{mn}: With the new $\overline{CW}(m)$, $\overline{OH}(m)$, and $Q(m)$, we can compute the transition matrix M and stationary probabilities π_i for unicast traffic. We can then compute the throughput from m to n as $t_{mn} = t_m \times \frac{G_{mn} \times d_{mn}}{\sum_r G_{mr} \times d_{mr}}$, where $t_m = \sum_{i:m \in S_i} \pi_i$.

5.2 Extensions to Receiver Model

Consider node m sending data to node n. Similar to broadcast, we decompose unicast packet loss rate L_{mn} into three components: (a) L^{rss}_{mn} – losses due to low RSS, (b) L^{syn}_{mn} – losses due to collision of synchronized transmissions, and (c) L^{asyn}_{mn} – losses due to collision with asynchronous transmissions. Assuming independence among them, we have: $L_{mn} = 1 - (1 - L^{\text{rss}}_{mn}) \times (1 - L^{\text{syn}}_{mn}) \times (1 - L^{\text{asyn}}_{mn})$.

The key extensions that we make are: (i) extend L^{rss}_{mn} to include RSS induced losses for both data and ACK packets, and (ii) extend $\ell_{mn|S}$ to include SINR induced slot-level loss due to collision between ACK/data, data/ACK, ACK/ACK (in addition to data/data), which can then be used to compute L^{syn}_{mn} and L^{asyn}_{mn} in the same way as in Section 4.2.2. Below we describe these extensions in detail.

Estimating L^{rss}_{mn}: Let $\ell^{\text{rss}}_{mn} = \Pr\{R_{mn} < \gamma_n\}$ and $\ell^{\text{rss}}_{nm} = \Pr\{R_{nm} < \gamma_m\}$. The combined RSS-induced loss on data and ACK is then

$$L^{\text{rss}}_{mn} = 1 - (1 - \ell^{\text{rss}}_{mn})^{T_\mu(m)/T_{\text{slot}}} \times (1 - \ell^{\text{rss}}_{nm})^{T_{\text{ACK}}(n)/T_{\text{slot}}}$$

where $T_{\text{ACK}}(n)$ is the duration of an ACK sent by n.

Estimating $\ell_{mn|S_i}$: We consider the following three cases of low SINR induced slot-level loss:

- C1: *Data loss due to collision with other data.* Data sent from m to n can get lost due to collision with data sent by other nodes in $S_i \setminus \{m\}$. This is identical to the broadcast case and the slot-level loss rate is $\ell^{\text{C1}}_{mn|S_i} = \Pr\{\frac{R_{mn}}{I^{\text{C1}}_{mn|S_i}} < \delta_n\}$, where $I^{\text{C1}}_{mn|S_i} = W_n + B_n + \sum_{t \in S_i \setminus \{m\}} R_{tn}$.
- C2: *Data loss due to collision with other ACKs and data.* In this case, a synchronization group G ($m \notin G$) exits the transmission mode while all nodes in $S_i \setminus G$ continues transmitting. The ACKs generated by receivers of senders in G could collide with data from m to n. To quantify such effects, let random variable $R_{\text{ack}}(m,n|S_i)$ denote the interference at n that is caused by ACKs sent by the receiver of sender m after m stops transmitting in state i (which we will analyze below). The slot-level loss rate caused by other ACKs and data is thus: $\ell^{\text{C2}}_{mn|S_i}(G) = \Pr\{\frac{R_{mn}}{I^{\text{C2}}_{mn|S_i}} < \delta_n\}$, where $I^{C2}_{mn|S_i} = W_n + B_n + \sum_{t \in S_i \setminus (G \cup \{m\})} R_{tn} + \sum_{t \in G} R_{\text{ack}}(t,n|S_i)$ is the total interference at n when m is transmitting data to n.
- C3: *ACK loss due to collision with other ACKs and data.* In this case, the synchronization group that m belongs to (denoted by G_m) exits the transmission mode while all nodes in $S_i \setminus G_m$ continue transmitting. ACKs sent by receivers of senders in $G_m \setminus \{m\}$ combined with data sent by nodes in $S_i \setminus G_m$ could collide with ACKs sent from n to m. The resulted slot-level loss rate is $\ell^{\text{C3}}_{mn|S_i} = \Pr\{\frac{R_{nm}}{I^{\text{C3}}_{nm|S_i}} < \delta_n\}$, where $I^{\text{C3}}_{nm|S_i} = W_m + B_m + \sum_{t \in S_i \setminus G_m} R_{tm} + \sum_{t \in G_m \setminus \{m\}} R_{\text{ack}}(t,m|S_i)$ is the total interference at sender m when receiver n is transmitting an ACK to m.

Among the above three cases, C2 (with different G) and C3 are *mutually exclusive* because only one G can be the first synchronization group that stops transmitting. Note that with our pruning strategies described in Section 4.1.3, we need not consider having two groups stop transmitting at the same time (because the transition probability would become too small).

For a given G, the probability for G to be the first group that stops transmitting in state i is $\frac{M(i,i'(G))}{\sum_{j \neq i} M(i,j)} = \frac{M(i,i'(G))}{1-M(i,i)}$, where $i'(G)$ is the resulted state after G stops transmission in state i. The combined loss rate for C2 and C3 can then be computed as:

$$\ell^{\text{C2+C3}}_{mn|S_i} = \left[\sum_{G: m \notin G} \frac{M(i,i'(G))}{1-M(i,i)} \ell^{\text{C2}}_{mn|S_i}(G) \right] + \frac{M(i,i'(G_m))}{1-M(i,i)} \ell^{\text{C3}}_{mn|S_i} \quad (23)$$

Assuming independence between C1 and C2/C3, the combined slot-level loss rate in state i can be computed as:

$$\ell_{mn|S_i} = 1 - (1 - \ell^{\text{C1}}_{mn|S_i}) \times (1 - \ell^{\text{C2+C3}}_{mn|S_i}) \quad (24)$$

The only remaining task is to analyze $R_{\text{ack}}(m,n|S_i)$. The main challenge is that m may have multiple receivers and RSS from different receivers are different. To address this, for each sender we compute the weighted average of interference that its receivers generate, where the weights are based on the traffic demands and delivery probabilities to the receivers. Specifically, we approximate the distribution of $R_{\text{ack}}(m,n|S_i)$ using a log-normal distribution, whose mean $\bar{\mathsf{R}}_{\text{ack}}(m,n|S_i)$ and variance $\mathsf{R}^{\text{var}}_{\text{ack}}(m,n|S_i)$ are computed as:

$$\bar{\mathsf{R}}_{\text{ack}}(m,n|S_i) = \sum_r \left[\frac{d_{mr}}{\sum_{r'} d_{mr'}} \times p^{\text{ack}}_{mr|S_i} \times \bar{\mathsf{R}}_{rn} \right] \quad (25)$$

$$\mathsf{R}^{\text{var}}_{\text{ack}}(m,n|S_i) = \sum_r \left[\frac{d_{mr}}{\sum_{r'} d_{mr'}} \times p^{\text{ack}}_{mr|S_i} \times \mathsf{R}^{\text{var}}_{rn} \right] \quad (26)$$

where $p^{\text{ack}}_{mr|S_i}$ denotes the probability for a transmission from sender m to successfully trigger an ACK from its receiver r in state i. To simplify the analysis of $p^{\text{ack}}_{mr|S_i}$, we ignore data loss due to collision with other ACK and only consider data loss caused by either low RSS or collision with other data. With such simplification, we can approximate $p^{\text{ack}}_{mr|S_i} \approx (1 - \ell^{\text{C1}}_{mn|S_i}) \times (1 - \ell^{\text{rss}}_{mn})^{T_\mu(m)/T_{\text{slot}}}$.

Finally, once all the loss rates L_{mn} are available and t_{mn} has been computed, we can compute the goodput as $g_{mn} = \eta \times t_{mn} \times \frac{1-L^{RMAX+1}_{mn}}{G_{mn}}$, where G_{mn} is the average number of transmissions per data packet, $(1 - L^{RMAX+1}_{mn})$ gives the packet delivery rate (after the initial transmission and *RMAX* retransmissions), and η is used to exclude the header and preamble overhead.

5.3 Putting It Together

Unlike for broadcast traffic, there is a tight coupling between the sender model and receiver model for unicast traffic. Specifically, in order to compute $\overline{CW}(m)$ and $\overline{OH}(m)$ before deriving the transition matrix M and the stationary probabilities π_i, the sender model requires the knowledge of packet loss rates L_{mn} (as described in Section 5.1). Meanwhile, the receiver model needs to know π_i in advance in order to convert slot-level loss rates into packet loss rates L_{mn} (as described in Section 5.2 and Section 4.2.2).

To break such inter-dependency, we apply an iterative framework to progressively refine our model. As summarized in Figure 1, we initially set $L_{mn} = 0$. During each iteration, we first apply the sender model to update $\overline{CW}(m)$, $\overline{OH}(m)$, M and π_i based on L_{mn} from the previous iteration. We then use the updated π_i to compute the new L_{mn}. For unsaturated unicast demands, we also iteratively update $Q(m)$ (which are initialized to 1). For quick convergence, we apply the relaxation procedure in Section 4.1.2 to update L_{mn} and $Q(m)$. Convergence is reached when the relative changes in L_{mn} and $Q(m)$ become small enough. In our evaluation, we find that the model always converges quickly within 10 iterations.

initialization: $L_{mn} = 0$ $(\forall m \forall n)$; $Q(m) = 1$ $(\forall m)$; *converged* = **false**
while (**not** *converged*)
  // sender model: see Section 5.1 and Section 4.1
  compute $\overline{CW}(m)$, $\overline{OH}(m)$ using L_{mn}
  derive transition matrix M from $\overline{CW}(m)$, $\overline{OH}(m)$ and $Q(m)$
  compute stationary probabilities π_i using M
  compute $Q^{\text{new}}(m)$ based on π_i and previous $Q(m)$
  // receiver model: see Section 5.2 and Section 4.2
  compute packet loss rates L_{mn}^{new} from slot-level loss rates using π_i
  // relaxation for quick convergence (currently we set $\alpha = 0.9$*)*
  $L_{mn} = \alpha \times L_{mn}^{\text{new}} + (1-\alpha) \times L_{mn}$
  $Q(m) = \alpha \times Q^{\text{new}}(m) + (1-\alpha) \times Q(m)$
  // test for convergence
  converged = **true** if changes in L_{mn} and $Q(m)$ are small enough
end

Figure 1: An iterative framework for modeling unicast traffic.

6. OBTAINING MODEL INPUTS

In this section, we describe how we obtain the various inputs to our model. To estimate pairwise RSS and the external interference at each node, namely R_{mn} and B_n, we measure RSSI at n when only m is transmitting. We only require $O(N)$ measurements because wireless is broadcast medium and all receivers can measure RSSI when a node transmits. From Reis *et al.* [24], $RSSI_{mn} = 10\log_{10}(\frac{R_{mn}+B_n}{W_n})$. For simplicity, we assume $B_n = 0$. (This holds when interference from external transmitters is negligible.) We then estimate R_{mn} by finding a log-normal distribution that best fits the measured RSSI. Let $\bar{\mathsf{R}}_{mn}$ and $\mathsf{R'}_{mn}^{\text{var}}$ denote the mean and variance of the best fitting log-normal distribution. The final RSS distribution is estimated as a log normal distribution with mean of $\bar{\mathsf{R}}_{mn}$ and variance of $\mathsf{R'}_{mn}^{\text{var}} \times \frac{T_{\text{preamble}}}{T_{\text{slot}}}$. We estimate RSS variance as $\mathsf{R'}_{mn}^{\text{var}} \times \frac{T_{\text{preamble}}}{T_{\text{slot}}}$ because we are interested in RSS variation in the time scale of slots while RSSI is measured as an average over the preamble period and $\mathsf{R'}_{mn}^{\text{var}}$ is $\frac{T_{\text{slot}}}{T_{\text{preamble}}}$ of the slot-level RSS variance.

As mentioned in Section 4.2.1, when the RSS distribution is available, we can estimate $\Pr\{R_{mn} < \gamma_n\}$ immediately from the distribution. In practice, because RSSI measurements are only available on received packets, estimating the true RSS distribution is hard. To get around the problem, we can estimate $\Pr\{R_{mn} < \gamma_n\}$ by directly computing the loss rate (*i.e.*, the fraction of packets that are lost) using the RSSI measurement data.

We find that when the delivery rate is too low (*e.g.*, below 10%), computing the mean and variance of RSS based on RSSI measurements yields significant bias because RSSI measurements are only available on received packets. Accurately estimating the true R_{mn} under such cases is an interesting subject on its own, and we leave it as future work. In our current testbed evaluation, we consider only the sender groups such that every node pair m and n within the sender group has either $L_{mn} \leq 90\%$ or $L_{mn} = 100\%$. For fair comparison with the UW model, in all 2-sender evaluation we do not apply the above filtering, and compare the estimated and actual values over all sender groups.

Our model also requires the values of a few radio-dependant constants. For testbed experiments, based on our hardware, we use -95 dBm as thermal noise, 2.5 dB as SINR threshold, and -85 dBm as CCA threshold. For simulation experiments, we use the default values in Qualnet, where the thermal noise is -92.52 dBm in 802.11a and -102.5191 dBm in 802.11b, SINR threshold is 2.5 dB, and CCA threshold is -85 dBm in 802.11a and -93 dBm in 802.11b.

7. SIMULATOR-BASED EVALUATION

We evaluate the accuracy of our model in both simulation and testbed settings. These two evaluation methodologies are complementary. Testbed experiments allow us to quantify accuracy in more realistic scenarios which are subject to fluctuation in the RF environment, measurement errors, and variations across real hardware. Simulation offers a more controlled environment and allows us to more comprehensively assess the accuracy of individual components in our model. Many simplifying assumptions in our model relate to the interaction of the MAC protocol, and any inaccuracy due to these assumptions impacts the simulator results as well.

7.1 Qualnet Modifications

We use Qualnet 3.9.5 for our evaluation. It has been shown to provide a relatively accurate and realistic simulation environment [27]. We make the following modifications to the Qualnet.

Correct desynchronization problem: The IEEE 802.11 standard states that when the medium is busy at any time during a backoff slot, the backoff procedure must be suspended without decreasing the value of the backoff timer. However in Qualnet, the backoff timer is decremented by propagation delay and causes time desynchronization. Such desynchronization results in an unrealistically low collision ratio, as reported in [3] and confirmed by our results. We fix the problem by ensuring that the backoff timer is not decremented when the medium is busy at any instant within a time slot.

Disable EIFS: According to the IEEE 802.11 standard, in DCF a frame transmission must use EIFS whenever a frame transmission begins but does not result in the correct reception of a complete MAC frame. However several research papers [19, 3] report that EIFS results in unfairness, and suggests disabling EIFS by setting EIFS duration to the same value as DIFS. Existing chipsets such as Atheros also have a configurable EIFS duration. We use the above method to disable EIFS, and leave modeling EIFS for future work.

Modify capture effects: In Qualnet, a receiver accepts frames with stronger signals only when they arrive earlier than reception of other frames. Recently, Kochut *et al.* [13] report that real wireless cards accept frames with stronger signals even if they arrive after reception has started. Therefore, we modify Qualnet to accept frames with stronger signals regardless of whether they arrive earlier or later than reception of other frames. In contrast to modifications used in [13], we also accept frames that arrive after preamble of the frame being received. This simplifies our model, and we plan to explore a detailed model of capture effects in our future work.

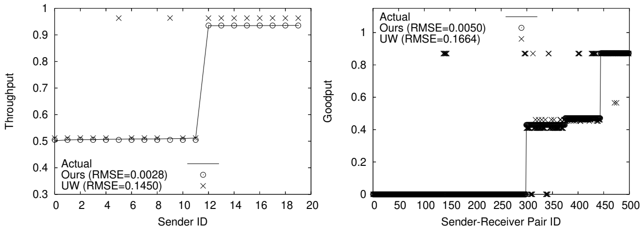
Support SINR model: The 802.11 implementation in Qualnet uses a Bit-Error-Rate (BER) model, where it computes SINR of the current packet, uses the SINR to determine the BER, and then converts the BER to the packet loss rate. To match Qualnet simulation, our model needs the same BER table as in Qualnet. However Qualnet source code does not reveal the BER table it uses for 802.11. To ensure consistency between our model and Qualnet, we implement the commonly used SINR model in both Qualnet and our model. If the BER table becomes available, our model can immediately support BER model by using BER table to map SINR to loss rate.

7.2 Evaluation Methodology

We evaluate our model for both broadcast and unicast by varying the number of simultaneous senders, the frequency band, and the network topologies. We consider both saturated demands and unsaturated demands. The demand is normalized by the physical layer data rate, and a sender with saturated demand has demand 1.

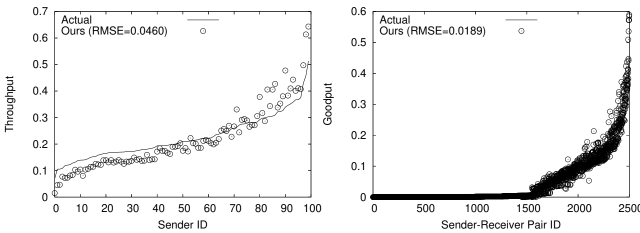
Throughout the evaluation, we use 25-node topologies. Senders generate 1024-byte UDP packets at a constant bit rate (CBR). The actual sending rate to the air may not be constant, however, due to variable contention delay. We use the lowest MAC data rates, *i.e.*, 6Mbps in 802.11a and 1Mbps in 802.11b. The communication ranges of 802.11a and 802.11b with the lowest data rates are 169 m and 348 m, respectively.

For each scenario, we conduct 10 random runs, where each run randomly selects the senders and receivers and the demands. We quantify the accuracy of our model by comparing with the actual



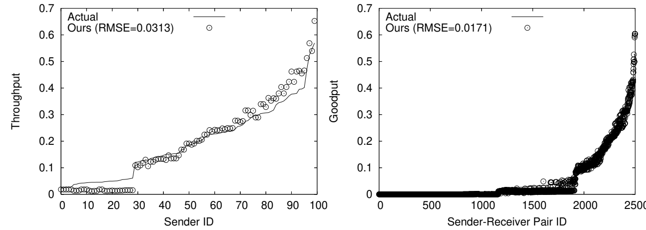
(a) throughput (b) goodput

Figure 2: 2 saturated broadcast senders using 802.11a in a 5×5 grid topology over an $300m \times 300m$ area.



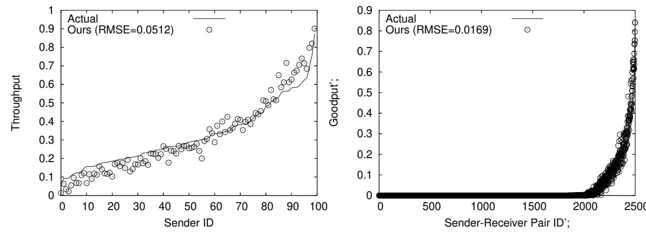
(a) throughput (b) goodput

Figure 3: 10 saturated broadcast senders using 802.11a in a 5×5 grid topology over an $300m \times 300m$ area.



(a) throughput (b) goodput

Figure 4: 10 saturated broadcast senders using 802.11b in a 5×5 grid topology in an $500m \times 500m$ area.



(a) throughput (b) goodput

Figure 5: 10 saturated broadcast senders using 802.11a in a 5×5 grid topology in an $500m \times 500m$ area.

normalized throughput and goodput (defined in Section 3) and computing the root mean square error (RMSE). RMSE is defined as $\sqrt{\frac{\sum_i (est_i - actual_i)^2}{P}}$, where P is total number of predictions. We also study the accuracy in detail using scatter plots of actual and estimated values. For clarity, in the scatter plots the data points are plotted in the increasing order of actual values.

We consider the following scenarios below: (i) 2 broadcast senders with saturated demands; (ii) N broadcast senders with saturated demands; (iii) N broadcast senders with unsaturated demands; (iv) N unicast senders with saturated demands; and (v) N unicast senders with unsaturated demands.

For the first scenario, we compare our model with both Qualnet simulation and UW model [24]. The UW model predicts the impact of interference in the presence of two broadcast senders with saturated demands. It is seeded using $O(N)$ measurements similar to ours – each node takes turn to broadcast packets and other nodes log RSSIs and packet delivery rate. Each node obtains its RSSI versus delivery rate profile using these measurements. To predict the impact of two senders trying to send simultaneously, it first estimates the probability with which senders defers based on the RSSIs they receive from each other. To estimate a receiver's goodput from a sender, it uses the standard SINR model, while treating transmissions from the second sender as additional interference at the receiver. Since there are no existing models for the other scenarios, we compare our model only with the actual values obtained in Qualnet.

7.3 Broadcast Traffic

We begin our evaluation by studying broadcast traffic, starting with the simple case of two senders with saturated demands.

7.3.1 Two Saturated Senders

Figure 2 shows the accuracy of throughput and goodput estimates of our model and the UW model. The graphs plot the actual values obtained in Qualnet and the predictions of the two models. The legend contains the RMSE values for the two models.

We see that both models perform well overall, though our model is more accurate. The RMSE in our model is within 0.005 while that of the UW model is 0.145 or more. The UW model also tends to have highly inaccurate predictions for a few cases.

The error in UW model is mainly because it assumes that a packet can be received as long as its SINR exceeds the threshold. It ignores the other condition that RSS should exceed the radio sensitivity for a packet to be received. For example, when there is only thermal noise (-95 dBm) and RSS is above -92.5, SINR would be above the 2.5 dB threshold. The UW model would thus predict 100% packet delivery. In reality, however, when RSS is between -92.5 dBm and -85 dBm (the radio sensitivity value for 802.11a in Qualnet), the delivery rate is in fact 0. Unfortunately, there is no simple extension to the UW model to accommodate the radio sensitivity constraint because the model builds RF profile directly based on delivery rate. With the radio sensitivity constraint, there is no direct translation between R_{mn} and delivery rate since their relationship changes from $Pr\{\frac{R_{mn}}{I_n+W_n} \geq \delta_n\}$ to $Pr\{R_{mn} \geq \gamma_n \wedge \frac{R_{mn}}{I_n+W_n} \geq \delta_n\}$. For a given delivery rate, R_{mn} is no longer unique.

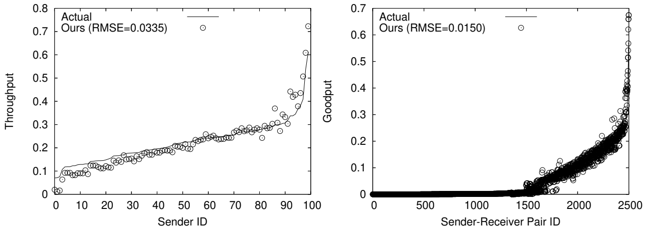
7.3.2 N Saturated Senders

Next we consider the case of N broadcast senders. Each sender has saturated demand, as before. We evaluate our model by varying the frequency band, network topology, and the number of senders.

Different frequency bands (802.11a and 802.11b): Figures 3 and 4 show the scatter plots of actual and predicted throughput and goodput under 802.11a and 802.11b. In each case, there are 10 broadcast senders with infinite demands. We see that our model is highly accurate in both cases, with less than 0.05 RMSE. The goodput error is lower than the throughput error because many receivers have no connectivity to one or more senders, and it is easier to predict the exact goodput for such receivers.

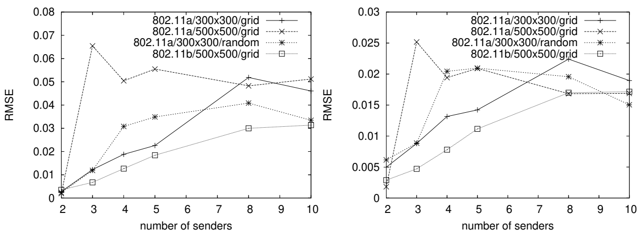
Different network topologies (grid and random): Figure 5 and 6 show the results for 10 broadcast senders using 802.11a in a $500m \times 500m$ grid topology and $300m \times 300m$ random topology, respectively. In each case, the model closely tracks the actual values and the error is around or below 0.05.

Different number of senders (2-10): Figure 7 plots throughput and goodput RMSE as a function of the number of broadcast senders. We see that the error tends to increase slightly with the number of senders due to more complex interactions. Yet under all numbers of senders, the model can keep RMSE within 0.07 for throughput estimation and within 0.025 for goodput estimation.



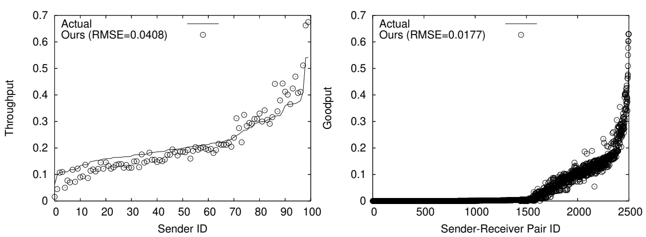
(a) throughput (b) goodput

Figure 6: 10 saturated broadcast senders using 802.11a in random topologies, where nodes are randomly placed in an $300m \times 300m$ area.



(a) throughput (b) goodput

Figure 7: RMSE under a varying number of sender.



(a) throughput (b) goodput

Figure 8: 10 unsaturated broadcast senders using 802.11a in a 5×5 grid topology over an $300m \times 300m$ area.

7.3.3 N Unsaturated Senders

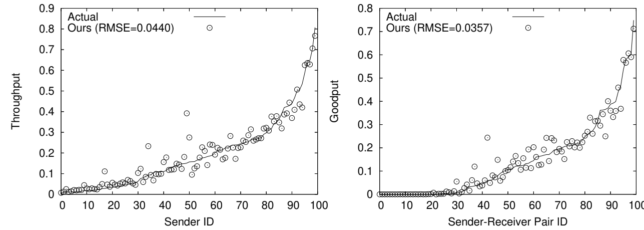
We now consider unsaturated senders and allow nodes to have different traffic demands. We assign each sender a normalized demand between 0.1 and 0.9 and use the corresponding inter-arrival time for CBR traffic. Figure 8 shows the results for 10 broadcast senders using 802.11a in a 5×5 grid topology over a $300m \times 300m$ area. We see that the accuracy of our model for unsaturated demands, which are harder to model, is high as well and comparable to its accuracy for saturated demands.

7.4 Unicast Traffic

In this section, we turn our attention to unicast traffic and evaluate how well the unicast extensions of our model perform.

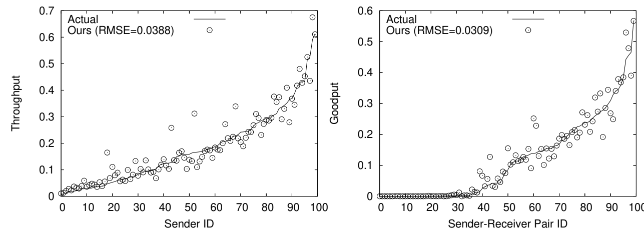
***N* saturated senders:** We start with the case of *N* saturated senders. Figure 9 shows the result for 10 unicast senders using 802.11a. As for broadcast traffic, the predictions of our model track the actual values closely, and the RMSE is within 0.05.

***N* unsaturated senders:** We conclude our simulation-based evaluation by studying the case of unsaturated unicast senders. As above, we have 10 senders using 802.11a in a 5×5 grid topology. The demand for each sender is assigned as for the broadcast setting in Section 7.3.3. Figure 10 shows the prediction results for this setting. Our model continues to yield accurate predictions. Not only is the net RMSE under 0.04, but we also do not have individual instances where the predictions of our model are highly inaccurate.



(a) throughput (b) goodput

Figure 9: 10 saturated unicast senders using 802.11a in a 5×5 grid topology over an $300m \times 300m$ area.



(a) throughput (b) goodput

Figure 10: 10 unsaturated unicast senders using 802.11a in a 5×5 grid topology over an $300m \times 300m$ area.

7.5 Summary

In this section, we used simulation to evaluate the accuracy of our model in many diverse settings which include broadcast and unicast traffic, unsaturated and saturated demands, and different number of senders. We find our model's predictions of throughput and goodput are accurate in all the settings that we considered, and its RMSE value is typically under 0.05. We also find that our model, while being more general, is also more accurate than a state-of-art model [24] for the specific case of 2 broadcast senders with saturated demands.

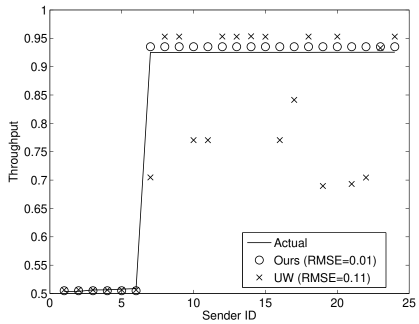
8. TESTBED-BASED EVALUATION

In this section, we evaluate our model using testbed experiments. Our goal is to quantify the accuracy of our model in real RF environments and with real hardware. We employ traces from two different testbeds for this purpose. Below, we describe these testbeds and the traces, followed by the evaluation results for each testbed.

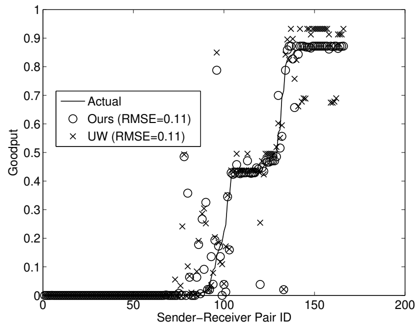
8.1 Testbeds and Traces

The two testbeds are our own indoor wireless testbed and the UW testbed used by Reis *et al.* [24]. Our testbed has 22 DELL dimensions 1100 PCs, located on the same floor of an office building. Each machine has a 2.66 GHz Intel Celeron D Processor 330 with 512 MB of memory, and is equipped with 802.11 a/b/g NetGear WAG511. Each machine runs Fedora Core Linux. We use *Madwifi* as the driver for the wireless cards, and use *click* to collect traces.

We collect the trace as follows. First, we let one node broadcast 1000-byte UDP packets at full speed for 1 minute and log received packets at all the other nodes. We repeat the process until every node in the testbed has broadcast once. We refer to this as 1-sender trace. Applying the approach described in Section 6 to the 1-sender trace gives us estimate of RSS between every pair of nodes and external interference at each node. Since there is a resident 802.11b/g wireless network that causes strong interference, we collect traces using only 802.11a on our testbed. Unless otherwise specified, each node uses 30 mW transmission power.

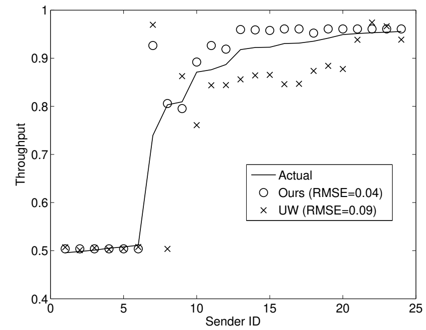


(a) throughput

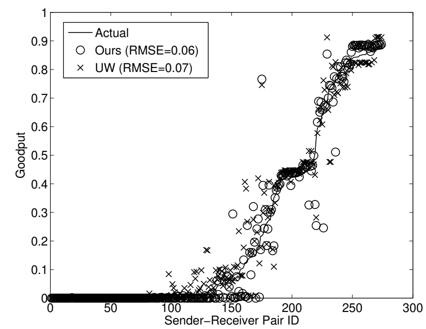


(b) goodput

Figure 11: 2 saturated 802.11a broadcast senders in UW traces.



(a) throughput



(b) goodput

Figure 12: 2 saturated 802.11b broadcast senders in UW traces.

In order to evaluate the accuracy of our model, we measure the actual sending and receiving rates under multiple senders. These traces are only needed for obtaining "ground truth" and not required for using the model. Given a specified number of senders k, we randomly select k nodes and let them broadcast simultaneously for 1 minute. All other nodes log received packets. In the 1-minute broadcasting period, the nodes send as fast as possible for the saturated demand experiments. For unsaturated demands, each sender is assigned a normalized demand which is total demand divided by the physical layer data rate. The normalized demand is selected randomly between 0.1 and 0.9 and specifies the maximum rate at which the sender can send. For each configuration, *i.e.*, the specified number of senders and demand type, we conduct 100 random runs with different set of k senders.

The UW testbed had 14-nodes inside an office building. The traces we use are same as those used for evaluating the UW model [24]. The collection methodology is similar to the above except that these traces contain only 2 broadcast senders with saturated demands. We study both 802.11a and 802.11b using these traces.

8.2 The UW Testbed

We first present the results for the UW testbed in this section and then for our testbed in the next section. Figure 11 shows scatter-plots of predicted and actual throughput and goodput under 802.11a. As we can see, our model closely tracks the actual throughput and goodput. UW model has higher error in the throughput prediction. Most mispredictions occur when the UW model incorrectly predicts that two senders defer to each other. This error is caused by the linear interpolation heuristics to estimate delivery probability for a hypothetical RSSI [24]. The heuristic implicitly assumes delivery probability is linearly proportional to RSSI, which may not hold in reality. Interestingly, UW model has comparable accuracy to our model in goodput prediction. A closer look reveals that for many links that have higher throughput error, their goodput is often close to 0 due to poor link quality. Such cases are easy to predict, which reduces overall goodput error.

Figure 12 shows the results for 802.11b. As for 802.11a, our model has more accurate throughput prediction than the UW model, while both models have comparable prediction errors for goodput.

8.3 Our Testbed

For our testbed, we evaluate our model by varying number of senders and using both saturated and unsaturated demands. Figure 13, 14, 15, and 16 show scatter plots of throughput and goodput under 2, 3, 4 and 5 senders with saturated broadcast demands. Since the UW model is only applicable to 2 senders, we compare with the UW model only for 2 senders. As we can see, our model tracks the actual throughput more closely than the UW model, and yields comparable accuracy for goodput prediction. This is also reflected in RMSE. For 3, 4, and 5-sender cases, our model yields estimation close to the actual rates: its RMSE is within 0.12.

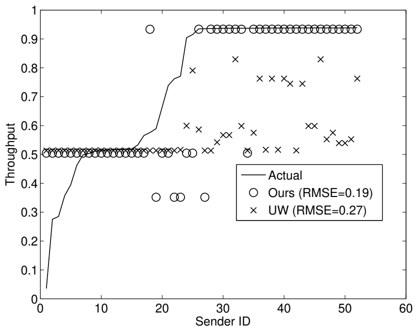
Figure 17 shows the results for 3 senders with unsaturated demands. As for saturated demands, our model maintains high accuracy: its RMSE is within 0.07.

8.4 Summary

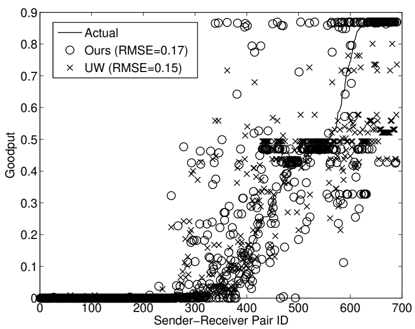
The testbed evaluation confirms that our model works well in real environments and using real hardware. Compared with simulation, predicting testbed performance is much more challenging due to factors such as biased and noisy measurements, as well as variation in RF condition. Despite these challenges, the results show that our model is effective in predicting throughput and goodput.

9. RELATED WORK

Considerable research has been done in the area of modeling wireless networks. Given space constraints, a detailed discussion is not feasible. We thus limit ourselves to a very brief survey to place our work in the overall context. We broadly classify the existing work into three categories. The first category analyzes the performance of IEEE 802.11 Distributed Coordinated Function (DCF) [2, 16, 8, 9]. While these models can estimate interference under an

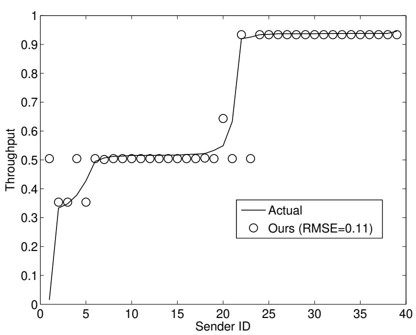


(a) throughput

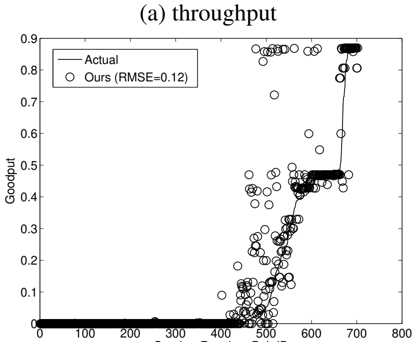


(b) goodput

Figure 13: 2 saturated 802.11a broadcast senders in our traces.



(a) throughput

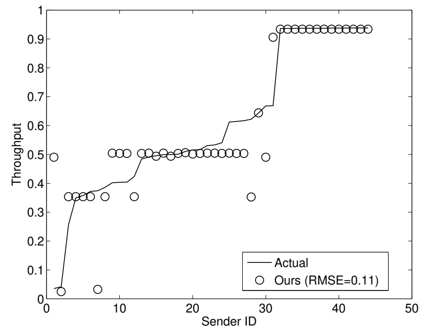


(b) goodput

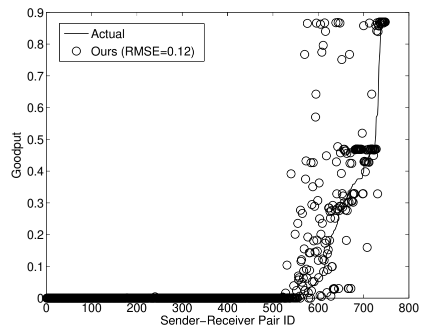
Figure 14: 3 saturated 802.11a broadcast senders in our traces.

arbitrary number of senders, they do not apply to networks where not all nodes can hear each other.

The second category of work targets general network topologies where not all nodes are within communication range [9, 24]. Because of the challenges presented by such topologies, existing models handle only restricted traffic scenarios. Garetto *et al.* develop a two-flow model [9], and Reis *et al.* model two competing broadcast senders [24]. Our work falls into this category and advances

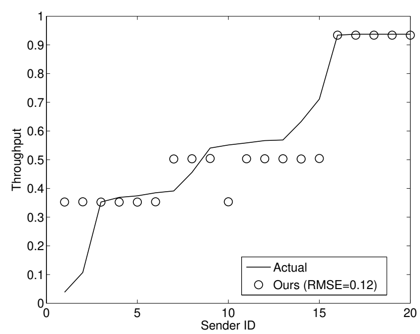


(a) throughput

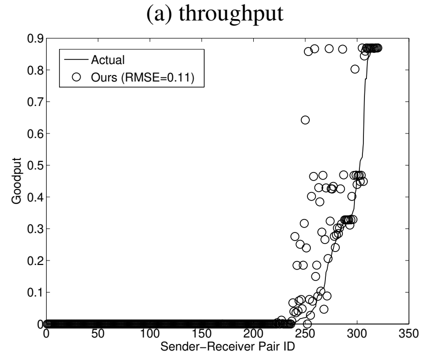


(b) goodput

Figure 15: 4 saturated 802.11a broadcast senders in our traces.



(a) throughput

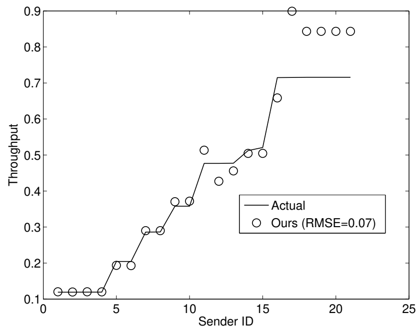


(b) goodput

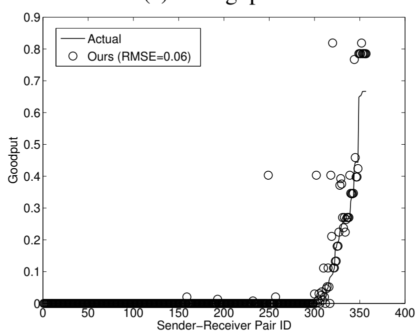
Figure 16: 5 saturated 802.11a broadcast senders in our traces.

the state-of-art by going beyond pairwise interference and modeling interference among an arbitrary number of senders for both broadcast and unicast transmissions.

The third category estimates the end-to-end throughput in multihop wireless networks [10, 11, 17, 8]. Since modeling end-to-end throughput is more difficult than one-hop throughput, to be tractable, such models only apply to specific scenarios. In particular, they either consider asymptotic behavior of wireless net-



(a) throughput



(b) goodput

Figure 17: 3 unsaturated 802.11a broadcast senders in our traces, where each sender uses 1 mW.

works [10], or assume optimal scheduling [11, 17], or are limited to single flow scenarios [8]. While our work focuses on one-hop throughput, because of its generality, we believe it is relatively easy to extend the model to predict end-to-end throughput if routing information is given. We will investigate this in the future.

10. SUMMARY

We developed a general model of wireless interference in static, multihop networks. It advances the state of the art by (i) estimating interference among an arbitrary number of senders, (ii) modeling the more common case of unicast transmissions, and (iii) modeling the general case of heterogeneous nodes with different traffic demands. Our model is seeded using easy-to-gather $O(N)$ measurements in an N-node network. It is based on a Markov chain that models in detail the interaction between different senders and receivers. Using simulations and measurements from two wireless testbeds, we showed that our model is accurate in a wide range of scenarios.

Acknowledgements

We thank Charles Reis for sharing the source code of UW interference model and anonymous reviewers for their feedback. This research is sponsored in part by National Science Foundation grants CNS-0546755, CNS-0627020, CNS-0546720, and CNS-0615104.

11. REFERENCES

[1] S. Agarwal, J. Padhye, V. N. Padmanabhan, L. Qiu, A. Rao, and B. Zill. Estimation of link interference in static multi-hop wireless networks. In *Proc. of Internet Measurement Conference (IMC)*, Oct. 2005.

[2] G. Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. In *IEEE Journal on Selected Areas in Communications*, Mar. 2000.

[3] H. Chang, V. Misra, and D. Rubenstein. A general model and analysis of physical layer capture in 802.11 networks. In *Proc. of IEEE INFOCOM*, Apr. 2006.

[4] Y. Cheng, J. Bellardo, P. Benko, A. C. Snoeren, G. M. Voelker, and S. Savage. Jigsaw: Solving the puzzle of enterprise 802.11 analysis. In *Proc. of ACM SIGCOMM*, Sept. 2006.

[5] D. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proc. of ACM MOBICOM*, Sept. 2003.

[6] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Proc. of ACM MOBICOM*, Sept. - Oct. 2004.

[7] L. F. Fenton. The sum of lognormal probability distributions in scatter transmission systems. *IRE Trans. Commun. Syst.*, CS-8, 1960.

[8] Y. Gao, J. Lui, and D. M. Chiu. Determining the end-to-end throughput capacity in multi-hop networks: Methodolgy and applications. In *Proc. of ACM SIGMETRICS*, Jun. 2006.

[9] M. Garetto, J. Shi, and E. Knightly. Modeling media access in embedded two-flow topologies of multi-hop wireless networks. In *Proc. of ACM MOBICOM*, Aug. - Sept. 2005.

[10] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2), Mar. 2000.

[11] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu. Impact of interference on multi-hop wireless network performance. In *Proc. of ACM MOBICOM*, Sept. 2003.

[12] V. Kawadia and P. R. Kumar. Principles and protocols for power control in ad hoc networks. In *IEEE Journal on Selected Areas in Communications (JSAC)*, Jan. 2005.

[13] A. Kochut, A. Vasan, A. U. Shankar, and A. Agrawala. Sniffing out the correct physical layer capture model in 802.11b. In *Proc. of ICNP*, Oct. 2004.

[14] D. Kotz, C. Newport, and C. Elliott. The mistaken axioms of wireless-network research. Technical Report TR2003-467, Dartmouth College, Computer Science, Jul. 2003.

[15] J. B. Krawczyk and S. Berridge. Relaxation algorithms in finding Nash equilibria. In Computational Economics from Economics Working Paper Archive at WUSTL, Jul. 1997.

[16] A. Kumar, E. Altman, D. Miorandi, and M. Goyal. New insights from a fixed point analysis of single cell IEEE 802.11 wireless LANs. In *Proc. of IEEE INFOCOM*, Mar. 2005.

[17] V. S. A. Kumar and M. Marathe. Algorithmic aspects of capacity in wireless networks. In *Proc. of ACM SIGMETRICS*, Jun. 2005.

[18] J. Li, C. Blake, D. S. J. D. Couto, H. I. Lee, and R. Morris. Capacity of ad hoc wireless networks. In *Proc. of ACM MOBICOM*, Jul. 2001.

[19] Z. Li, S. Nandi, and A. K. Gupta. Improving fairness in IEEE 802.11 using enhanced carrier sensing. In *IEEE Communications*, Oct. 2004.

[20] D. Malone, K. Duffy, and D. Leith. Modeling the 802.11 distributed coordination function in nonsaturated heterogeneous conditions. *IEEE/ACM Transactions on Networking*, 15(1), Feb. 2007.

[21] A. Mishra, V. Brik, S. Banerjee, A. Srinivasan, and W. Arbaugh. A client-driven approach for channel management in wireless LANs. In *Proc. of IEEE Infocom*, Apr. 2006.

[22] L. M. S. C. of the IEEE Computer Society. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. *IEEE Standard 802.11*, 1999.

[23] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Soft.*, 1982.

[24] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Measurement-based models of delivery and interference. In *Proc. of ACM SIGCOMM*, Sept. 2006.

[25] E. Rozner, Y. Mehta, A. Akella, and L. Qiu. Traffic-aware channel assignment in enterprise wireless networks. In *Proc. of ICNP*, Oct. 2007.

[26] S. Schwartz and Y. Yeh. On the distribution function and moments of power sums with lognormal distributions. *Bell Systems Technical Journal*, 61, 1982.

[27] M. Takai, J. Martin, and R. Bagrodia. Effects of wireless physical layer modeling in mobile ad hoc networks. In *Proc. of ACM MOBIHOC*, Oct. 2001.