On Packet Loss Rates in Modern 802.11 Networks

Ramanujan K Sheshadri, Dimitrios Koutsonikolas
University at Buffalo, SUNY, Buffalo, NY
Email: {ramanuja, dimitrio}@buffalo.edu

***Abstract*—The knowledge of link packet loss rates (PLRs) at different PHY layer configurations is vital for a number of wireless network optimization schemes. However, the very large number of PHY layer configurations offered by modern 802.11 n/ac networks has made probing-based PLR estimation at each available configuration extremely challenging. In this paper, we seek to answer the question "How to estimate the PLRs at each available PHY layer configuration with minimal overhead?" Our analysis of the PLR datasets collected from three 802.11 n/ac testbeds reveals that, for any given link, there are several configurations with similar PLR. However, capturing this similarity using well-known link quality indicators like RSSI, or PHY layer features such as MCS or number of MIMO streams is hard. Consequently, we explore the approach of clustering the available PHY layer configurations into a small number of clusters with similar PLR, independent of any other parameter, and only probe one representative configuration in each cluster. Using two real-world case studies – rate adaptation and multihop routing, we show that the proposed clustering-based PLR estimation helps network optimization schemes to reach optimal configurations faster leading to significant performance improvements.**

I. Introduction

The unstable channel conditions in 802.11 networks make periodic link quality estimations an indispensable task in order to optimally exploit the link capacity. A popular link quality metric is the link Packet Loss Rate (PLR), typically measure via *probing* for different PHY layer configurations, e.g., modulation and coding schemes (MCSs). Various WLAN optimization schemes e.g., transmission power control and rate adaptation (RA) algorithms rely on PLR measured at the MAC layer to enhance their performance. Link quality routing metrics in multihop Wireless Mesh Networks (WMN)[1], [2], [3] use PLR at the network layer, alone or in combination with other metrics (e.g., link bandwidth), to find optimal paths between two nodes. However, despite the fact that the PLR is such a heavily relied upon metric, probing-based PLR estimation at each available PHY configuration is extremely challenging in modern 802.11n/ac networks. This is because, unlike legacy 802.11 a/b/g standards which offer only a handful of PHY configurations (different bitrates), the new features introduced in modern 802.11 n/ac standards offer a very large number of different options. E.g., 802.11n offers 128 different bitrate options (represented using different MCS IDs) through 8 different MCSs, 4 MIMO streams, 2 channel widths (20 MHz and 40 MHz), and 2 guard intervals (400 ns and 800 ns). Consequently, the mundane strategy employed in legacy 802.11 a/b/g networks of sequentially probing all the available configurations would result in prohibitively high overhead in 802.11n/ac networks.

Many RA algorithms try to address this problem by using one of the following methods: (i) Discard MCSs that have redundant bitrates completely from consideration. The popular Ath9K RA algorithm (ARA) [4] employs this approach wherein among different MCS IDs that offer the same bitrate only one is considered for active probing. The inherent assumption here is that MCSs with same bitrate will have similar PLR. However, as we show later in this paper, this is not necessarily true. (ii) Use selective probing where only MCSs that are most likely to offer better throughput than the currently used MCS are probed incrementally [5], [6]. Although this approach does reduce the candidate pool of MCSs needed to be probed, it still does not scale in 802.11 n/ac networks.

In the case of link-quality based routing metrics, the inability to accurately estimate the PLR at all the available bitrates results in considerable degradation of throughput [7]. Unlike RA algorithms, routing protocols cannot use data packets to probe for PLR. Instead, they rely on special periodic network-layer probe packets to measure the PLR. In order to avoid excessive overhead, broadcast probes are used instead of unicast probes. However, the broadcast probes, which are typically transmitted at the lowest bitrate, do not provide accurate PLR estimations for higher bitrates that are used to transmit the actual data packets, leading to poor performance of the state-of-the-art link-quality metrics in 802.11 n/ac networks.

In this paper, we seek to answer the question – "How to estimate the PLRs at each available PHY configuration with minimal overhead?" We explore the idea of leveraging any existing similarities in PLRs of different configurations, to collectively estimate the PLRs of many configurations together in order to reduce the overhead. More specifically, we ask two sub-questions: (a) Is there indeed any similarity in the PLRs of different PHY configurations? (b) How to quantify/capture the similarity in PLRs of different configurations? To answer these questions, we collected PLR traces from 170 different links across three different testbeds (two 802.11n testbeds and one 802.11ac testbed). Our analysis reveals that, even though there does exist considerable similarity in PLRs of different configurations, it is hard to capture this similarity using popular link-quality indicators like RSSI, or PHY layer features such as MCS, bitrate, and number of MIMO streams.

Consequently, we explore clustering configurations for a given link independent of any parameter and purely based on their individual PLRs. We employ hierarchical clustering – a widely popular clustering algorithm used for data analysis. Interestingly, we observe that, for any given link, it is possible to cluster all available MCSs into three or four clusters in most

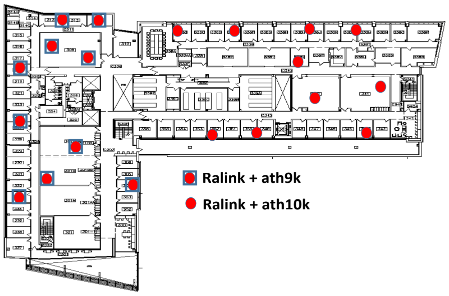


Fig. 1: Floorplan of the UBMesh testbed deployment.

of the cases wherein the difference between the PLR of every MCS in a cluster and that cluster's centroid is less than or equal to 5 percentage points. This observation suggests that probing only one single MCS within each cluster is sufficient to estimate, albeit with some approximation, the PLRs of all the available MCSs within that cluster.

We finally evaluate the potential benefit of this approach in practice by considering two real-world case studies, namely rate adaptation and multihop routing. In the first case study, we select two popular RA algorithms and replace their inherent PLR estimation methods with the proposed PLR estimation through MCS clustering. Our experiments reveal that PLR estimation through MCS clustering elevates their median UDP and TCP throughputs by approximately 30-40%, in both interference and interference-free environments. For the second case study, we consider the popular link quality ETT routing metric [2], [3] and show that estimating PLRs through MCS clustering elevates its median UDP throughput by 138%.

The rest of the paper is organized as follows: Section II describes our testbeds and the PLR dataset. Section III analyzes the collected PLR datasets, followed by the description of the MCS clustering technique in Section IV, and its evaluation via two case studies in Sections V and VI. Section VII discusses the related work and Section VIII concludes the paper.

II. Experimental Setup and Data Collection

Testbed: Our study is conducted on UBMesh [8] (Figure 1), an experimental indoor testbed deployed in an academic building. The testbed consists of 20 nodes, with each node equipped with two WiFi radios. Each node is a desktop PC running Ubuntu Linux 14.04. The first radio in all 20 nodes is a Ralink RT2860 802.11n WiFi card that supports 2X3 MIMO. The second radio in 10 of them is an Atheros Ath9K 802.11n WiFi card that supports 3X3 MIMO, and in the other 10 an Atheros Ath10k 802.11ac WiFi card that supports 3X3 MIMO. All three cards are controlled by their respective open source drivers. In the rest of the paper, we refer to the whole 20 node testbed with Ralink cards in use as Ralink-testbed, the 10 nodes equipped with Atheros Ath9k cards as Ath9k-testbed, and the 10 nodes equipped with Atheros Ath10k cards as Ath10k-testbed.

Dataset collection: The PLR data was collected from all possible links in the testbed (170 links in total), for all the available MCSs in an interference free environment. The Ralink-testbed PLR dataset was collected with 20 MHz channel width in the 2.4 GHz frequency band, the Ath9k-testbed dataset was collected with both 20 MHz and 40 MHz channel widths in the 5 GHz frequency band, and finally the Ath10k-testbed dataset was collected with 20 MHz, 40 MHz, and 80 MHz channel widths in the 5 GHz frequency band.

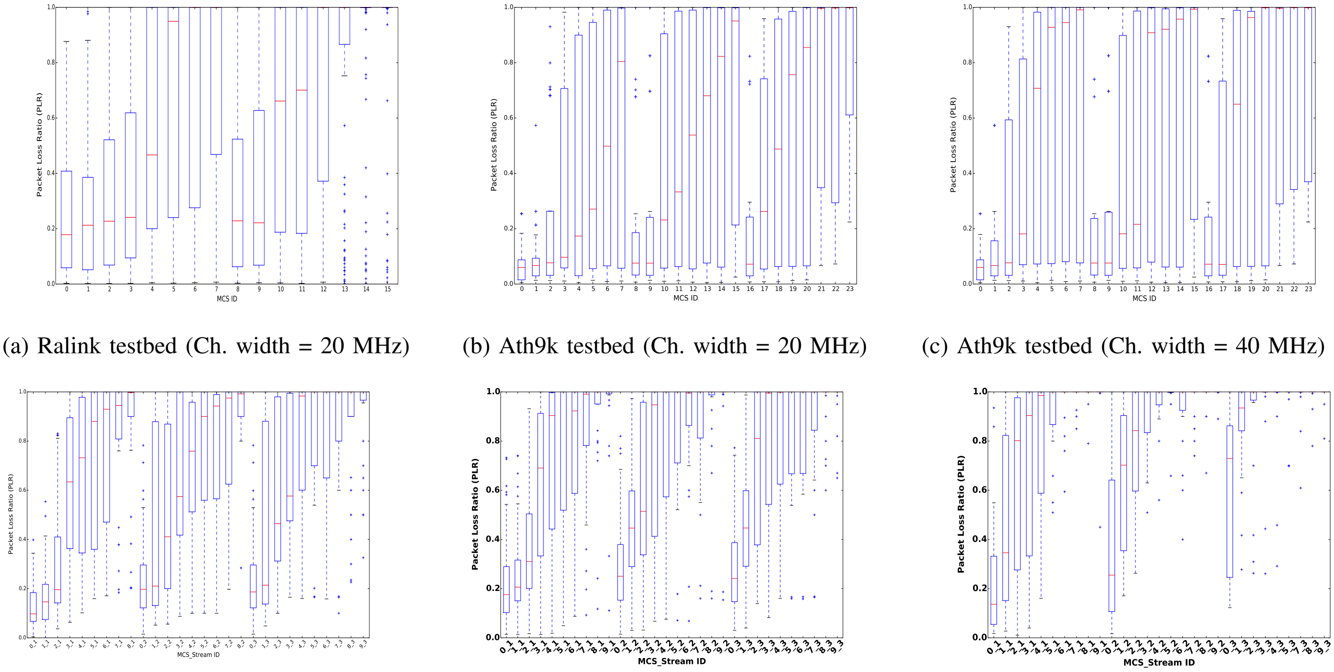
To measure the PLR at each MCS, we disabled the MAC layer retransmissions in the driver and manually fixed the MCS on the sending node. Subsequently, every node in the testbed was in turn made to transmit 1500 byte unicast UDP packet sequentially to all its neighbors at a rate of 100 packets per second for 120 seconds. This process was repeated iteratively for all the available MCSs. The PLR of an MCS was then calculated as the ratio of the number of packets that were either lost or corrupted to the total number of packets sent. Figure 2 shows box-plots of the PLR distribution for each MCS, for all the links in the three testbeds. The graphs show that the PLR datasets we collected are diverse and representative of real world deployments. Additionally, we recorded the RSSI for each link, as reported by the WiFi driver, during the PLR data collection.

Remark: In the case of 802.11n, the MCS index represents a unique combination of modulation, coding rate, and number of MIMO spatial streams, *e.g.,* MCS 8 indicates a configuration of BPSK modulation, coding rate of 1/2, and 2 MIMO streams. The 802.11n 2x3 Ralink testbed supports 16 different MCS Indexes (MCS 0 - MCS 15) and the 802.11n 3X3 Ath9k-testbed supports 24 different MCS Indexes (MCS0 - MCS 23). On the other hand, in the case of 802.11ac, the MCS index represents the unique combination of modulation and coding scheme only and the number of MIMO streams needs to be mentioned explicitly. In Figure 2, we use MCS ID to denote the MCS index in the case of 802.11n (Figures 9a, 9b, 9c), and MCS_Stream ID to denote the combination of MCS and number of streams in the case of 802.11ac (Figures 9d, 9e, 9f), *e.g.,* MCS0_2 represents modulation of BPSK, coding rate of 1/2 and 2 MIMO streams. However, in the remaining of the paper, for simplicity, we use the term MCS ID in both 802.11n and 802.11ac, to denote the combination of modulation, coding rate, and number of MIMO streams, with MCS ID taking different values for each standard (0-23 in 802.11n, 0_1, ..., 9_1, 0_2, ..., 9_2, 0_3, ..., 9_3 in 802.11ac).

III. PLR dataset analysis

A. Rate of change of PLR

The graphs in Figure 2 show an expected trend where the PLRs start to increase with more aggressive MIMO, and/or MCS configurations. However, a closer look at the results reveals a non-uniform rate of increase in the PLR as the MCS ID increases. The change in the PLR from a given MCS to the next one (with same MIMO configuration) is relatively small initially but becomes more drastic after a particular MCS. This pattern appears to be more pronounced in links with moderate and poor signal strengths. To illustrate this with an example, we plot the PLRs for every MCS ID for three sample links from our Ralink-testbed in Figure 3. The three links have good



(a) Ralink testbed (Ch. width = 20 MHz)

(b) Ath9k testbed (Ch. width = 20 MHz)

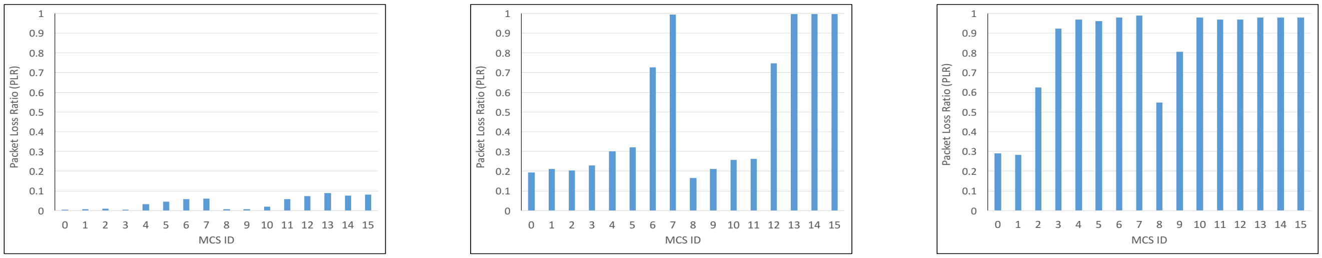
(c) Ath9k testbed (Ch. width = 40 MHz)

(d) Ath10K testbed (Ch. width = 20 MHz)

(e) Ath10K testbed (Ch. width = 40 MHz)

(f) Ath10k testbed (Ch. width = 80 MHz)

Fig. 2: Link PLR boxplots in the three testbeds. Link PLRs span the whole range from 0-100%.



(a) Link-A: High signal strength

(b) Link-B: Moderate signal strength

(c) Link-C: Poor signal strength

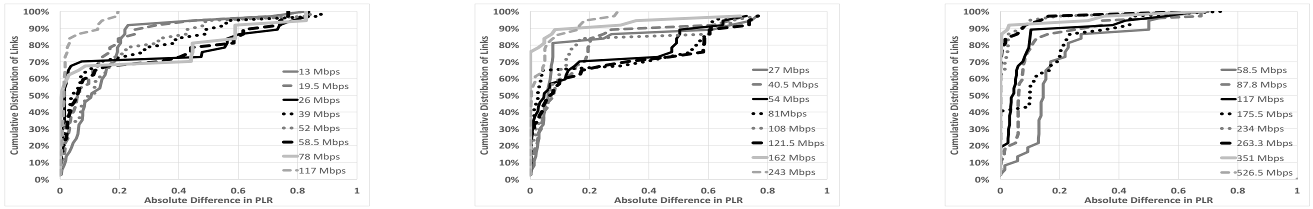
Fig. 3: Rate of change of PLR.

(RSSI -35 dBm), moderate (RSSI -60 dBm) and poor signal strengths (RSSI -80 dBm), respectively. While link-B and link-C clearly exemplify the described pattern in PLR variation, for link-A due to high signal strength the changes in PLRs for different MCS IDs are more subtle. As a result of this non-uniform rate of change in PLRs, many MCS IDs end-up having similar PLRs. For instance, in link-B MCS ID 0, 1, 2, 3, 4, 9, 10 and 11 have PLRs that differ with one another by less than 0.1 (10 percentage points). These MCS IDs not only correspond to different modulation and coding configurations but MCS IDs 0 to 4 implement spatial diversity while MCS 9 to 11 implement spatial multiplexing. Similarly, in both link-A and link-C we see many MCS IDs corresponding to different modulation and MIMO configurations to have very similar PLRs. Although such similarity in PLRs exists in every link, it is not yet established how to capture and quantify it. Motivated by this, we proceed to find ways to capture the similarity in PLRs of different MCS IDs using various PHY layer features or network quality indicators.

B. Do MCS IDs with same bitrate have similar PLRs?

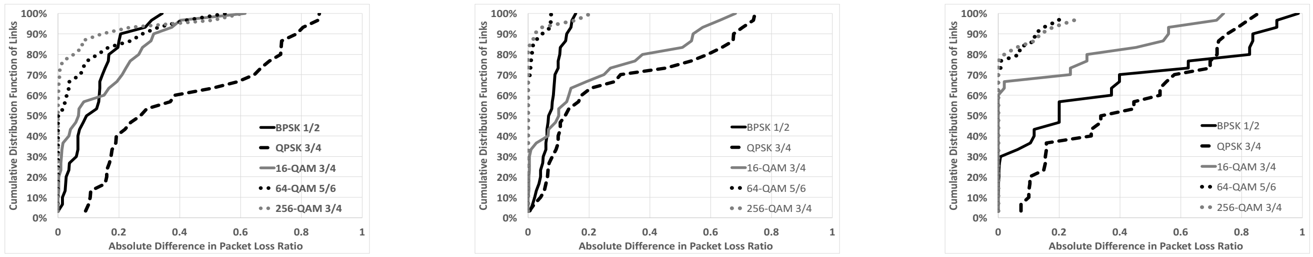
RA algorithms like [4] avoid probing all the available MCS IDs by discarding MCS IDs that offer redundant bitrates and create an MCS ID candidate pool with only monotonically increasing bitrates. For example, since MCS ID '5' and MCS ID '11' both results in a bitrate of 52 Mbps (with a 20 MHz channel width), [4] considers only MCS ID '11' and discards MCS ID '5'. The inherent assumption is that the MCS IDs with same bitrates would have similar PLRs.

Figure 4 plots the Cumulative Distribution Function (CDF) of the absolute difference between the maximum and minimum PLRs of MCS IDs with same bitrate, for all the links in the Ath10k-testbed. The results for the Ralink-testbed and the Ath9k-testbed are similar and omitted due to space constraints.



(a) Ath10k testbed (Ch. width = 20 MHz)
(b) Ath10k testbed (Ch. width = 40 MHz)
(c) Ath10k testbed (Ch. width = 80 MHz)

Fig. 4: Difference in PLRs of MCS IDs with same bitrate.



(a) Ath10k testbed (Ch. width = 20 MHz)
(b) Ath10k testbed (Ch. width = 40 MHz)
(c) Ath10k testbed (Ch. width = 80 MHz)

Fig. 5: Difference in PLR of MCS IDs with same MCS configuration but different MIMO configuration.

The graphs show significant dissimilarity with the absolute difference extending upto 80 percentage points (0.8) in certain cases. Likely causes for such large differences include different MIMO and MCS configurations of MCS IDs with the same bitrate. E.g., MCS ID '5' uses spatial diversity with QPSK modulation, while MCS ID '11' uses spatial multiplexing with BPSK modulation. Depending on the channel characteristics, one feature or the other (modulation vs. MIMO configuration) may play a more important role in determining the link PLR.

C. Do MCS IDs with same MCS configuration have similar PLRs?

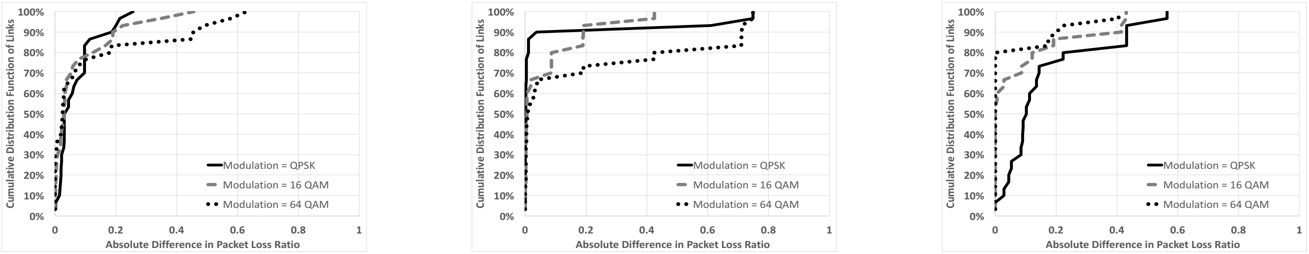
We now verify if different MCS IDs with the same MCS configuration have similar PLRs irrespective of their MIMO configuration. For each link we calculate the absolute difference between the maximum and the minimum PLR in the PLR set obtained from MCS IDs with same MCS configuration. Figure 5 shows the CDF of the difference in PLR between MCS IDs with the same modulation and coding scheme for all the links in Ath10k-testbed. We observe that MCS IDs with same MCS configuration do not necessarily have similar PLRs. Even among MCS IDs with robust MCS configurations like BPSK-1/2 and QPSK-3/4 the PLRs often change considerably. On the other hand, MCS IDs with very aggressive MCS configurations like 64-QAM-5/6, and 256-QAM-3/4 often have similar PLRs, close to 100%, especially with 40 MHz and 80 MHz channel widths. Overall, we observe that the MIMO configuration for a link can affect the PLR significantly for a given modulation and coding scheme.

D. Do MCS IDs with same modulation and MIMO configuration have similar PLRs?

For each MIMO configuration (number of MIMO streams), we cluster MCS IDs with same modulation (but different coding rate), and verify if the PLRs of the MCS IDs within each cluster are similar. E.g., in an 802.11n link, a QPSK modulation with 1-stream MIMO configuration (spatial diversity) would have MCS IDs '1' and '2' clustered together, and a 64-QAM modulation with 1-stream MIMO configuration would have MCS IDs '5', '6', and '7' clustered together. Figure 6 plots the CDF (Ath10k-testbed) of the absolute difference between the maximum and minimum PLR in the PLR set obtained from all MCS IDs that have same modulation configuration and 1-stream MIMO configuration (spatial diversity). We observe that there is reasonable similarity in PLRs of MCS IDs with lower modulations (QPSK) for channel widths 20 MHz and 40 MHz. However, the similarity reduces for higher modulations like 16-QAM, and 64-QAM. The picture is different at 80 MHz. Most of the MCS IDs with higher modulations tend to have similar PLRs close to 100% while PLRs of MCS IDs with lower modulations (QPSK) tend to be significantly dissimilar.

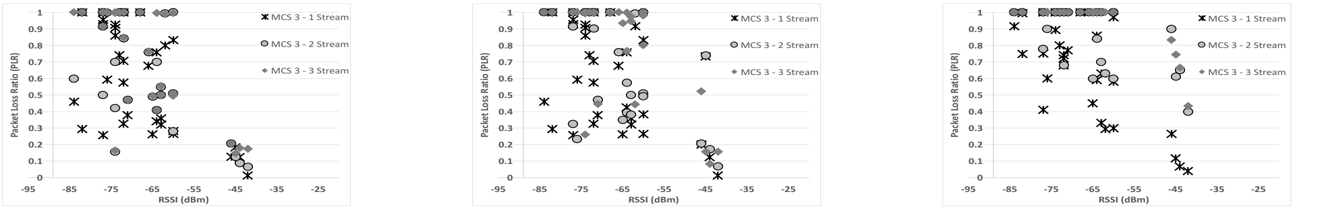
E. Do links with similar RSSI have similar PLRs?

RSSI has been one of the most accessible link quality indicators. If links with similar RSSI have similar PLRs, then links with similar RSSI could probe different subsets of MCS IDs and share their results with each other, contributing to a low-overhead collaborative PLR estimation scheme. Figure 7 plots the PLR distribution for three different MCS IDs (3_1,



(a) Ath10k testbed (Ch. width = 20 MHz) (b) Ath10k testbed (Ch. width = 40 MHz) (c) Ath10k testbed (Ch. width = 80 MHz)

Fig. 6: Difference in PLR of MCS IDs using same MCS configuration and MIMO spatial diversity (1 stream).



(a) Ath10k testbed (Ch. width = 20 MHz) (b) Ath10k testbed (Ch. width = 40 MHz) (c) Ath10k testbed (Ch. width = 80 MHz)

Fig. 7: PLR distribution Vs. RSSI.

3_2, 3_3), obtained from each link in the Ath10k-testbed, against the link's measured RSSI. We observe that the PLRs for the same MCS ID in different links with very similar RSSI can be very different, up to 80% in a few cases, *e.g.,* in Figure 7a when RSSI = -75dBm. We conclude that the intuitive idea of collaborative PLR estimation in links with similar RSSI cannot provide accurate results. The performance of MIMO links highly depends on the multipath structure and RSSI is too coarse-grained to capture differences in the multipath structure which can result in significantly different PLR [9], [10].

We thus conclude that the PLRs of different MCS IDs cannot be captured by PHY layer features such as bitrate, MCS, or MIMO configuration at the link level or by standard link quality indicators like RSSI across different links. Consequently, we now now take a step back and analyze the PLR distribution in each link irrespective of any parameter to better understand how PLRs vary for different MCS IDs.

IV. Hierarchical Clustering

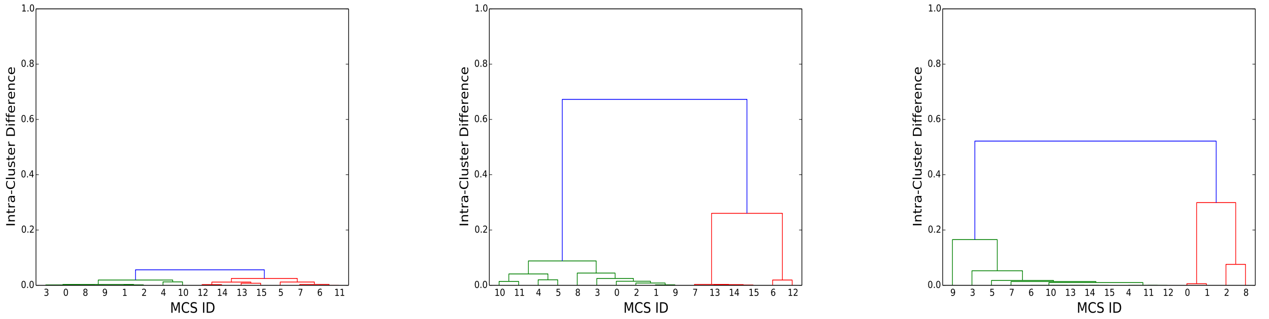
Hierarchical clustering is a popular clustering technique widely used for data analysis. The advantage of employing hierarchical clustering as opposed to other popular clustering techniques like K-means is that hierarchical clustering outputs a hierarchy, a structure that is more informative than the unstructured set of clusters returned by other flat clustering techniques. Moreover, hierarchical clustering does not require us to pre-specify the number of clusters. In this section, we use hierarchical clustering to cluster the MCS IDs of a given link in clusters with similar PLR.

Hierarchical clustering measures the similarity between two data points using the euclidean distance between them. In our case, since the dataset has only one PLR value for each MCS ID (per link), the similarity between MCS IDs is measured as the absolute value of the numeric difference between their respective PLRs.

The clustering algorithm uses an agglomerative (bottom-up) approach where each MCS ID starts as a single element in its own cluster, and in each subsequent iteration two clusters that have the smallest difference between their respective centroids (calculated as the average PLR of the cluster) are merged together until all the clusters merge together to form one cluster.

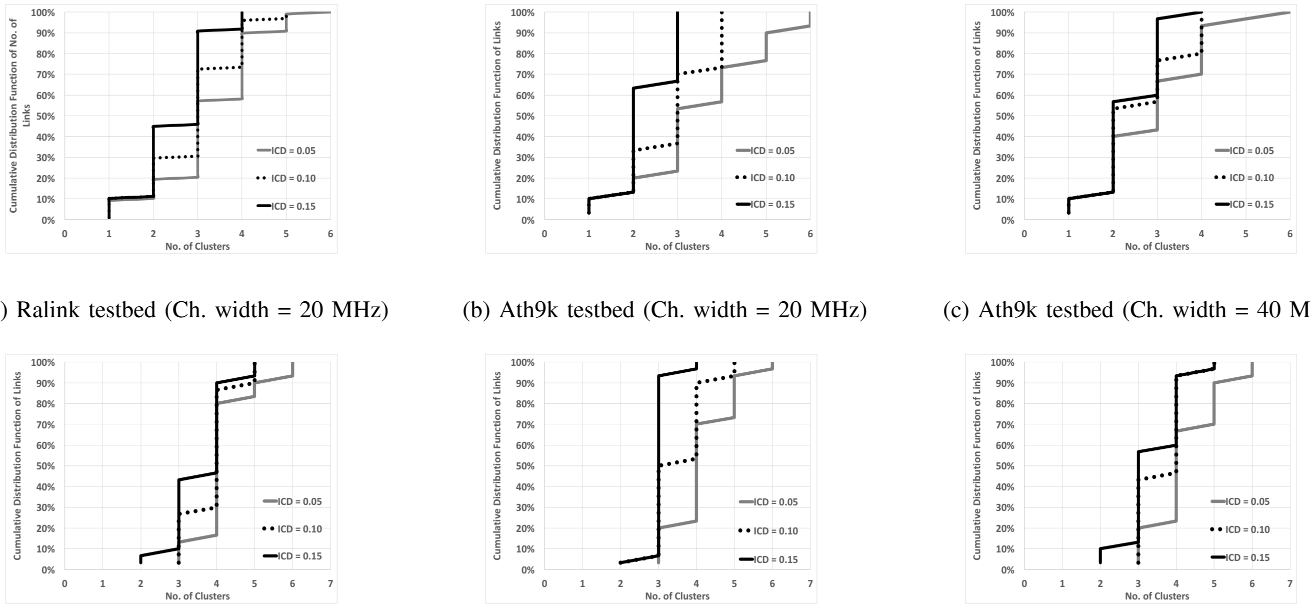
To illustrate the clustering process with an example, we plot dendrograms in Figure 8, for the three sample links - link-A, link-B, and link-C – that we considered earlier in Figure 3. The y-axis of the dendrograms shows the Intra Cluster Difference (ICD) for each cluster. The ICD represents the upper bound of the absolute difference between the PLR of an MCS ID in the cluster and the cluster's centroid. In other words, an ICD value of 'X' ($X > 0$) indicates that in any given cluster, the absolute difference between the PLR of any MCS ID in the cluster and that cluster's centroid should be in the range [0, 'X'].

For a given ICD value, the number of clusters formed varies for each link depending on how similar the PLRs of different MCS IDs are for that link. E.g., in the case of link-A, since all the PLRs are close to each other, an ICD of 0.1 can accommodate all the MCS IDs into a single cluster. On the other hand, for link-B and link-C, an ICD of 0.1 results in 3 and 4 different clusters, respectively. Decreasing the ICD increases the number of clusters formed for a given link, *e.g.,*



(a) Link-A: High signal strength (b) Link-B: Moderate signal strength (c) Link-C: Poor signal strength

Fig. 8: Dendrograms showing hierarchical clustering for the three links from Figure 3.



(a) Ralink testbed (Ch. width = 20 MHz) (b) Ath9k testbed (Ch. width = 20 MHz) (c) Ath9k testbed (Ch. width = 40 MHz)

(d) Ath10k testbed (Ch. width = 20 MHz) (e) Ath10k testbed (Ch. width = 40 MHz) (f) Ath10k testbed (Ch. width = 80 MHz)

Fig. 9: Number of clusters per link, for different ICDs in the Ralink, Ath9k, and Ath10k testbeds.

an ICD of 0.05 results in 2, 4, and 5 clusters for link-A, link-B and link-C, respectively.

The CDF graphs in Fig. 9 show the number of clusters formed for each link in the three testbeds for ICD values – 0.05, 0.10, and 0.15. Interestingly, we see that even for a small ICD value of 0.05, the median number of clusters formed in all the cases is less than or equal to 4 and the maximum number of clusters at most 6. Increasing the ICD results in fewer clusters at the cost of smaller similarity between PLRs of MCS IDs within a cluster.

A. *Significance of clustering*

Recall that our goal is to identify similarities in PLRs of different MCS IDs that can be leveraged during PLR estimation. With this in mind, we now discuss the significance of clustering MCS IDs during PLR estimation. Clustering MCS IDs raises an interesting prospect of representing the PLRs of all the MCS IDs in each cluster collectively as one *cluster-PLR*, instead of representing the PLR of each MCS ID individually. This reduces the entire MCS ID set to be represented by few clusters. Consequently, it should suffice to probe one MCS ID from a cluster to estimate the PLRs of all the MCS IDs in the cluster. The number of probes required to estimate PLRs of all the MCS IDs is equal to the number of clusters formed.

However, using a single value to represent the PLRs of all MCS IDs within a cluster will undoubtedly lead to erroneous PLRs. Thus, it is important to first analyze how much error is affordable. E.g., if we assume that the PLR of all MCS IDs within each cluster can be represented by the centroid value of the cluster, then an ICD of 0.1 guarantees that the error in the PLR estimation (actual PLR - cluster-PLR) will be $<= 0.10$ (10 percentage points). But is this error acceptable or should the ICD be lowered further? The answer to this largely depends on the application under consideration. A low ICD will ensure a low error, but will increase the number of clusters (and thus the overhead), while a high ICD will decrease the number of clusters but increase the error for each MCS ID.

In the following, we evaluate the efficacy of determining PLRs using MCS ID clusters in two real-world case studies: rate adaptation and multihop routing. The broader question wen are answering via these two studies is: *Is approximate estimation of PLRs of all available MCS IDs preferable to accurate estimation of PLRs of a only few MCS IDs?*

V. Case Study I: Rate Adaptation

A. *Ath9k RA: (ARA)*

We begin with the ARA algorithm which is the default RA algorithm used in the Ath9k driver. As previously mentioned, ARA excludes MCS IDs that have redundant bitrates from consideration. The algorithm sets up a rate table of MCS IDs containing monotonically increasing bitrates and employs a simple walk-up-and-down approach on the rate table to appreciate and depreciate the MCS ID selected for transmission. Periodically, the algorithm probes the MCS ID immediately above the currently used MCS ID in the rate table and if the probe is successful, the current MCS ID is appreciated. When the PLR of the current MCS ID drops below a threshold, it is depreciated to the MCS ID immediately below the current MCS ID in the rate table.

Implementing MCS ID clustering: While we continue to keep the core idea of the algorithm, *i.e.,* the walk-up-and down approach, we modify the way in which the MCS IDs are stored in the rate table. In our modification, we first include all available MCS IDs and arrange them in increasing order of their bitrates. As the PLRs of different MCS IDs are measured, we begin clustering MCS IDs (using ICD = 0.5) based on their respective PLRs. The clusters are arranged in increasing order of their highest bitrate. Periodically during a probing interval, we probe a random MCS ID in the cluster immediately higher in the rate table and if successful, the current MCS ID is appreciated to the MCS ID that has the highest bitrate in that cluster. In case the PLR of the MCS ID in the current cluster drops below a threshold, the current MCS ID is depreciated to the MCS ID that has the highest bitrate in the cluster immediately below the current cluster.

Advantages of MCS ID clustering: First, since all the MCSs are included in the modified algorithm, the chance of selecting the optimal MCS ID is higher. Second, instead of moving up and down among individual MCS IDs, moving up and down clusters ensures that we skip many MCS IDs that have PLRs similar to the currently used. We evaluated the performance of the original and modified ARA algorithm over all the links in the Ath9k testbed with a channel width of 20 MHz, with and without interference, using iperf. The interference-free experiments were run at night while the interference experiments were conducted during the day time when people in the building used the campus WiFi as usual. For each link, we ran each version of the algorithm 5 times and calculated the average throughput. Figures 10a and 10b plot the CDF of the average UDP and TCP throughput without/with interference. We observe that the modified ARA using MCS ID clustering improves both UDP and TCP throughputs by nearly 28% and 30%, respectively, in the median case in an interference-free environment and by 34% and 40%, respectively, in the median case, in the presence of interference.

B. *Minstrel_HT: (MHT)*

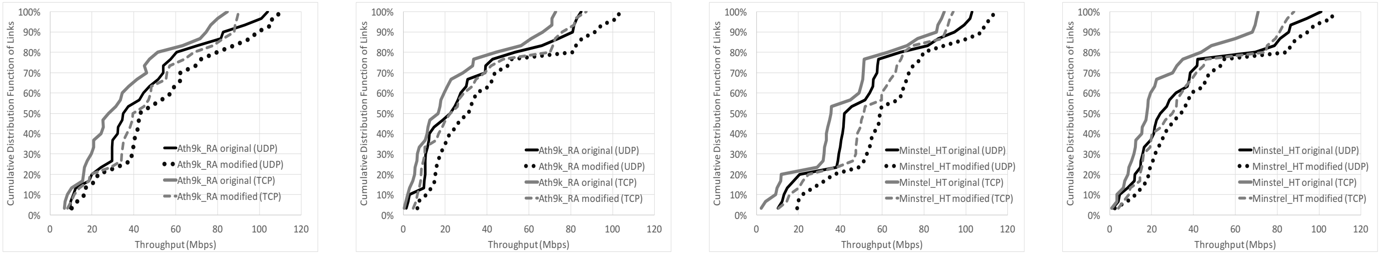
The MHT algorithm [11] divides the available bitrate pool into groups based on varying number of MIMO streams, channel width, and guard interval. It then randomly selects MCS IDs from each group to populate a sampling table. Periodically, the algorithm probes the next MCS in the sampling table and updates the MCS ID's Estimated Moving Average of its PLR. The MCS ID to use is then selected as the MCS ID with the highest link capacity – calculated as the product of delivery probability (1 - PLR) and bitrate of that MCS ID.

Implementing MCS ID clustering: Like in the case of ARA, we modify the algorithm to cluster MCS IDs (ICD = 0.05) based on their individual PLRs. We keep the random probing part as it is. When the PLRs of probed MCS IDs are updated, we use these updated PLRs to re-calculate the centroid of the cluster the MCS ID is part of, and use this new centroid value to update the PLR of the remaining un-probed MCS IDs of that cluster. Finally, the optimal MCS ID is calculated like before – as the MCS ID with the highest link capacity.

Advantages of MCS clustering: The major criticism of MHT is the randomization in probing. Very often the algorithm gets stuck using sub-optimal MCS IDs because the PLR statistics of optimal MCS IDs would not be updated. Clustering helps overcome this issue by allowing the PLRs of the un-probed MCS IDs to be updated based on the PLRs of other probed MCS IDs within the same cluster. As a result, the updated PLRs of the un-probed MCS IDs are adjusted to better reflect the changing channel conditions. The CDFs in Figures 10c and 10d, obtained with the same methodology as in Section V-A, show that the modified MHT algorithm improves the median TCP and throughputs by 35% and 36%, respectively, without interference, and by 36% and 34%, respectively, with interference.

VI. Case study II: Routing in WMNs

Routing in multihop WMNs has always been a difficult challenge as links often suffer from poor quality due to interference and inherent channel conditions. Consequently, a number of link quality based routing metrics have been proposed over the past 15 years, e.g., [1], [2], [12], [13]. Here we consider the popular Expected Transmission Time (ETT) metric [2], [3]. The ETT metric uses the link PLR and the link bandwidth (*B*) to estimate the time required to successfully transmit a unicast packet of size *S*, over a link, including MAC retransmissions. It can be calculated in one of the following two ways. *Method I*: Using the formula $ETT = \frac{1}{1-PLR} * \frac{S}{B}$ [2], where the link bandwidth is estimated using the packet-pair technique [14]. *Method II*: By estimating the throughput for each MCS as the product between the



(a) ARA – No interference (b) ARA – Interference (c) MHT – No interference (d) MHT – Interference

Fig. 10: CDF of throughputs measured with original and modified versions of Ath9k RA (ARA) and Minstrel_HT (MHT).

link delivery probability at that MCS (1-PLR) and the MCS's bitrate, selecting the highest throughput ($hi-thput$), and using this value to calculate ETT as $ETT = \frac{S}{hi-thput}$ [3]. The ETT metric of a path in both these cases is calculated as the sum of all the ETTs of individual links in the path.

While the throughput gains of ETT over other routing metrics such as the traditional Hop Count and ETX [1] remain undisputed in legacy 802.11a/b/g networks, in our recent work [7] we showed that these throughput gains do not carry forward into modern 802.11 n/ac networks. The main reason for this is attributed to the methods (suitable for legacy 802.11 a/b/g networks) used to estimate the PLRs and the link bandwidth. In particular, the packet pair technique for link bandwidth estimation is proved to be unsuitable for 802.11n/ac networks in [7], making the first method for ETT calculation unusable. We thus consider Method II to calculate ETT, and focus on improving PLR estimation through MCS ID clustering.

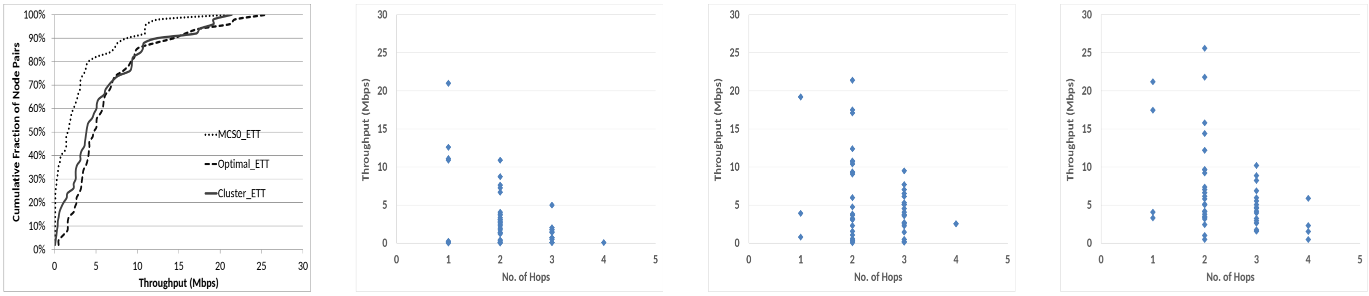
As we mentioned previously, unlike RA algorithms, routing protocols do not use data packets to estimate link PLRs, but instead rely on special network-layer probe packets sent periodically. Since sending these probes at all MCSs may result in high overhead, the approach in [1], [2] is to use broadcast probes sent only at the lowest MCS ID, and to use this estimate for all higher MCS IDs as well. While this approach was shown to work well for legacy 802.11a/b/g networks, it often fails in 802.11 n/ac networks [7] which offer a various PHY configurations like different channel widths and MIMO configurations, in addition to different modulation and coding schemes. On the other hand, probing all possible MCS IDs [3] is not a viable option in modern 802.11n/ac networks due to excessive overhead. Keeping this in mind, we now proceed to test if the proposed clustering technique for PLR estimation can help improve the ETT metric's performance in 802.11n/ac networks.

A. Experiment setup

To isolate the impact of network-wide probing and focus on the effectiveness of the proposed PLR estimation through MCS ID clustering, we used static routing based on our PLR dataset. In practice, we envision that network-wide probing at all available MCS IDs will take place at coarse time scales (e.g., once a day during a downtime period) to form the clusters, while cluster-based probing would be used in fine-grained intervals (e.g., every second [1], [2], [7]) to update the PLR estimates. We compare the UDP throughput achieved with three different versions of the ETT metric, differing in the way the highest–link–throughput is determined: (a) Using MCS ID clustering (Cluster_ETT): Cluster all the MCS IDs for each link (with ICD = 0.05) using the PLRs from our dataset, and for each cluster calculate the product of the centroid delivery rate and the highest bitrate among the bitrates of all MCS IDs in the cluster. Select the highest product among all the clusters as the highest–link–throughput. (b) Using PLR of MCS ID 0 (MCS0_ETT): Calculated as the maximum product of MCS ID 0's PLR and bitrate. This is based on the assumption that PLR at MCS 0 can approximate the PLR at any MCS [1] and serves as the lower performance bound. (c) (Optimal_ETT): Calculate the product of each MCS ID's PLR and its corresponding bitrate, and select the maximum product. This approach is not practical due to the high overhead of probing all at available MCS IDs and is used as the upper performance bound. The packet size for the three ETTs is assumed to be same and equal to 1500 bytes. The ETT values calculated with each method are used as link weights for the *Dijkstra's* algorithm which is run offline to obtain the minimum ETT paths for 50 random node pairs. The path returned for each node pair with a given ETT method is fixed and Iperf traffic is used to measure the UDP throughput. To avoid any uncertainties caused due to a sub-optimal MCS selection by the underlying RA, we manually fix the MCS ID that gives the highest throughput (based on the above calculations) for each link at the transmitting node. The results for each node pair are averaged over 5 runs.

B. Results

Figure 11a shows the CDF of the average UDP throughputs achieved for each node pair in the three cases. Cluster_ETT outperforms MCS0_ETT, improving the median throughput by 138% (3.8 Mbps Vs. 1.6 Mbps). Interestingly, the median throughput of Cluster_ETT is lower than the median throughput of Optimal_ETT by only 21% (3.8 Mbps Vs. 4.6). Figures 11b-11d show the reason for the throughput gains of Cluster_ETT over MCS0_ETT. Unlike MCS0_ETT,



(a) CDF of Throughput (b) MCS0_ETT (c) Cluster_ETT (d) Optimal_ETT

Fig. 11: Throughput gains and Throughput vs. Path Length with MCS0_ETT, Cluster_ETT, and Optimal_ETT.

Cluster_ETT and Optimal_ETT tend to select more longer paths consisting of high quality shorter hops. Interestingly, the path length of Cluster_ETT is same to that of Optimal_ETT in nearly 80% of the cases, while MCS0_ETT selected shorter paths than Optimal_ETT in nearly 60% of the cases. This proves that the Cluster_ETT is able to successfully differentiate between high and low throughput paths.

VII. Related Work

Even though the topic of loss rate in 802.11 networks has been extensively studied in the past, very few works have actually focused on evaluating link PLRs [15], [16], [10]. While [15] focuses on evaluating the packet delivery rate in 802.11n links when operating only at the highest bitrate, [16] focuses on the impact of channel bonding on PLR. However, both these studies are not concerned with the actual estimation of the PLRs, which is the core focus of this work. [10] proposes a delivery model based on "effective SNR" using the Channel State Information (CSI) received from feedback, to estimate the packet delivery rate. The major drawback of this approach is that the CSI relayed back may suffer from feedback delay, and the SNR based CSI itself may not be always be accurate [17]. Moreover, many commercial WiFi NICs do not expose the CSI data to the higher layers making this approach difficult to use in practice.

VIII. Conclusion

In this work, we studied PLRs in modern 802.11 networks, trying to answer the question "How to estimate the PLRs at each available PHY layer configuration with minimal overhead". Our analysis of 3 PLR datasets revealed that there is considerable similarity in the PLRs of different MCSs, however, capturing this similarity using well-known link quality indicators such as RSSI, or PHY layer features such as MCS, bitrate, or number of MIMO streams is hard. Consequently, we proposed clustering different MCS IDs based on their PLR only, independent of any other parameter, and showed that, for a given 802.11 n/ac link it is always possible to cluster all available MCS IDs into a small number of clusters. This observation suggests that probing only one single MCS within each cluster is sufficient to estimate, albeit with some approximation, the PLRs of all the available MCSs within that cluster. Finally, we demonstrated the efficacy of the proposed cluster-based PLR estimation using two real world applications – rate adaptation in WLANs and multihop routing in WMNs.

References

[1] D. S. J. D. Couto, D. Aguayo, J. C. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," in *Proc. of ACM MobiCom*, 2003.

[2] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *Proc. of ACM MobiCom*, 2004.

[3] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and evaluation of an unplanned 802.11b mesh network," in *Proc. of ACM MobiCom*, 2005.

[4] "ath9k foss wireless driver for atheros ieee 802.11n pci/pci-express based chipsets," http://wireless.kernel.org/en/users/Drivers/ath9k.

[5] I. Pefkianakis, Y. Hu, S. H. Wong, H. Yang, and S. Lu, "MIMO Rate Adaptation in 802.11n Wireless Networks," in *Proc. of ACM Mobicom*, 2010.

[6] B. Radunovic, A. Proutiere, D. Gunawardena, and P. Key, "Dynamic channel, rate selection and scheduling for white spaces," in *Proc. of ACM CoNEXT*, 2011.

[7] R. K. Sheshadri and D. Koutsonikolas, "Comparison of Routing Metrics in 802.11n Wireless Mesh Networks," in *Proc. of IEEE INFOCOM*, 2013.

[8] "UBMesh: An 802.11a/b/g/n wireless mesh network testbed at UB." http://www.cse.buffalo.edu/faculty/dimitrio/research/ubmesh/index.html.

[9] L. Deek, E. Garcia-Villegas, E. Belding, S.-J. Lee, and K. Almeroth, "Joint rate and channel width adaptation for 802.11 mimo wireless networks," in *Proc. of IEEE SECON*, 2013.

[10] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Predictable 802.11 packet delivery from wireless channel measurements," in *Proc. of ACM SIGCOMM*, 2010.

[11] F. Fietkau and D. Smithies, "minstrel_ht: New rate control module for 802.11n." http://lwn.net/Articles/376765.

[12] S. Kim, O. Lee, S. Choi, and S.-J. Lee, "MAC-Aware routing metric for 802.11 wireless mesh networks," in *Proc. of the IEEE (PIMRC)*, September 2009.

[13] Y. Yang, J. Wang, and R. Kravets, "Designing routing metrics for mesh networks," in *Proc. of WiMesh*, 2005.

[14] S. Keshav, "A control-theoretic approach to flow control," *Proc. of the conference on Communications architecture and protocols*, 1993.

[15] K. Pelechrinis, T. Salonidis, H. Lundgren, and N. Vaidya, "Experimental characterization of 802.11n link quality at high rates," in *Proc. of WiNTECH*, 2010.

[16] L. Deek, E. Garcia-Villegas, E. Belding, S.-J. Lee, and K. C. Almeroth, "The impact of channel bonding on 802.11n network management," in *Proc. of the 7th ACM/SIGCOMM International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, December 2011.

[17] S. Gatreux, V. Erceg, D. Gesbert, and R. W. Heath, "Adaptive modulation and mimo coding for broadband wireless data networks," in *Proc. of IEEE Communications*, 2002.