

Online Optimization of 802.11 Mesh Networks *

Theodoros Salonidis
Thomson
Paris, France
theodoros.salonidis@thomson.net

Georgios Sotiropoulos
ETH Zurich
Zurich, Switzerland
ge.sotiropoulos@gmail.com

Roch Guerin
University of Pennsylvania
Philadelphia, USA
guerin@ee.upenn.edu

Ramesh Govindan
University of Southern
California
Los Angeles, USA
ramesh@usc.edu

ABSTRACT

802.11 wireless mesh networks are ubiquitous, but suffer from severe performance degradations due to poor synergy between the 802.11 CSMA MAC protocol and higher layers. Several solutions have been proposed that either involve significant modifications to the 802.11 MAC or legacy higher layer protocols, or rely on 802.11 MAC models seeded with off-line measurements performed during network downtime.

We introduce a technique for online optimization of 802.11 wireless mesh networks using rate control at the network layer. The technique is based on a lightweight model that characterizes the feasible rates region of an operational 802.11 wireless mesh network. Unlike existing 802.11 modeling approaches, the parameters of this model can be estimated online, incur minimal overhead and can be realized using standard probing mechanisms at the network layer. Using analysis and extensive measurements over a wireless mesh network testbed, we validate the assumptions on which the model is built, and explain the principles behind the choice and estimation of its parameters. The benefits of the model and its solution in terms of fairness, throughput and stability are demonstrated operationally for a range of multi-hop topologies and configurations.

Categories and Subject Descriptors

C.2.1 [Computer Communication Networks]: Network Architecture and Design—*Wireless communication*; C.4 [Performance of Systems]: [Modeling techniques]

General Terms

Algorithms, Experimentation, Measurement, Performance

Keywords

802.11, CSMA, Modeling, Optimization, Wireless Mesh Networks

*This work was undertaken at the Thomson Paris Research Lab, when the second author was an intern and the third and fourth authors were visitors.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CoNEXT'09, December 1–4, 2009, Rome, Italy.

Copyright 2009 ACM 978-1-60558-636-6/09/12 ...\$10.00.

1. INTRODUCTION

802.11 wireless mesh networks deployed in urban and rural areas around the globe enable low cost wireless Internet access and emerging wireless community applications. Despite their widespread use, these networks are plagued by well-known performance problems such as lack of predictability, unfairness or even complete starvation. These problems are due to poor synergy between the 802.11 MAC protocol and higher layers of the protocol stack. Several solutions have been proposed but most require modifications to either the 802.11 MAC protocol or higher layer legacy protocols such as TCP.

In this paper, we advocate a solution that uses end-to-end, optimization based rate control at the *network layer*. Network layer rate control is attractive because it does not require modifications to legacy protocols and can be readily incorporated in today's deployments using traffic shapers and rate limiters that are widely available. Furthermore, an optimization framework makes it possible to dynamically control rates, which enables city operators to run mesh networks at maximum efficiency, or in the near future lets users of community meshes harmoniously co-exist by running decentralized optimization mechanisms. Despite these benefits, such a solution has not been used in existing mesh network deployments. This is in part because when rate control is applied, it is implemented using static rate limiters and ad hoc rules of thumb, which most often result in network under-utilization. This is what we seek to remedy.

The key challenge in optimizing the performance of an 802.11 mesh network through rate control, is the efficient and accurate estimation of the feasibility region, i.e., the rates that can be simultaneously sustained by the network at a given time. There are three dimensions to realizing such an estimation. First, it should be accurate so as to neither severely over-utilize or under-utilize the network. Second, it should have a simple representation that can be easily incorporated in an optimization procedure. Third, it should be performed online, i.e., during network operation. Clearly, it may not be feasible to perfectly meet all these requirements in a real-world wireless mesh network, and trade-offs are unavoidable. Existing approaches [17, 21] are based on detailed models of 802.11 and focus on accuracy at the cost of significant complexity when it comes to incorporating them in optimization procedures, e.g., non-linear representations. They also typically rely on extended measurement periods during network downtime [20, 21, 25, 30].

Our approach to tackling the problem is multi-pronged.

We first introduce a model that represents the 802.11 feasibility region using as parameters the extreme points of a convex hull. This representation provides a simple characterization of the feasi-

bility region that can be readily incorporated into a simple convex optimization procedures that supports a wide range of throughput and fairness objectives. In addition, the use of extreme points bypasses the “exhaustive” exploration of the feasibility region typically required by more detailed 802.11 models. Using analysis and extensive measurements in a wireless mesh testbed, we provide a structured validation of the model over progressively complex network configurations. We articulate the assumptions behind the choices of extreme points combinations, identify limitations of the model, and demonstrate that, although it does not capture the full details of the operation of the 802.11 MAC, it provides in practice a reasonable approximation of the feasibility region.

We then introduce a technique for online computation of the model parameters. The computations rely on a simple interference model and estimation of link capacities, obtained as maximum UDP throughputs when links transmit alone in backlogged mode. Since links are unlikely to operate in this mode during network operation, we introduce a light-weight technique to solve this capacity estimation problem. The technique is based on network layer broadcast probes, and requires $O(N)$ measurements taken *during* network operation, where N is the number of nodes. We show that the technique achieves good accuracy and can operate efficiently at a time scale of a few minutes.

Finally, we demonstrate the benefits and validity of the model by incorporating it into an optimization framework capable of realizing various fairness and efficiency objectives. The realization of these objectives is validated experimentally, while also establishing their relatively low computational overhead. Our results demonstrate that online optimization-based network layer rate control in mesh networks is feasible at a time scale of a few minutes, and that it can help eliminate a number of common performance problems in such networks, e.g., avoid starvation and enforce fairness objectives without sacrificing much throughput.

2. RELATED WORK

There is a large body of literature on 802.11 modeling and wireless optimization. We focus on two categories most relevant to our work.

802.11 throughput prediction. Several models exist for interference estimation and throughput prediction in 802.11 multi-hop networks. Most of them are based on [4], which captures the effect of IEEE 802.11 binary exponential backoff in single-hop networks, and on [6], which captures the effect of carrier sensing in multi-hop networks. These models vary in the accuracy with which they model interference (either based purely on geometry [7, 14, 15, 18] or seeded with actual measurements [20, 21, 25, 30]), and their prediction power (single-hop throughput prediction [15, 18, 20, 23, 25, 30] or multi-hop throughput prediction [14, 21]).

The above models are hard to apply in operational multi-hop 802.11 networks for two reasons. First, many of the models do not provide closed form expressions for throughput. Therefore, to predict optimal multi-hop throughput in such networks, one must exhaustively search through the feasible rate region defined by these models. This search can become prohibitively expensive as the number of flows increases. One exception is the simplified model of [21], which characterizes feasibility using non-linear constraints at the potential cost of reduced accuracy. Second, all existing measurement based models (including [21]) require a separate measurement phase where all links are activated backlogged in specific patterns (individual node activations in [20, 25, 30] and pairwise link activations in [21, 23]). As a result, [20, 25, 30] require $O(N^2)$ and [21, 23] requires $O(L^2)$ measurements (for an L -link network), and each such measurement typically lasts sev-

eral seconds to collect sufficient statistics. In practice, this imposes extended downtime and complicates network operation with additional signaling mechanisms to switch between measurement and regular operation. We note that recently, Ahmed *et al.*, introduced a technique to significantly reduce the time of these measurements in client-AP WLANs [2], but it requires extensive firmware modifications and is not applicable to multi-hop wireless mesh networks.

Relative to these works, we advance measurement-based throughput optimization in multi-hop wireless networks towards online operation.

Optimization and congestion control. Several works have applied utility maximization optimization frameworks to multi-hop wireless networks [9, 11, 34, 37]. They yield analytical performance bounds and distributed algorithms that achieve the optimal solution. However, they assume simplified interference models and MAC protocols like TDMA [9, 37], ALOHA [34], or CDMA [11], for which, unlike 802.11 CSMA MAC, closed-form expressions exist that can be incorporated in an optimization formulation.

Several solutions have also been proposed to the problems that exist between 802.11 and higher layers, e.g., replacing 802.11 with a TDMA MAC protocol [27, 32]. Alternatively, congestion control approaches have been proposed that operate on top of a CSMA MAC protocol. Rangwala *et al.* propose congestion control mechanisms at the transport layer [28], Xu *et al.* [36] use active queue management based on local wireless neighborhood information, and [3, 26, 31] suggest using back-pressure to perform congestion control. All these approaches improve throughput and fairness, but do not allow the specification of well-defined throughput/fairness objectives. Furthermore, most still require changes to either 802.11 MAC [3, 27, 31, 32] or to TCP itself [3, 28, 31].

In contrast, we seek to optimize performance without modifications to the 802.11 MAC or the transport layer. Our approach can be implemented on off-the-shelf 802.11 hardware using simple network layer measurements and traffic shapers that are widely available. It is independent of the other layers and can potentially be used with any MAC layer or transport layer mechanism. By identifying feasible rates, our approach enhances co-operation between the 802.11 MAC and the transport layer. Its rate control operates at a longer time scale than congestion control and masks imperfections of the 802.11 MAC that appear at the transport layer. Furthermore, it can shape traffic to achieve well-defined utility-based throughput/fairness objectives and provide flow isolation.

3. MODEL AND APPROACH

In this section, we first define the feasible rates region of an 802.11 multi-hop wireless network and then introduce a model to estimate it.

3.1 Feasibility Region Definition

The 802.11 MAC protocol can be viewed as a function f that maps link input rates \mathbf{x} (representing traffic load) to link output rates \mathbf{y} (representing throughput) over a time period T : $\mathbf{y} = f^{(T)}(\mathbf{x})$. When \mathbf{x} is unconstrained and nodes are allowed to transmit at the nominal radio bandwidth, the output rate of each link is generally less than its input rate due to collisions, interferences, or poor channel conditions. These manifest themselves either as MAC backoffs and retransmission delays or as network layer packet losses.

In this paper, we explore the problem of *constraining* \mathbf{x} to a vector x^c , by rate-limiting each link in the network, such that the output rate y_l of each link l is equal to its input rate, modulo the inevitable network layer packet loss p_l induced only by channel errors. If it were possible to do this, the output rate on each link l would be related to its input rate as $y_l = x_l^c \times (1 - p_l)$. The set of

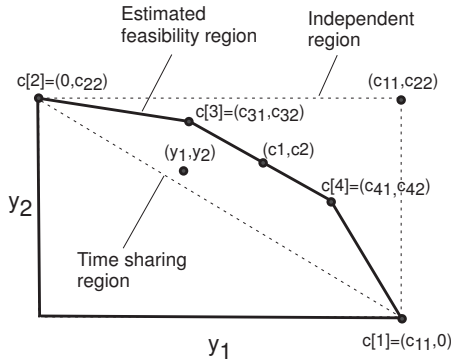


Figure 1: Example of feasibility region estimation by the convex model for $L=2$ links and $K=4$ extreme points.

output rate vectors \mathbf{y} that result from input rates \mathbf{x}^c defines a *feasible rate region*¹ for 802.11. Intuitively, when channel conditions are such that 802.11 MAC operations (power control, retransmissions or adaptation of modulation data rate) are sufficient to recover most channel errors, $y_l \approx x_l^c$ for all l .

Computing the entire feasibility region of an operational wireless network is a formidable task. As mentioned earlier, previous approaches focus on modeling the detailed operation of 802.11 but cannot support online operation because they require measurements during network downtime. We consider a model that captures the boundary of the feasibility region with a set of parameters that can be estimated online. We model the feasibility region by a convex polytope. Each point $\mathbf{c} = (c_1, \dots, c_L)$ in the convex hull (boundary) of the polytope is determined as linear combination of K extreme points $\mathbf{c}^{[k]} = (c_{k1}, \dots, c_{kL})$ as follows:

$$\mathbf{c} = \sum_{k=1}^K \alpha_k \mathbf{c}^{[k]} \quad (1)$$

$$\sum_{k=1}^K \alpha_k = 1 \quad (2)$$

$$\alpha_k \geq 0, k = 1, \dots, K \quad (3)$$

The model estimates as feasible any set of output rates $\mathbf{y} = (y_1, \dots, y_L)$ that lies inside the polytope, including the convex hull.

To understand how this model captures feasibility and interference consider the two-link scenario of Figure 1, where the feasibility region is characterized by four extreme points. The two primary extreme points $\mathbf{c}^{[1]} = (c_{11}, 0)$ and $\mathbf{c}^{[2]} = (0, c_{22})$ correspond to the maximum output rates when each link transmits alone. If the two links do not interfere when they transmit simultaneously, the set of feasible rates is bounded by the rectangular “Independent region”. If the links interfere but are scheduled in a mutually exclusive manner, the feasible rate region is represented by the “Time sharing” region, where the sum of normalized output rates does not exceed unity ($y_1/c_1 + y_2/c_2 \leq 1$). The two secondary extreme points $\mathbf{c}^{[3]}$ and $\mathbf{c}^{[4]}$ capture partial interference when the two links transmit simultaneously.

The model is attractive mainly for two reasons. First, it is simple and parsimonious: K parameters (extreme points $\mathbf{c}^{[k]}$) are sufficient to characterize the feasibility region. Second, it consists of linear constraints. This allows network optimization using standard convex optimization techniques that also lend themselves to decentralized solutions.

¹Also termed *stability region* in the control-theoretic literature.

We note that TDMA or time-sharing CDMA networks can be characterized by convex feasibility region where the extreme points are readily provided by SINR formulas. However, it is not evident how such a model can be applied to the 802.11 CSMA MAC protocol. Applying this model to a real-world 802.11 network requires addressing two challenges: (i) defining extreme points for adequate characterization of the feasibility region (ii) computing these extreme points during network operation, in a non-intrusive manner.

3.2 Extreme Points Computation

Computing extreme points calls for applying input rates that yield output rates on the boundary of the feasibility region, and for performing such measurement repeatedly to accommodate the time-varying nature of wireless link quality and interference patterns.

Since *primary* extreme points correspond to output rates when links transmit alone at maximum input rate, they are defined as the maximum UDP throughput when a link transmits in isolation in backlogged mode. This simple definition belies significant computational challenges, since during network operation links hardly ever transmit alone or in backlogged mode. Section 5 introduces an online capacity estimation procedure to compute primary extreme points.

Secondary extreme points are challenging in both their definition and computation. One approach is to schedule all combinations of links transmitting and backlogged, and use the resulting output rates as secondary extreme points. Although this does not necessarily guarantee output rates at the boundary of the feasibility region, our experiments have shown that it works well in practice. On the other hand, it requires $O(2^L)$ measurements and a separate measurement phase during network downtime. Hence, it is not amenable to online computation.

Our approach to characterizing secondary extreme points is to compute them based on combining primary extreme points with an interference model. The interference model relies on two simplifying assumptions. First, it assumes pair-wise interference, where interference between two links is independent from the interference of other links. Second, it assumes binary interference, where two links are either mutually exclusive or do not interfere. For the two-dimensional configuration of Figure 1, this binary interference model maps to the Time Sharing (in the case of interference) and Independent Regions (in the case of no interference).

The identification of binary interference patterns across links is derived from a conflict graph, where each vertex corresponds to a unidirectional link and each edge identifies interference between the two links that correspond to its endpoint vertices. The independent sets of the conflict graph indicate sets of links that can transmit simultaneously without interference. The secondary extreme points are constructed based on the maximal independent sets of the conflict graph, with the latter computed using a fast maximal clique enumeration algorithm [22] run on the complement of the conflict graph. Each maximal independent set m is represented by a 0-1 $L \times 1$ vector $\mathbf{v}[m]$ where non-zero elements denote the links of this independent set. Each vector $\mathbf{v}[m]$ is mapped to a secondary extreme point $\mathbf{c}^{(2)}[m]$ by multiplying it by an $L \times L$ diagonal matrix $\mathbf{C}^{(1)}$, where each column corresponds to a primary extreme point $\mathbf{c}^{(1)}[k]$:

$$\mathbf{c}^{(2)}[m] = \mathbf{C}^{(1)} \mathbf{v}[m], m = 1, \dots, M \quad (4)$$

where M is the number of maximal independent sets in the conflict graph. Thus, Eq. (4) replaces the unit entries of $\mathbf{v}[m]$ with the capacities of the corresponding links.

3.3 Discussion

Our model is clearly an approximation and relies on assumptions that do not fully capture the real-world complexity of 802.11 mesh networks and could lead to inefficiencies. First, the feasibility region of 802.11 is known [5, 7, 17, 18] to not be necessarily convex. Further, the assumption of pair-wise and binary interference used to compute extreme points is clearly an approximation.

For example, [24] provides experimental evidences that 802.11 interference is not binary and [20, 25, 30] further validated this by providing accurate throughput predictions using non-binary interference models. Similarly, [18] demonstrates non-convexity for various topologies using models and simulations. Interestingly though, we have observed that in most cases the feasibility region in [18] can be adequately captured by a convex envelope.

In general, it is not obvious that the model’s convexity assumption will translate in poor accuracy when estimating the feasibility region, and we are not aware of analytical or experimental studies quantifying this inaccuracy. The pair-wise interference assumption was successfully used in recent 802.11 models [18, 21, 23]. The binary interference assumption has been shown in [20, 24, 25, 30] to affect accuracy, but the validity of this conclusion across different binary interference models is unclear.

4. MODEL VALIDATION

In this section, we experimentally validate our assumptions and the accuracy of the model’s throughput estimates. Since our focus is on assessing the model’s accuracy, we do not concern ourselves with feasibility considerations for on-line measurements. Specifically, we measure directly maxUDP throughput and rely on an approach for capturing pair-wise binary interference patterns that doesn’t lend itself to an on-line solution. Section 5 discusses how to overcome these limitations in an operational setting. Next, we introduce the testbed before proceeding with our experimental validation, first on two-link topologies and then on larger topologies with multiple links and multi-hop flows.

4.1 Wireless Mesh Testbed

Our 18-node mesh network testbed (Figure 2) spans a parking lot and multiple floors of three multi-story office buildings. It consists of both indoor and outdoor links, thus providing a rich variety of wireless conditions. Each node is equipped with an Atheros 802.11a/b/g AR5212 mini-PCI wireless interface connected to an external omni-directional 5 dBi antenna. Each wireless interface is controlled by the Madwifi driver (v. 0.9.4) running on Linux kernel v.2.6.18. All cards are configured in the ad-hoc demo mode. In all experiments we use 802.11g with RTS/CTS disabled and the transmit power is fixed to 19dBm for all nodes. We use iperf² to generate and measure traffic in the mesh network. We present validation results for 1 Mb/s and 11 Mb/s data rates, with modulation data rate adaptation disabled. The validation results with data rate adaptation enabled were similar. However, the maxUDP throughput estimation problem (addressed in Section 5) is much more challenging in this case and is a topic of further investigation. It should be noted that data rate adaptation has not been incorporated in any of the existing models for multi-hop 802.11 networks.

4.2 Binary LIR Interference Model

The Link Interference Ratio (LIR) is a metric that measures interference between link pairs in CSMA networks [24]. LIR is defined as follows:

$$LIR = \frac{c_{31} + c_{32}}{c_{11} + c_{22}} \quad (5)$$

²<http://sourceforge.net/projects/iperf>

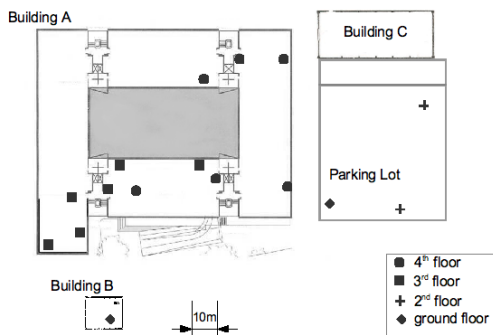


Figure 2: Wireless mesh network testbed.

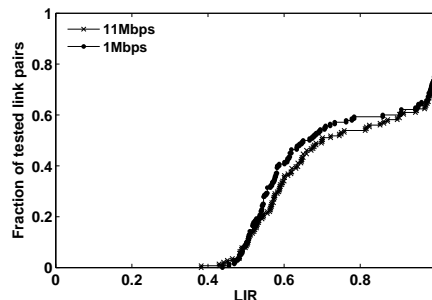


Figure 3: CDF of LIRs of several link pairs in the testbed at 1 Mb/s and 11 Mb/s data rates.

where c_{11}, c_{22} and c_{31}, c_{32} are UDP throughputs when the links are backlogged and transmit individually and simultaneously, respectively. $LIR = 1$ indicates no interferences, with lower LIR’s indicating a higher degree of interference.

Figure 3 depicts the CDF of LIRs at 1 Mb/s and 11 Mb/s data rates for 141 link pairs in our testbed. For both data rates, several LIR values are between 0.5 and 1, which indicates non-binary interference, but most of them are either below 0.7 or above 0.95

We therefore use a binary interference model based on an LIR threshold. Links with $LIR > 0.95$ are classified as “non-interfering” and their feasibility region is given by the “Independent region” defined by their primary extreme points $(c_{11}, 0)$ and $(0, c_{22})$ and the secondary extreme point (c_{11}, c_{22}) . Otherwise, links are classified as “interfering” and their feasibility region is the “Time sharing region” defined only by the primary extreme points $(c_{11}, 0)$ and $(0, c_{22})$.

The use of a high LIR threshold is expected to accurately capture non-interfering link pairs. As a result, we next focus on interfering link pairs.

4.3 Validating Link Pairs Characterization

We evaluate the ability of the maxUDP/binary LIR model to characterize the feasibility region in several configurations of interfering link pairs. This is an important validation step since pair-wise link interference has been successfully used as a building block for modeling larger topologies [18, 21, 23]. In Section 6 we use pairwise link interference to construct the feasibility region and optimize the performance of networks with multi-hop flows.

4.3.1 Methodology

Topology Classes. We categorize interfering links into three topology classes [16]. In a Carrier Sense (CS) topology, the two trans-

mitters can sense each other. In the Information Asymmetry (IA) topology the transmitters cannot sense each other but *one* receiver can sense the other link’s transmitter. In a Near Far (NF) topology, the transmitters cannot sense each other, but each receiver can sense the other link’s transmitter. These topology classes yield different LIRs and different bandwidth sharing properties.

Experiments. We compute the feasibility region of several interfering link pairs from the CS, IA or NF topology classes. For each link pair we consider various configurations where the links use the same data rates ((1,1) Mb/s and (11,11) Mb/s) or different data rates (1,11) Mb/s. Some configurations with bad channel conditions yield UDP packet losses, as MAC retransmissions fail to mask channel losses.

For each configuration, we perform ten back-to-back iterations, each consisting of two phases. In the first phase each link transmits alone in backlogged mode for 30s and its UDP throughput and UDP packet loss rate p_l are recorded. These maximum UDP throughputs form the primary extreme points $(c_{11}, 0)$ and $(0, c_{22})$ of the estimated feasibility region (Figure 1). In the second phase, we apply several pairs of input rates x_l (30s each) within the “Independent region” defined by these two points (see Figure 1). The output rates y_l are marked as *feasible* if they are within 2% of $(1 - p_l) \times x_l$ for both links.

In order to properly validate our model, we needed to fully characterize the feasibility region of each link pair. This in turn required a large number of experiments (input rates) that took time to perform. In order to minimize discrepancies caused by changes in the feasibility region over the duration of the experiments, we sought to minimize external variability factors. As a result all experiments took place during nights and weekends, and sniffers were used to limit operation to periods of low activity on the testbed. We also excluded iterations of any experiment where measured maximum UDP throughputs and UDP losses were 5% below the average.

Metrics. We evaluate the model’s accuracy on interfering link pairs using *false positives* (FPs) and *false negatives* (FNs). FPs measure overestimates, where output rates (y_1, y_2) are estimated feasible by the model but measured as infeasible during the experiment. FNs measure underestimates, where the output rates are estimated infeasible by the model but measured as feasible during the experiment. Ideally, both FPs and FNs should be kept to a minimum.

4.3.2 Results

Figure 4 summarizes the FPs and FNs for all two-link configurations identified as interfering, when maxUDP throughputs c_{11} , c_{22} are used as link capacities. Accuracy varies across configurations, but the approach produces only 94 FPs out of 3026 points tested across all configurations, thus ensuring conservative predictions and rates that remain inside the feasibility region. Things are more variable for FNs. There are very few FNs under CS configurations, which means that the approach accurately characterizes the feasibility region. This holds across data rate combinations and different channel conditions with and without UDP packet losses. The main reason is that mutual carrier sensing forces the links to operate close to the time sharing region.

The number of FNs is higher in IA and NF configurations. The source of inaccuracies is the capture effect present in those scenarios. Since the two links are not coordinated through carrier sensing, they transmit in parallel and transmissions overlap at the receivers. When capture occurs, the receivers can decode several packets despite the overlapping transmissions. Hence, both links transmit successfully and the actual feasibility region rises above the time sharing line. An extreme example of this inefficiency is shown in Figure 5 where the time sharing region predicted by the model missed approximately 40% of the feasibility region.

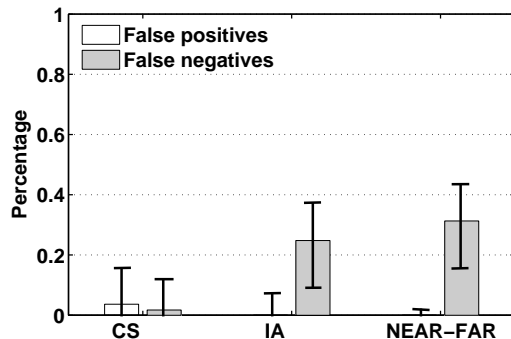


Figure 4: FP and FN mean errors (with max and min observed values) for all interfering two-link configurations.

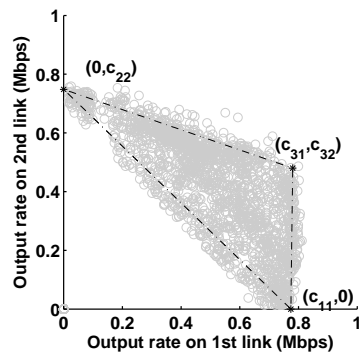


Figure 5: Extreme example of fraction of the feasibility region missed by the two primary extreme points model in an IA topology at 1 Mb/s data rate. Most of this region was recovered using a 3-point model with LIR point (c_{31}, c_{32}) as third secondary extreme point.

This inefficiency could be addressed by adding as a third secondary extreme point, the backlogged UDP throughputs (c_{31}, c_{32}) when the two links transmit simultaneously. Figure 5 shows that this third point recovered most of the lost fraction of the feasibility region. We applied this three-point model to all tested configurations and it eliminated almost all FNs while increasing by 3% the false positives in the IA and NF configurations in Figure 4. Despite its excellent accuracy, the three-point model is not amenable to online computations—it would require $O(L^2)$ measurements where links are scheduled to transmit alone and in pairs during a separate measurement phase. However, as we see next, it can serve as a reference to derive the error incurred because of the maxUDP/binary LIR interference assumption.

4.4 MaxUDP/Binary LIR Error Computation

The previous experiments show that a model which uses the maxUDP throughput of individual links as primary extreme points and a binary interference rule based on an LIR threshold of 0.95 to classify links as non-interfering, can provide a reasonable yet at times conservative estimate of the feasibility region. Given these findings, it is of interest to assess whether the choice of an LIR threshold of 0.95 is the most appropriate, and more generally the extent of the errors committed by the model in practical settings. We thus seek to develop a more principled understanding of the type and magnitude of errors associated with the model and dif-

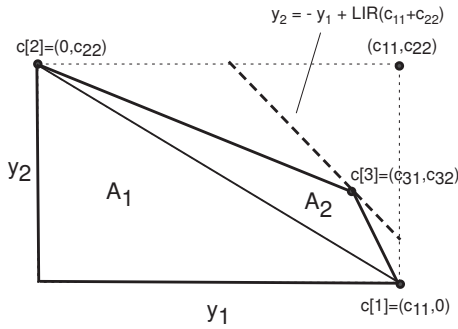


Figure 6: Computation of FP and FN errors of binary LIR model. Each LIR value corresponds to several points (c_{31}, c_{32}) forming the dotted line. A higher LIR value corresponds to a parallel line toward the (c_{11}, c_{22}) .

ferent choices of LIR thresholds. For that purpose, we rely on the additional finding that emerged from our experiments regarding the relative accuracy of a three-point convex model using the simultaneously backlogged throughputs (c_{31}, c_{32}) as secondary extreme point. Specifically, although such a three-point model is not practical when considering online measurements, it accurately captures the feasible region even in instances of high capture. As a result, we use this model as a starting point to develop a better understanding of the impact of the LIR threshold selection on our proposed binary interference model, and more generally estimate its expected error as a function of link characteristics.

Consider Figure 6 that depicts the feasibility region of a two link configuration. The two primary extreme points $c[1] = (c_{11}, 0)$ and $c[2] = (0, c_{22})$ delineate the time-sharing region (region A_1), as well as the boundaries of the independent region through the point (c_{11}, c_{22}) and the corresponding fine-dotted lines. The secondary extreme point $c[3] = (c_{31}, c_{32})$ shows the throughput that can be realized when both links are simultaneously backlogged, together with the additional feasibility region (region A_2) when using the three-point model. The binary model selects as its estimate of the feasibility region either the time-sharing region (region A_1) or the independent region (fine-dotted rectangle) based on the position relative to the LIR threshold of the measured LIR value associated with $c[3]$. In other words, if the LIR value associated with $c[3]$ is less than the LIR threshold, then the two links are assumed to interfere, and their feasibility region is taken to be the time-sharing region. In contrast, if the LIR value associated with $c[3]$ is greater than the LIR threshold, then the two links are considered as non-interfering and their feasibility region is taken to be the independent region. In the first case, there should be no FPs, but FNs will occur for all throughput combinations in A_2 , *i.e.*, the FN error is equal to $\frac{A_2}{A_1 + A_2}$. Conversely, in the second case there are no FNs, but FPs are recorded for all throughput combinations in the complement of $A_1 \oplus A_2$ to the rectangular independent region, *i.e.*, the FP error in this case is equal to $\frac{c_{11}c_{22} - (A_1 + A_2)}{A_1 + A_2}$.

From Figure 6, we see that a high LIR value corresponds to shifting the secondary extreme point $c[3]$ “upwards” to the corner point (c_{11}, c_{22}) of the independent region. This is why a large LIR threshold ensures a small region for FPs, but a correspondingly large FN area when the LIR falls just below the threshold. Figure 6 also shows that when $c[1] \neq c[2]$, the LIR value alone does not fully specify the magnitude of the possible error that depend on the actual combination of throughputs, c_{31} and c_{32} , observed in realizing the LIR value (all points on the thick dotted line of Figure 6

yield the same LIR value but different areas³ A_2). Nevertheless, the representation of Figure 6 allows us to compute the magnitude of the overall (FN and FP) error when using the proposed binary LIR model with a given threshold and for an observed distribution of link LIRs. Specifically, using the LIR distribution of Figure 3, the expected FP error was 2% and the expected FN error for $LIR_{th} = 0.95$ was only 13.3%. This is much lower than the FN error of 40% reported for the “extreme” example of Figure 5. We evaluated other LIR thresholds and found that a value of 0.95 offers a reasonable compromise between FN and FP errors for this particular LIR distribution. More generally, the methodology captured in Figure 6 provides a systematic approach to both select an appropriate LIR threshold and estimate the error of the binary LIR model given an LIR distribution in the mesh network.

4.5 Network Validation

We now validate the model’s accuracy in more complex configurations in our mesh network testbed. In this case, it is impossible to sample the entire feasibility region as we did for the case of link pairs. Instead, we use an optimization framework (detailed in Section 6) to generate test input rate vectors that hit the boundary of the feasibility region. More specifically, the test input rates are computed based on the proportionally fair⁴ output rates of the feasibility region estimated by our model.

Methodology. We use ten network configurations that include different combinations of data rates at 1 Mb/s and 11 Mb/s. Each configuration has up to six simultaneous interfering multi-hop flows and a maximum route length of four hops. Routes are initialized using the ETT metric [24] and remain fixed for the duration of each experiment.

Each experiment runs for ten iterations, with each iteration consisting of two phases. The first phase estimates feasibility region. It first measures the maxUDP throughputs c_{li} , UDP packet losses p_l and the LIRs for all the links used in the configuration. Then, based on the c_{li} values of primary extreme points and a binary LIR interference model with threshold 0.95, it constructs the conflict graph and secondary extreme points using the procedure in Section 3.2.

The second phase computes and injects the test input rates. First, it computes the proportional fair output rates y_s . Then, it determines the test input rates as $x_s = y_s / (1 - p_s)$. The path loss p_s is estimated based on the measured link loss rates p_l in the path s as $1 - \prod_{l \in s} (1 - p_l)$. Then, the second phase tests both the estimated input rate vectors x_s and several scaled-up versions. Scaling factors of 1.1, 1.2, and 1.5 are used to test for possible instances of underestimates. This scaling searches for the feasibility region boundary in the direction of the proportional fair rate vectors x_s .

Results. The model over-estimates if any of the achieved throughputs when applying the test input rates x_s , is less than the corresponding estimated output rate. In other words, the estimated output rates y_s are not feasible. The model under-estimates if the output rates y_s were feasible but could be scaled-up by some factor without violating feasibility.

Over-estimation results are summarized in Figure 7, which shows a scatter plot of the estimated output rates y_s versus achieved throughputs using the data from all tested configurations. Most points are

³When $c[1] = c[2]$, it is easy to show that all possible realizations of the LIR yield the same FN and FP error areas.

⁴The maximum aggregate throughput objective would tend to create test points close to the primary extreme points. On the other hand, max-min fair points would tend to lie in the interior of the true feasibility region. We selected the proportional fairness objective that provides a trade-off between maximum utilization and fairness.

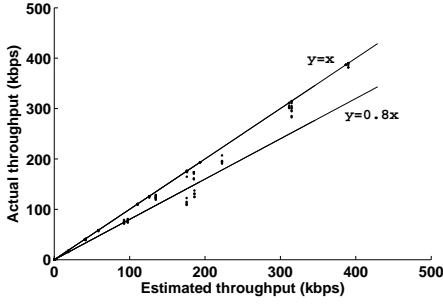


Figure 7: Model over-estimation performance. Points on the ($y=x$) line indicate output rates correctly estimated as feasible.

concentrated on the upper line ($y = x$), which indicates perfect estimation. The maximum error was 38% and only 10 points fall below the lower line ($y = 0.8x$), which indicates the 20% error region.

Under-estimation results are summarized in Figure 8. Figure 8(a) presents the CDF of the ratio of achieved throughput over estimated output rate for all scaling factors. Feasible points are on the right-hand side of the figure and close to unity. As the scaling factor increases, the estimation error also increases, as indicated by the progressive shift to the left of the scaled CDFs. This provides evidence that the model is not producing significant under-estimates, i.e., in most instances the scaled estimated output rates are not feasible.

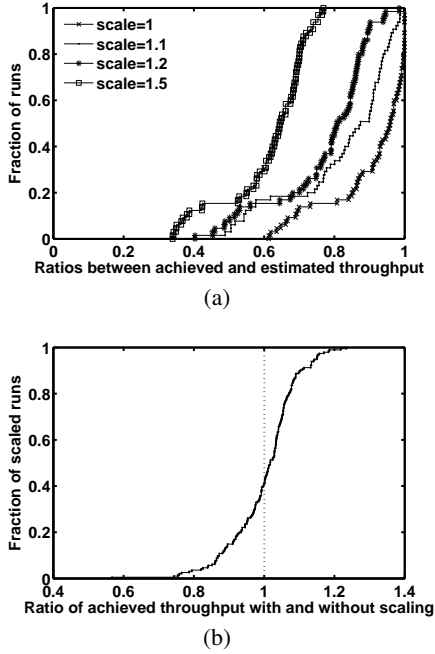


Figure 8: Model under-estimation performance. (a) CDF of ratio of achieved throughput over estimated output rate for various scaling factors. (b) CDF of ratio of scaled over non-scaled achieved throughput.

To better quantify the magnitude of the gap that exists between the capacity predicted by the model and the maximum feasible value, we compare achieved throughput with and without scaling. If by scaling the input rate the achieved throughput increases, then

the estimate produced by the model left some capacity unused. Figure 8(b) plots the CDF of the ratio of highest achieved throughput when scaling over the achieved throughput without scaling. The model under-estimates capacity by about 20% in the worst case and around 10% on average. This seems a reasonable price to pay for the model's simplicity, which, as we show next, is instrumental in developing a practical on-line solution.

5. ONLINE APPROXIMATIONS

The previous section showed that maxUDP throughput combined with a binary LIR interference model yields an adequate approximation of the feasibility region in practice. In this section, we show how this approximation can be realized using online measurements. Sections 5.1 through 5.4 present and evaluate a light-weight technique for online estimation of maxUDP throughput. Next, Section 5.5 presents and evaluates a two-hop interference model that approximates the binary LIR interference model.

5.1 Online Capacity Estimation

MaxUDP throughput captures the inherent quality of each link when transmitting alone in backlogged mode. However, in an operational network direct measurement of this quantity are not feasible because links transmit neither alone nor backlogged. An additional challenge is to estimate this quantity using measurements at the network layer without access or modifications to lower layers of the protocol stack.

Our solution has two components. First, we use a capacity representation that relates maxUDP throughput to the packet loss rate experienced by the MAC protocol. The packet loss rate is measured online using network-layer broadcast probes which incur low overhead. However, when interference is present this packet loss rate includes losses due to both channel errors and collisions. Hence, we design a channel loss rate estimator which filters out collisions and recovers the channel loss rate used in the capacity representation formula. We evaluate the estimation accuracy of our approach and compare it to AdHoc Probe [10], a tool that has been proposed to estimate path capacity in multi-hop wireless networks.

5.2 Link capacity representation

We express the maxUDP throughput T of each link as a function of the channel loss rate p_l , as follows:

$$T = \frac{P}{t_{idle} + t_{tx}} \quad (6)$$

where P is the UDP payload size and t_{idle} and t_{tx} are the average idle and transmission times, respectively, approximated as follows:

$$t_{tx} = \frac{P + H}{(1 - p_l^{ETX})T_{nom}}$$

where H is the UDP header size and T_{nom} the nominal throughput. ETX equals $1/(1 - p_l)$ and is the expected number of MAC retransmissions assuming independent losses with probability p_l .

$$t_{idle} = \begin{cases} F(1, [ETX] - 1), & \text{if } ETX < m \\ (F(1, m - 1) + \sigma \frac{([ETX] - m)(W_m - 1)}{2}), & \text{otherwise} \end{cases}$$

where σ is the 802.11 slot duration, W_0 and W_m , the minimum and maximum contention window size, respectively, m is the backoff stage where the contention window size becomes maximum, and $F(a, b) = \sigma \sum_{i=a}^b \frac{2^i W_0 - 1}{2}$ is the total average backoff time between backoff stages a and b .

All quantities in Eq. (6) are either known in advance or depend on p_l . T_{nom} can be computed given 802.11 MAC parameters, data

packet size and data rate based on [19]. W_0 and W_m and m are given by the 802.11 specification. The header H and the packet payload P are also known, and ETX depends on p_l .

The packet loss rate p_l is measured by a probing system that uses network layer broadcast packets. Broadcast packets are not subject to MAC retransmissions and reflect the packet loss rate experienced by the MAC protocol. The packet loss rate p_l is computed as $1 - (1 - p_{DATA})(1 - p_{ACK})$. p_{DATA} and p_{ACK} are DATA and ACK packet loss rates, respectively. These rates are measured as fraction of lost DATA and ACK broadcast probes over a probing window. The broadcast probes emulating DATA and ACK are sent at the data rate and packet size of the DATA and ACK packets, respectively.

During network operation, packet losses are due to both channel errors and collisions and the measured loss rates p_{DATA} and p_{ACK} will be higher than if the links transmitted alone. In order to use the maxUDP throughput representation, we must be able to distinguish collisions from channel errors and use Eq. (6) with a p_l computed from the estimated channel loss rates of p_{DATA} and p_{ACK} . Therefore, the problem of online maxUDP throughput estimation translates to the problem of separating channel losses from collisions. Previous solutions to this problem [29, 35] have been designed for client-AP WLAN traffic scenarios or require low-level access to firmware. Next, we present a technique that applies to multi-hop wireless mesh networks and relies only on the loss pattern of the network layer broadcast probes.

5.3 Channel Loss Rate Estimator

Our technique is based on three previous experimental observations for 802.11-based mesh networks in both city-wide [1] and long-distance rural deployments [8]: (i) collision losses are independent from channel losses, hence packet loss increases due to collisions (ii) collisions create bursty loss patterns, and (iii) for the majority of links, losses occur independently when collisions and interference are not present. Extensive experiments on our testbed (not shown due to lack of space) have confirmed these observations.

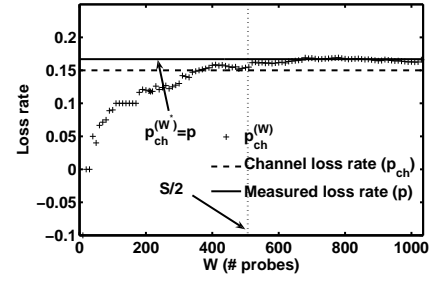
During network operation, the estimator runs continuously at the receiver of each link and is based on the loss patterns observed during each probing window S . The receiver computes the packet loss rate p by dividing the number of received probe packets by the total probe packets S sent during the probing window. In general p includes both collisions and channel errors. The problem is to recover the channel loss rate p_{ch} (i.e., the fraction of lost packets due to channel losses), from p .

The main idea is to scan the probing window with smaller sliding windows, each of size W , to identify segments that contain only channel losses. The sliding windows scan the probing window with steps of one probe. This provides $S - W + 1$ starting positions within the probing window. For the i -th step, the packet loss rate is computed as $n_i^{(W)}/W$, where $n_i^{(W)}$ is the number of lost packets (“errors”) in the sliding window. For each window size W , the channel loss rate $p_{ch}^{(W)}$ is then set to the minimum of these packet loss rate estimates:

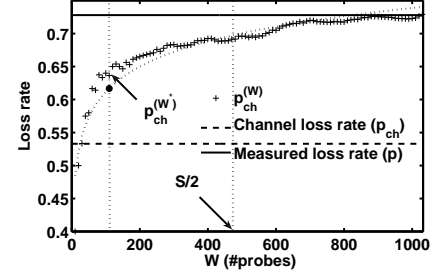
$$p_{ch}^{(W)} = \frac{1}{W} \min(n_1^{(W)}, \dots, n_{(S-W+1)}^{(W)}), W = W_{min}, \dots, S \quad (7)$$

The minimum window parameter W_{min} corresponds to the coarsest estimation of loss rate and we set it to 10 samples.

Which W provides the best estimate of the channel loss rate p_{ch} ? A very small W may capture very few losses and therefore underestimate the channel error rate; a very large W may capture both channel and collision losses and therefore over-estimate. For small W , the estimate $p_{ch}^{(W)}$ is smaller than p because some windows see too few losses (are not long enough to accurately “average out” channel losses). Then, $p_{ch}^{(W)}$ increases with W until it reaches the



(a) Measured loss rate p reached before $S/2$.



(b) Measured loss rate p reached after $S/2$.

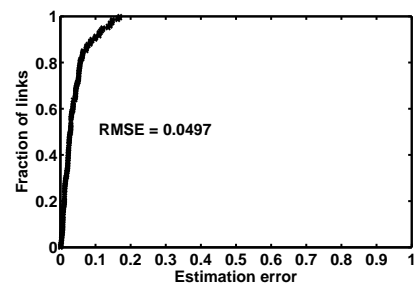
Figure 9: Examples of the two cases of the channel loss rate estimator. The measured loss rate p (horizontal line) and true channel loss rate (dotted horizontal line) were measured in two sequential experiments with and without interference, respectively.

measured packet loss rate p for $W = S$ ($p_{ch}^{(S)} = p$). Based on this observation, we design our filter around two cases.

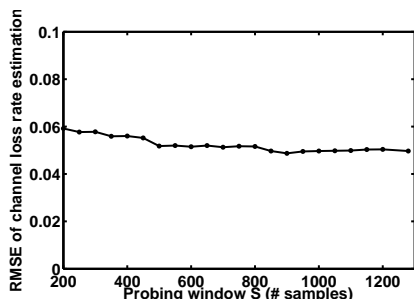
Case 1: If $p_{ch}^{(W)}$ increases steeply and reaches p fast, this is a strong indication that p_{ch} is close to p . In addition, reaching and staying at a loss rate fast is an indication of uniform packet losses. The reason is that a small W already captures all losses observed in the probing window. We use a median criterion to identify this case. If $p_{ch}^{(W)}$ exceeds $0.99p$ before $W = S/2$ then p_{ch} is estimated as p . Figure 9(a) shows an example of this case for a link in our testbed.

Case 2: When the median criterion does not hold, p_{ch} is lower than p . We approximate the sequence $p_{ch}^{(W)}$ by a logarithmic curve of the form $f(w) = a \ln(w) + b$ for w in $[W_{min}, S]$. We then find the point w^* , where $f(w)$ has maximum curvature. This is the point after which the curve no longer continues to rapidly rise and instead follows either a steady state or slow rise. Then, the optimal window size is selected as $W^* = \lfloor w^* \rfloor$ and the channel loss rate is estimated as $p_{ch} = p_{ch}^{(W^*)}$. Figure 9(b) shows an example of a link in our testbed, where the logarithmic fit estimation is applied.

Evaluation of channel loss rate estimator. We evaluate the channel estimator’s accuracy and the time scale over which the accuracy is maintained. We run experiments in our testbed, each consisting of two phases. In the first phase the nodes broadcast probes alone in backlogged mode and measure channel loss rate p_{ch} (ground truth). In the second phase they broadcast probes simultaneously and measure packet loss rate (p) subject to interference. The estimator is also applied during the second phase. We used a probing period 0.5s and a probing window $S = 1280$ probes. All results have been aggregated over several traffic scenarios with 40 links at 1Mb/s and 11Mb/s, resulting in over 1000 measurements.



(a) Error CDF for $S=1280$ probes.



(b) RMSE vs. probe window size S .

Figure 10: Channel loss rate estimation accuracy across 40 links at 1 Mb/s and 11 Mb/s and probing period of 0.5 s.

Figure 10(a) presents the CDF for the estimation error of the channel loss rate for a probing window of $S = 1280$ probes (640s). The technique achieves high estimation accuracy of keeping the error below 5% for 70% of all runs and providing a Root Mean Square Error (RMSE) of 0.0497. Figure 10(b) shows that the RMSE is slightly increased to 6% as the probing window size decreases until $S = 200$ probes (100s). This demonstrates the robustness of this technique for different probing window sizes and its ability to estimate capacity at a time scale of a few minutes.

5.4 Capacity estimation performance

We evaluate the accuracy of the capacity estimation technique and compare it to Ad Hoc Probe, a tool proposed in [10] to estimate capacity of multi-hop wireless paths. Ad Hoc Probe sends several packet pairs and estimates capacity as the inverse of the minimum dispersion (spacing) between the packets of each packet pair.

We run two-phase experiments on 30 links at 1 Mb/s and 11 Mb/s in our testbed. Each experiment is run on a single link and consists of two phases. During the first phase, we set the link to transmit backlogged and measure the maxUDP throughput. During the second phase, we activate the probing system and send simultaneously 200 broadcast probes of our capacity estimator (probing period 0.5 s) and 200 unicast packet-pairs of Ad Hoc Probe in the presence of interfering background traffic. Figure 11 shows that AdHoc probe consistently fails to predict maxUDP throughput, resulting in extremely high estimation error in some cases. On the other hand, our capacity estimator yields low error (RMSE=12%) and is independent of background traffic.

Ad Hoc Probe is not appropriate for link maxUDP throughput estimation for the following reasons. First, in absence of interference it estimates a higher value closer to the nominal throughput. This is because Ad Hoc Probe estimations are based on minimum dispersion (delay) estimates which don't take into account the inherent link channel losses. Second, in the presence of interferences

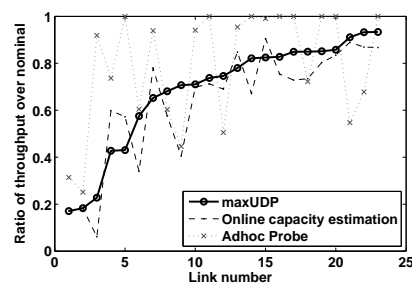
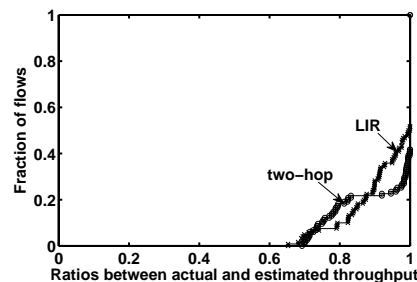


Figure 11: MaxUDP throughput and estimated capacity by our technique and Ad Hoc Probe, all normalized over nominal throughput for 23 links transmitting at 1 Mb/s or 11 Mb/s in our testbed.

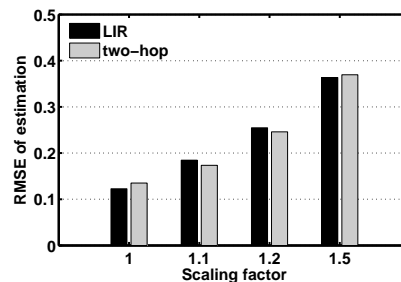
the minimum delay filter will filter out congestion but again will not account for losses.

5.5 Two-hop Interference Model

We use a two-hop pair-wise binary interference model, where each link interferes with all links adjacent to its node endpoints and all the links adjacent to their one-hop neighbors. This interference model was also used in the congestion control protocol of [28] and can be easily used during network operation. We evaluate the two-hop approximation using as reference the binary LIR interference model which was shown to perform well in Section 4. Figure 12(a) presents the CDF of prediction error in the exper-



(a) Over-estimation evaluation



(b) Under-estimation evaluation

Figure 12: Binary LIR vs. two-hop interference model. As in Section 4.5, we evaluate for (a) over-estimation by looking at the estimation feasibility and (b) underestimation by looking at the RMSE of estimation for different scaling factors.

iments of Section 4.5 using binary LIR and two-hop interference models and measured maxUDP throughput as link capacities. The two-hop model yields low error, and hence provides a realizable

solution. Fig. 12(b) compares the RMSE of the two interference models for scaled input rates. The RMSE is increased for both models when scaling the input rates, showing that they are both near-optimal in terms of total network capacity. We conclude that the two-hop model provides an excellent approximation of the binary LIR model.

6. ONLINE OPTIMIZATION

In this section, we implement an optimization framework that incorporates the model, and we evaluate its online performance with TCP traffic in our testbed.

6.1 Implementation

Our decentralized implementation of the optimization framework uses *Click* [12] and consists of the following modules.

Routing module. Routes, neighborhood information and channel loss rate estimates are disseminated using the *Srccr* routing protocol that comes with *Click*. This information is used by each node to update its *Srccr* topology database, compute routes using Dijkstra's routing algorithm with ETT [13] as link metric, and execute the optimization algorithm in the optimizer module. Our only modification to *Srccr* was the addition of channel loss rate estimates to the route updates.

Capacity estimation module. Capacity estimation is executed continuously at each node and is implemented using the *Click* probing system, which sends periodically broadcast probes at data rate and size equal to the ones of DATA packets and broadcast probes equal to the ACK packet size at 1 Mb/s data rate, to measure loss rates of DATA packets and ACK packets, respectively. As detailed in Section 5, for each link of a node, the capacity estimation module estimates the corresponding DATA and ACK channel loss rates, combines them into an estimated channel loss rate, which is then used in Eq. (6) to compute the link capacity.

Optimizer module. This module takes as input the channel loss rates, neighborhood relationships, and the routing matrix R coming from the routing module, and outputs a set of target output rates $\mathbf{y} = (y_1, \dots, y_S)$ for the network. First, it computes the extreme points $\mathbf{c}[k] = (c_{k1}, \dots, c_{kL})$ using the procedure of Section 3.2. Then, it uses the routing matrix and extreme points in the following convex optimization problem:

$$\begin{aligned} & \text{Maximize} && \sum_{s=1}^S U(y_s) \\ & \text{subject to:} && \sum_{s=1}^S R_{l,s} y_s \leq \sum_{k=1}^K \alpha_k c_{kl}, \quad l = 1, \dots, L \\ & && \sum_{k=1}^K \alpha_k = 1 \quad \alpha_k \geq 0, k = 1, \dots, K \end{aligned}$$

where S is the number of flows, $R_{l,s}$ are binary routing variables indicating whether flow s is routed through link l . In the above formulation, the network routing matrix translates the multi-hop flow rates to link rates that lie in the feasibility region, created by the primary extreme points and the pair-wise interference model based on the procedure in subsection 3.2.

The utility function $U(\cdot)$ is given by:

$$U(y_s) = \begin{cases} \frac{y_s^{1-\alpha}}{1-\alpha}, & \text{if } \alpha \neq 1 \\ \log(y_s), & \text{otherwise} \end{cases}$$

which is a well-known family that provides a wide range of objectives that trade-off fairness and throughput. Given the network output rate vector $\mathbf{y} = (y_1, \dots, y_S)$, the module selects the subset of rates y_s for which it is a source and generates the correspond-

ing input rates, as $x_s = y_s / (1 - p_s)$. The path loss p_s is estimated based on the channel loss rates p_l in the path s as $1 - \prod_{l \in s} (1 - p_l)$.

Rate control module. This module uses *Click BandwidthShaper* element to rate limit the flows according to the optimized rates x_s .

Rate Control Time Scale. The time scale at which rate control is adjusted depends on the capacity estimation interval (determined by the probing window size and period), and the optimizer module computation time.

The capacity estimation interval is constrained by the network layer probing system. Probing frequency should be low enough (every 0.5 s in our system) to keep the overhead low, but enough probes should be used to ensure sufficient accuracy. As illustrated in Figure 10(b), stable channel loss rates and hence capacity estimates can be obtained using a probing window size that corresponds to few minutes (100s to 640s). In all the experiments, a probing period of 0.5s and a probing window size of $S=200$ probes were used.

The optimizer module computation time consists of the time to compute the extreme points of the feasibility region, and the time to solve the optimization problem. The extreme points computation uses the maximal clique enumeration algorithm of [22]. According to [22], the algorithm can in practice compute about 100,000 maximal cliques per second if the graph is sparse, with linear increase in computation time as density increases. The worst-case scenario in our testbed was a conflict graph which resulted in 200 extreme points, computed in less than 10 ms. The convex optimization problem was solved with Matlab, which in this scenario terminated in less than 3 s.

In our system, the bottleneck is therefore the capacity estimation module.

6.2 Evaluation

The previous sections have shown that the optimization framework should enable precise (rate) control of UDP traffic to realize specific throughput objectives. We now turn to potentially more realistic scenarios involving TCP traffic, where interactions between TCP and rate control may affect the overall outcome. We also scale down the computed optimal rate of each TCP flow by a small factor to provide air time for TCP ACKs in the reverse direction. As in [21], this factor is $(1 - \frac{A+H}{A+H+D})$, where A , H and D are IP/TCP header, TCP ACK and TCP payload sizes, respectively.

We evaluate the performance of the optimization framework in terms of aggregate throughput, fairness (captured by Jain's Fairness Index (JFI)), and ability to isolate TCP flows. Flow isolation is quantified by a feasibility metric and a stability metric. The feasibility metric is the ratio of achieved TCP throughput over the optimized rate limit. The stability metric measures variations in the TCP throughput measurements of each flow across different experiments in the same configuration. It is defined as the ratio of the difference between throughput measurement and their mean over this mean. If all TCP flows achieve the optimized rates and show little variation, then they have been perfectly isolated.

All scenarios are run in two phases. In the first phase, rate control is disabled, while enabling the measurements in the probing system and dissemination of packet losses. In the second phase, the routes are fixed and rate control is enabled. We compare TCP throughput with rate control disabled (TCP-noRC) and enabled (TCP-RC). We use TCP-RC with two objectives: Maximum aggregate throughput (TCP-Max) and proportional fairness (TCP-Prop).

6.3 Results

We first study a simple TCP starving scenario in mesh networks, which involves one 2-hop and one 1-hop TCP flow transmitting

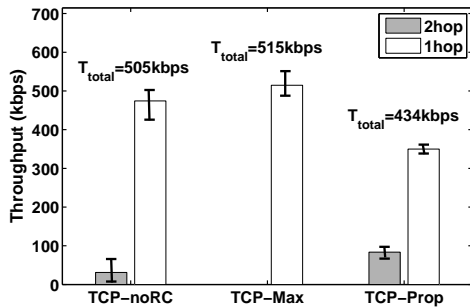


Figure 13: Performance of two-flow upstream TCP starvation scenario in our testbed with and without rate control at 1 Mb/s data rate. Bars plots are mean values. Error bars are max and min values.

upstream over a 2-hop path to a gateway node [33]. Figure 13 summarizes the results for such a scenario running with link speeds of 1 Mb/s in our testbed. TCP-noRC achieves similar distribution as TCP-Max, hence achieving close to maximum aggregate throughput. However, it starves the 2-hop TCP flow, because TCP ACKs (coming in the reverse direction) are lost due to collisions with TCP data packets [33]. TCP-Prop addresses starvation by increasing the throughput of the starving flow at some expense in aggregate throughput. We explore this further in Figure 14. We also observe that rate control helps isolate the two flows. In both TCP-Max and TCP-Prop, TCP flows achieve low variability.

Figure 14 presents results for several multi-hop, multi-flow configurations with link rates of both 1 Mb/s and 11 Mb/s in our testbed.

Figure 14(a) shows the CDF of the ratio of aggregate TCP with rate control over aggregate TCP-noRC for all tested scenarios. TCP-Max achieves up to 45% more aggregate throughput, indicating that it outperforms TCP-noRC when throughput maximization is the objective. TCP-Prop achieves over 80% of TCP-noRC aggregate throughput in 80% of the scenarios, indicating that the aggregate throughput penalty is not high. Also according to Figure 14(b), TCP-Prop improves fairness over TCP-noRC.

Figure 14(c) shows that 70% of the TCP flows achieve above 90% of their optimized rate limits and are considered feasible for practical purposes. The remaining 30% achieve a lower fraction, which for a few even drops down to 35% of their optimized rate target. This problem is not due to TCP ACK and TCP data collisions because the optimized rates already provide air time to the ACKs. Upon closer inspection of the data, we found that the reason is excessive losses (higher than 80%) on intermediate links of each path that cannot be masked by the 802.11 MAC protocol retransmissions. Rate control cannot help TCP in this case.

Figure 14(d) shows good stability when RC is applied: for 70% of the flows, the throughputs across different experiments on the same scenario deviate by less than 10% from the mean. These measurements correspond to the feasible flows of Figure 14(c). Under noRC, only 40% of the links achieve such stability.

7. CONCLUSIONS AND FUTURE WORK

We have addressed a pressing performance problem in 802.11 mesh networks using optimization-based rate control at the network layer. At the heart of our technique is a model that (i) can adequately characterize the 802.11 feasibility region in real-world mesh networks (ii) can be used in convex optimizations that support a wide range of objectives (iii) can have its parameters estimated

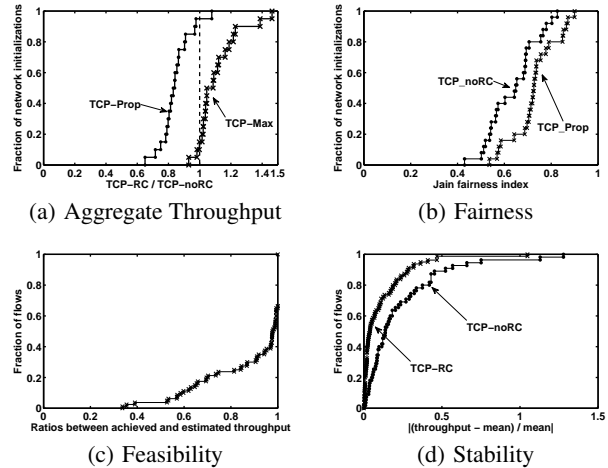


Figure 14: TCP performance with and without rate control across several multi-hop/multi-flow scenarios at 1 Mb/s and 11 Mb/s data rates in our testbed.

online using light-weight measurements at the network layer. We provided a structured validation of this model and incorporated it in an optimization framework in a wireless mesh testbed, demonstrating that such an approach can be implemented and operate at a time scale of a few minutes, during network operation. We showed that this framework can achieve predictable throughput, it can provide a wide range of optimization objectives that trade off throughput and fairness and can isolate TCP flows increasing their stability. On the other hand, rate control cannot help TCP flows when links in the path have high channel error losses. This problem can be addressed either by removing such links using the routing protocol or by increasing the MAC retransmission limit.

There are several directions in which we plan to extend this work, above and beyond a more comprehensive investigation of how the approach performs when 802.11 rate adaptation is turned on. Of particular interest is to incorporate routing as part of the optimization problem. This clearly increases complexity but may afford significant benefits in both increasing overall throughput and avoiding links with high channel error rates that decrease TCP performance. Additionally, while our capacity and loss rate estimators appear robust, we need to test them over a broader range of topologies and environments. Exploring filtering techniques used to detect sharp state transitions, e.g., as median filters for image edge detection, may prove useful, including in dealing with 802.11 data rate adaptation mechanisms.

8. ACKNOWLEDGMENTS

This work was supported by the OPNEX project of the European Community Seventh Framework Programme (FP7-ICT-224218).

9. REFERENCES

- [1] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level Measurements from an 802.11b Mesh Network. In *Proc. ACM SIGCOMM*, Portland, OR, Aug. 2004.
- [2] N. Ahmed, U. Ismail, S. Keshav, and D. Papagiannaki. Online Estimation of RF Interference. In *Proc. ACM CoNEXT*, Madrid, Spain, Dec. 2008.

- [3] U. Akyol, M. Andrews, P. Gupta, J. Hobby, I. Saniee, and A. Stolyar. Joint scheduling and congestion control in mobile ad-hoc networks. In *Proc. IEEE INFOCOM*, Phoenix, AZ, USA, Apr. 2008.
- [4] G. Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, 18(3):535–547, Mar. 2000.
- [5] T. Bonald and A. Proutiere. Flow-level stability of utility-based allocations for non-convex rate regions. In *Proc. IEEE CISS*, Princeton, NJ, USA, Mar. 2006.
- [6] R. Boorstyn, A. Kershenbaum, B. Maglaris, and V. Sahin. Throughput Analysis in Multihop CSMA Packet Radio Networks. *IEEE Transactions on Communications*, 35(3):267–274, Mar. 1987.
- [7] H. Chang, V. Misra, and D. Rubenstein. A General Model and Analysis of Physical Layer Capture in 802.11 Networks. In *Proc. IEEE INFOCOM*, Barcelona, Spain, Apr. 2006.
- [8] K. Chebrolu, B. Raman, and S. Sen. Long-Distance 802.11b Links: Performance Measurements and Experience. In *Proc. ACM MobiCom*, Los Angeles, CA, USA, Sep. 2006.
- [9] L. Chen, S. Low, and J. Doyle. Joint Congestion Control and Media Access Control Design for Ad Hoc Wireless Networks. In *Proc. IEEE INFOCOM*, Miami, FL, USA, Mar. 2005.
- [10] L.-J. Chen, T. Sun, G. Yang, M.Y. Sanadidi, and M. Gerla. Ad hoc probe: path capacity probing in wireless ad hoc networks. In *Proc. WICON*, Budapest, Hungary, Jul. 2005.
- [11] M. Chiang. Balancing Transport and Physical Layers in Wireless Multihop Networks: Jointly Optimal Congestion Control and Power Control. *IEEE Journal on Selected Areas in Communications*, 23(1):104–116, Jan. 2005.
- [12] Click. <http://read.cs.ucla.edu/click/>.
- [13] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Proc. ACM MobiCom*, Philadelphia, PA, USA, Sep. 2004.
- [14] Y. Gao, J. Lui, and D. Chiu. Determining the End-to-end Throughput Capacity in Multi-Hop Networks: Methodology and Applications. In *Proc. ACM SIGMETRICS*, Saint-Malo, France, Jun. 2006.
- [15] M. Garetto, T. Salonidis, and E. Knightly. Modeling per-flow throughput and capturing starvation in CSMA multi-hop wireless networks. In *Proc. IEEE INFOCOM*, Barcelona, Spain, Apr. 2006.
- [16] M. Garetto, J. Shi, and E. Knightly. Modeling media access in embedded two-flow topologies of multi-hop wireless networks. In *Proc. ACM MobiCom*, Cologne, Germany, Sep. 2005.
- [17] N. Hedge and A. Proutiere. Packet and flow level performance of multi-hop wireless networks. In *Proc. IEEE GLOBECOM*, San Francisco, CA, USA, Nov. 2006.
- [18] A. Jindal and K. Psounis. Characterizing the Achievable Rate Region of Wireless Multi-hop Networks with 802.11 Scheduling. *IEEE Transactions on Networking*, 16(1):63–76, Aug. 2009.
- [19] J. Jun, P. Peddabachagari, and M. Sichitiu. Theoretical Maximum Throughput of IEEE 802.11 and its Applications. In *In Proc. International Symposium on Network Computing and Applications (NCA)*, Cambridge, MA, USA, Apr. 2003.
- [20] A. Kashyap, S. Ganguly, and S. Das. A Measurement-based Approach to Modeling Link Capacity in 802.11-based Wireless networks. In *Proc. ACM MobiCom*, Montreal, Canada, Oct. 2007.
- [21] Y. Li, L. Qiu, Y. Zhang, R. Mahajan, , and E. Rozner. Predictable Performance Optimization for Wireless Networks. In *Proc. ACM SIGCOMM*, Seattle, WA, USA, Aug. 2008.
- [22] K. Makino and T. Uno. New algorithms for enumerating all maximal cliques. In *In Proc. 9th Scandinavian Workshop on Algorithm Theory*, Humlebaek, Denmark, Jul. 2004.
- [23] D. Nicolescu. Interference Map for 802.11 Networks. In *Proc. ACM Internet Measurement Conference (IMC)*, San Diego, CA, USA, Oct. 2007.
- [24] J. Padhye, S. Agarwal, V. Padmanabhan, L. Qiu, A. Rao, and B. Zill. Estimation of Link Interference in Static Multi-hop Wireless Networks. In *Proc. ACM Internet Measurement Conference (IMC)*, Berkeley, CA, USA, Oct. 2005.
- [25] L. Qiu, Y. Zhang, F. Wang, M. Han, and R. Mahajan. A general model of wireless interference. In *Proc. ACM MobiCom*, Montreal, Canada, Oct. 2007.
- [26] B. Radunovic, C. Gkantsidis, D. Gunawardena, and P. Key. Horizon: Balancing TCP over Multiple Paths in Wireless Mesh Network. In *Proc. ACM MobiCom*, San Francisco, CA, USA, Sep. 2008.
- [27] B. Raman and K. Chebrolu. Design and Evaluation of a New MAC Protocol for Long-Distance 802.11 Mesh Networks. In *Proc. ACM MobiCom*, Cologne, Germany, Aug. 2005.
- [28] S. Rangwala, A. Jindal, K. Jang, K. Psounis, and R. Govindan. Understanding congestion control in multi-hop wireless mesh networks. In *Proc. ACM MobiCom*, San Francisco, CA, USA, Sep. 2008.
- [29] S. Rayanchu, A. Mishra, D. Agrawal, S. Saha, and S. Banerjee. Diagnosing Wireless Packet Losses in 802.11: Separating Collision from Weak Signal. In *Proc. IEEE INFOCOM*, Phoenix, AZ, USA, Apr. 2008.
- [30] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, , and J. Zahorjan. Measurement-based models of delivery and interference. In *Proc. ACM SIGCOMM*, Pisa, Italy, Sep. 2006.
- [31] B. Scheuermann, C. Lochert, and M. Mauve. Implicit Hop-by-hop Congestion Control in Wireless Multihop Networks. *Ad Hoc Networks (Elsevier)*, 6(2):260–286, Apr. 2008.
- [32] A. Sharma and E. Belding. Freemac: Framework for Multi-channel MAC Development on 802.11 Hardware. In *ACM SIGCOMM PRESTO*, Seattle, WA, USA, Aug. 2008.
- [33] J. Shi, O. Gurewitz, V. Mancuso, J. Camp, and E. Knightly. Measurement and Modeling of the Origins of Starvation in Congestion Controlled Mesh Networks. In *Proc. IEEE INFOCOM*, Phoenix, AZ, USA, Apr. 2008.
- [34] X. Wang and K. Kar. Cross-layer Rate Control in Multi-hop Wireless Networks with Random Access. In *Proc. ACM MobiHoc*, Urbana-Champaign, IL, USA, May 2005.
- [35] K. Whitehouse, A. Woo, F. Jiang, J. Polastre, and D. Culler. Exploiting the capture effect for collision detection and recovery. In *In Proc. IEEE workshop on Embedded Networked Sensors (EmNetS-II)*, Sydney, Australia, Apr. 2005.
- [36] K. Xu, M. Gerla, L. Qi, and Y. Shu. Enhancing TCP fairness in ad hoc wireless networks using neighborhood RED. In *Proc. ACM MobiCom*, San Diego, CA, USA, Sep. 2003.
- [37] Y. Yi and S. Shakkottai. Hop-by-hop Congestion Control Over a Wireless Multi-hop Network. *IEEE Transactions on Networking*, 15:133–144, Mar. 2007.