Experimental Evaluation of Application Performance with 802.11 PHY Rate Adaptation Mechanisms in Diverse Environments

Yi Yang, Mahesh Marina and Rajive Bagrodia
Mobile Systems Lab
Computer Science Department
University of California, Los Angeles
Email: {yangyi, mahesh, rajive}@cs.ucla.edu

Abstract— **We examine the impact of physical layer rate adaptation mechanisms on the performance of real applications over 802.11 wireless links in diverse channel environments. Our evaluations are based on a testbed with real wireless devices equipped with commodity 802.11 hardware and a hardware channel emulator. We consider two different and well-known 802.11 rate adaptation mechanisms (Onoe and SampleRate) and study their performance under several realistic workloads, including multimedia streaming and web browsing. We observe that the application performance with different rate adaptation mechanisms is dependent on the specific tradeoffs these mechanisms make at the link layer in an application-oblivious manner between improving throughput and limiting frame loss. More importantly, their relative performance for a given workload is quite sensitive to the channel quality and environment. These observations highlight the importance of choosing the rate selection strategy adaptively in an application and channel aware manner.**

I. Introduction

Wireless LAN (WLAN) technology based on IEEE 802.11 standard [1] is being commonly used in offices and hotspots for indoor wireless Internet access. The success of 802.11 technology has led to newer usage scenarios, including community mesh networks [2] and multimedia distribution/data networking in the home. Given the widespread use of 802.11-based networks, it is critically important to understand the performance characteristics of wide range of applications (with different QoS requirements) on such networks when operating under diverse channel environments.

In this paper, our focus is on 802.11 physical layer (PHY) data rate adaptation mechanisms that are usually implemented in the 802.11 MAC layer. Such a mechanism adjusts the data rate in response to time-varying channel conditions: the basic idea is to exploit good channel conditions by using higher rates for improved efficiency (throughput), and improve transmission reliability when channel gets worse by lowering the rate. The 802.11 PHY provides several widely different data rates (differing in modulation and coding) for use by higher layers — 802.11b rates range from 1 to 11Mbps, whereas 802.11a/g extend this range to 54Mbps. In such settings, clearly the PHY rate adaptation (via dynamic rate selection) plays a key role in determining performance observed at higher layers. This together with the fact that 802.11 MAC/PHY specifications leave the rate adaptation mechanism to vendor discretion led to many proposals to optimize throughput with some constraint on frame error rate (FER).

Several recent studies have been directed toward experimental evaluation of 802.11 PHY rate adaptation mechanisms [3], [4], [5], [6], [7]. Notwithstanding their contribution to the understanding of real-world performance of various rate control mechanisms, these studies have one main limitation: they are largely based on throughput measurements with backlogged UDP traffic. At first glance, this may seem reasonable given that the primary motivation behind rate adaptation is throughput improvement by taking advantage of good channel conditions. However, this metric alone is not sufficient to predict application layer performance in general as it may also depend on additional metrics such as frame loss rate; these metrics in turn are affected by the interactions among rate adaptation, MAC ARQ mechanism, frame length etc. Moreover, some of these studies [3], [6] are specific to a certain channel environment (specifically, an office environment), whereas the rest of them are limited in their experiment control (in terms of configurability and reproducibility) or realism.

In this paper, our goal is experimental characterization of the interaction between applications and PHY rate adaptation mechanisms over 802.11 wireless links in different environments. Towards this end, we take a unique approach through the use of a hardware channel emulator [8]. Such a channel emulator provides us with a highly realistic testbed due to its detailed signal-level emulation of the wireless channel. In addition, it offers high degree of control in terms of experimenting with a wide range of channel conditions in a repeatable manner. For our evaluations, we consider three different channel environments representing home, office and suburban usage scenarios respectively; these environments are realized using a subset of channel models being considered by the 802.11 Task Group n (TGn) [9]. Further, the real-time nature of the emulator permits application-level perfor-

This work has been funded by the NSF under the Network Research Testbed grant “WHYNET: Scalable Testbed for Next Generation Mobile Wireless Networking Technologies” (award number ANI-0335302).

mance evaluation when real wireless devices (running real applications) such as laptops with commodity 802.11 hardware communicate via the emulator.

We focus on FER-based rate adaptation mechanisms (e.g., [4], [10], [3], [11]) in this paper as they are relatively more practical than the alternative set of SNR-based mechanisms (e.g., [12], [13]) and naturally robust across different channel environments [3]. In particular, our evaluations consider Onoe [10] and SampleRate [4], [2] as two representative rate adaptation mechanisms. We choose these two specific mechanisms as they were not only shown to be the most effective among FER-based mechanisms [4] but also are sufficiently different in their design.

As per applications, we use a broad set of realistic workloads consisting of CBR/UDP traffic, adaptive video streaming [14], large file transfers and web (HTTP/TCP) traffic. Together these workloads cover dominant types of traffic characteristics of 802.11 networks [15], albeit indirectly in some cases. For instance, we do not directly consider peer-to-peer (p2p) traffic which amounts to nearly 20% of overall workload in [15]; instead it is captured via file transfer and web traffic in our workloads, which is reasonable given that most p2p data traffic uses TCP either directly or via HTTP [16].

Our results indicate that the PHY rate selection strategy must be adaptively chosen based on *both* channel quality and application characteristics for best overall application-level performance. In the process of optimizing raw throughput at the link layer in an application-independent manner, different rate adaptation mechanisms (with different levels of aggressiveness) provide markedly different tradeoffs with the number of packet losses seen by higher layers. Consequently, their throughput and loss rate behaviors at the link layer can be quite different. The net effect of this tradeoff on the application performance depends on the application-specific characteristics as might be expected. We quantify the impact of this tradeoff for Onoe and SampleRate on performance of several common applications under realistic channel environments.

More interestingly, the observed performance of a given application with different rate adaptation mechanisms is quite sensitive to the channel quality and the environment with neither Onoe nor SampleRate exhibiting superior performance throughout. Specifically, *significant performance reversals* are seen between the two mechanisms across different regions of the parameter space. For instance, SampleRate performance of TCP bulk file transfer with TGn channel model D is 5x better than Onoe for low path loss values, whereas it becomes 10x worse for very high path loss; we observe similar patterns for other workloads as well. The above observations suggest that throughput optimization at the link layer in an application-oblivious manner is ineffective when designing PHY rate adaptation mechanisms. Rather these mechanisms in conjunction with other MAC operations (including frame size selection, scheduling and ARQ) must be attuned to the application requirements when adapting to varying channel conditions, further emphasizing the need for cross-layer optimization.

The rest of the paper is organized as follows. In Section II, we briefly describe the two rate adaptation mechanisms used in our study, namely Onoe and SampleRate. Section III describes our experimental setup. Section IV presents our results and forms the core of the paper; this section is further divided into several subsections according to the type of workload. We conclude in Section V.

II. 802.11 PHY Rate Adaptation Mechanisms

Existing 802.11 PHY rate adaptation mechanisms can be classified into two broad categories depending on how they estimate the channel quality. In FER-based mechanisms [4], [3], [11], the channel quality is implicitly estimated from FER (or related) measurements obtained by intermittent probing at higher rates (much like the way TCP probes by increasing the sending rate to estimate available bandwidth). On the other hand, SNR-based mechanisms [12], [13], explicitly estimate the channel quality using the physical layer information (e.g., SNR) typically at the receiver. Here we limit our attention to FER-based mechanisms as they are relatively more practical and naturally robust across different channel environments [3], and leave comprehensive evaluation also involving SNR-based schemes and hybrid schemes [6] for future work. Among the FER-based mechanisms, we consider Onoe [10] and SampleRate [4], [2] as two representative rate adaptation mechanisms because they were not only shown to be the most effective in their category [4] but also are sufficiently different in their design. Below we briefly review these two mechanisms to serve as background for the rest of the paper.

A. Onoe

Onoe [10] is the default rate adaptation mechanism in all wireless cards based on Atheros chipsets. It is a credit-based strategy in that it maintains credits for the currently used rate on a per-destination basis to aid in the decision to increase the data rate. Initially, the rate is set to 24Mbps for 802.11a/g, and 11Mbps for 802.11b, with zero credits for that rate. Subsequently, Onoe decides on the rate and updates credits periodically every observation interval (default one second) based on success of frame transmissions and retransmission count during the previous observation interval. It steps down to a next lowest rate if either none of the transmissions were successful in the previous interval, or more than ten frames were transmitted with average retries exceeding one. Credit count is decremented if more than 10% of the frames retried during the previous observation interval, and incremented otherwise. If the credit count reaches a threshold (10) then Onoe shifts to a next higher rate. Clearly, Onoe is conservative in moving to higher rates — it takes at least ten observation intervals for a rate increase, whereas rate decrease can happen in just one interval.

B. SampleRate

This mechanism lays special emphasis on lossy links (typical in outdoor environment). The design of SampleRate [4], [2] is based on the following insight: for lossy links, using higher data rates with higher loss rates can result in higher

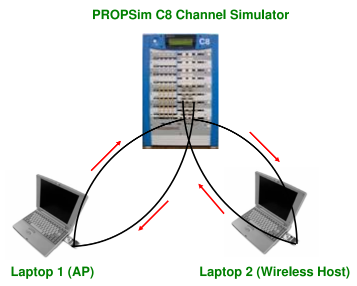


Fig. 1. A schematic of our testbed. The black lines represent RF cables, whereas the red arrows show signal flow.

raw link layer throughput; in other cases, highest data rate with low loss rate gives highest throughput. SampleRate works as follows. It selects the data rate on a per-frame basis. Initially, it uses the highest possible rate (11Mbps in 802.11b and 54Mbps in 802.11a/g). Afterwards, the *eligible* rate with smallest estimated average frame transmission time (including loss recovery) is selected. A data rate is eligible if successful transmission time without retry using that rate is smaller than average transmission time of current rate and use of that rate did not result in four consecutive transmission failures. To estimate the frame transmission times at eligible rates other than the current rate, every tenth frame is sent using a data rate randomly chosen from the set of eligible rates excluding the current rate.

III. Experimental Setup

Fig. 1 illustrates our testbed setup. We use the PROPSim C8 wideband multichannel simulator [8] for fine-grained wireless channel emulation. By default, this channel emulator supports only one-way channels. Since 802.11 MAC requires bidirectional communication (for exchanging DATA and ACK frames), we enable such two-way communication using a pair of one-way channels provided by the emulator and a combination of two-way splitters, isolators and different types of connecters. Two laptops (Dell Latitude D600 model) form the end hosts for the wireless link in our testbed. Both laptops are equipped with a commodity 802.11 wireless PC cards having an external antenna port (Proxim ORiNOCO Gold 802.11b/g) in order to connect to the channel emulator using RF cables. To minimize leakage, we shield the cards by wrapping them with copper foil.

Proxim cards we used are based on Atheros chipsets for which open-source linux drivers are available. In particular, we use the widely used Multi-band Atheros Driver for WiFi (MADWiFi) [10], which already includes implementations for Onoe and SampleRate rate adaptation mechanisms. Laptops in our testbed run Fedora 2.6.10 kernel. We configured the laptops in 802.11 infrastructure mode such that one of them acts an access point (AP) and the other as the wireless host (see Fig. 1). We use 802.11b in all our experiments with RTS/CTS disabled. We use the default MAC retry limit (4) with same rate across all retransmissions.

We use a subset of TGn channel models [9] in our experiments to represent diverse environments typical of 802.11

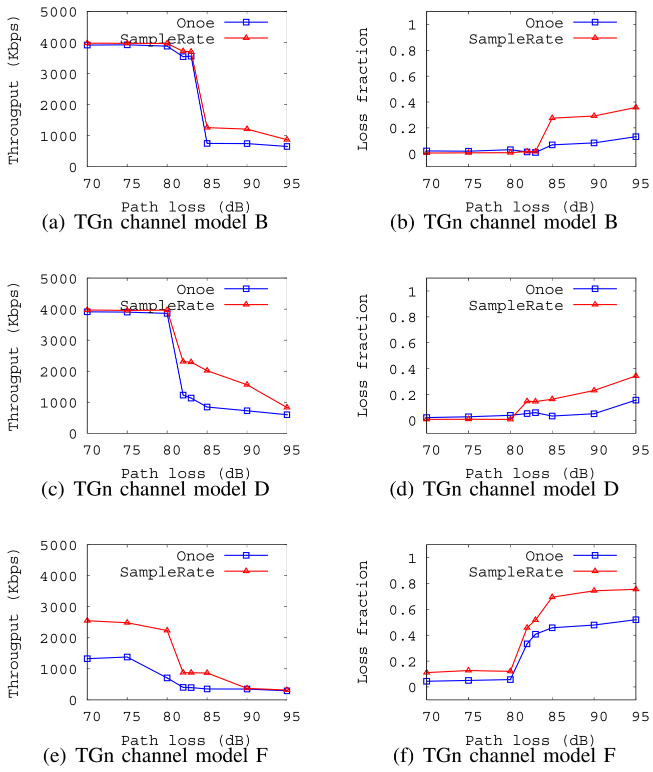


Fig. 2. Average throughput and packet loss rate for CBR/UDP traffic with Onoe and SampleRate across three TGn channel models (B, D and F).

deployments. Specifically, we consider the following three models: (i) Model B with 15ns rms delay spread — residential home or small office environment; (ii) Model D with 50ns rms delay spread — a typical office environment, non-line-of-sight (NLOS) conditions; (iii) Model F with 150ns rms delay spread — a large open space (indoor and outdoor) environment, NLOS conditions. For each of the above models, we further vary the path loss value in our evaluations (using a parameter in the channel emulator) to create a wide range of channel conditions. Throughout we use a fixed transmit power (10dBm) at the sender.

IV. Results

A. CBR/UDP Traffic

We begin our study of the interaction between applications and rate adaptation mechanisms with UDP performance of Onoe and SampleRate in different environments. The UDP workload is useful in several respects. As already mentioned, most previously reported results are based on UDP traffic. So using an identical type of workload allows us to relate results from our testbed with those in the literature. Using UDP traffic also eases the analysis of the relationship between throughput and other network-centric metrics (e.g., loss rate) of importance to applications because it does not entail application or transport layer adaptation. Finally, it is indicative of the non-adaptive multimedia workloads.

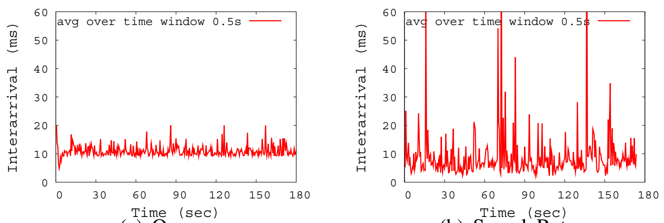
We generate UDP traffic from the AP to the wireless host in our testbed (see Fig. 1) using the Multi-Generator (MGEN) [17], a well-known open-source software from the PROTEAN research group at NRL. Besides traffic generation, MGEN also includes a tool called DREC for logging and analyzing statistics at the receiver. We examined four metrics: throughput, loss rate, packet latency and packet inter-arrival time (jitter). We observed latency performance to be similar to that of throughput in all our experiments. So we do not show latency results in the interest of space. We experimented with a wide range of traffic loads (between 1Mbps and 7Mbps) by varying the packet generation rate in MGEN while keeping the packet size fixed (1000 bytes). Since the performance behaviors we report are seen across all loads, here we present results only for one traffic load (4Mbps) for brevity.

Fig. 2 shows the relative performance of Onoe and SampleRate in terms of average throughput and packet loss rate for three different TGn channel models (B, D, F). For each channel model, we exercise different rate adaptation mechanisms over a wide range of channel conditions by varying the path loss value. Unless otherwise mentioned, each data point in the paper corresponds to an average value taken over a 180 second period. From Fig. 2, it can be clearly seen that the channel model (environment) heavily influences the region and extent of performance differentials between the two mechanisms.

SampleRate consistently outperforms Onoe in terms of throughput (Fig. 2(a),(c),(e)) with progressively greater improvements as we go from channel model B to D to F; relative performance of SampleRate over Onoe also gets better as we transition from low to intermediate path loss values. The throughput differences between Onoe and SampleRate can be attributed to their design differences. Recall from Section II.A that Onoe uses a conservative credit-based strategy to increase the data rate — sufficient number of credits (10) must be accumulated before a higher rate is tried, which can happen only when less than 10% frames are retried in ten successive observation intervals (each of one second duration). So Onoe is much more likely to stay at the current rate even if it experiences a small percentage of frame losses. SampleRate, on the other hand, has a much weaker constraint on frame losses to use higher rates — a higher rate is used provided that rate has a smaller average frame (re-)transmission time than the current rate, which is gleaned from continuous probing at promising alternate rates.

Relative performance of Onoe and SampleRate with respect to loss rate is exactly opposite from that of throughput (Fig. 2(b),(d),(f)) — Onoe always has a lower loss rate in all channel models with greater reductions seen as path loss value increases. This behavior can be explained using similar explanation as above. In particular, Onoe's conservative use of lower rates relatively improves its ability to provide higher reliability of frame transmissions, hence fewer frame losses that go unrecovered by the MAC ARQ mechanism.

Onoe exhibits much better jitter performance compared to SampleRate (Fig. 3). In particular, the packet inter-arrival time with Onoe varies over a small range [5, 20]ms, whereas it



(a) Onoe (b) SampleRate

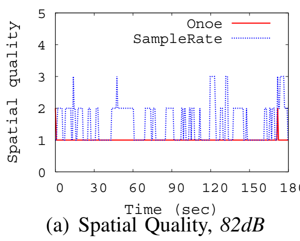
Fig. 3. Packet inter-arrival time (jitter) for CBR/UDP traffic with Onoe and SampleRate (TGn channel model D, *85dB* path loss).

goes up to 60ms with SampleRate. Onoe remains steady at a rate that provides a high probability of successful frame transmission. SampleRate, on the other hand, is aggressive in trying higher rates and potentially risks frame losses in the process. This in turn triggers the 802.11 MAC ARQ mechanism (backoff and retransmission). The exponential latencies due to the backoff strategy explain the frequent large spikes seen in the packet inter-arrival time with SampleRate.

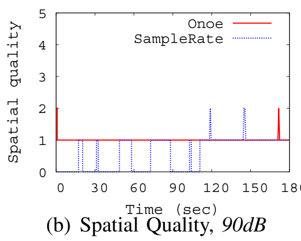
Reversals in relative performance between different rate adaptation mechanisms (Onoe and SampleRate in our case) depending on the choice of metric have several important implications for application-level performance. Generally speaking, different applications can be viewed as maximizing different utility functions, which in turn are defined based on a set of metrics such as throughput and loss rate. From this viewpoint, the impact of performance differences between rate adaptation mechanisms on application performance may be different for different applications. Even for the same application, the impact can be very different depending on the channel characteristics and sensitivity of the application to different metrics. For instance, the performance of loss intolerant applications such as FTP may get severely hurt as packet loss rate increases even though it may accompany raw link layer throughput improvement. Even loss tolerant multimedia applications may be very sensitive to the random loss of certain packets (e.g., I frames in MPEG video streams). This issue will be addressed in detail in the context of specific applications next.

B. *Adaptive Video Streaming*

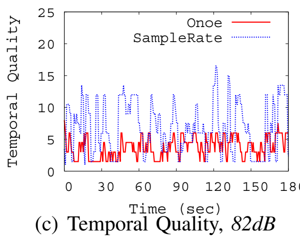
Having looked at traffic that reflects non-adaptive streaming of multimedia in the previous subsection, we now move to adaptive streaming media applications that adapt media quality in response to changing network conditions. In best-effort service networks like the Internet, adaptive media streaming when running on top of TCP-friendly congestion control protocols allows for efficient sharing of network resources (e.g., bandwidth) while gracefully adapting media quality. Recent characterization studies of streaming video traffic [18] find that currently the bulk of the streaming traffic uses TCP as opposed to UDP. Based on this observation, our study uses a TCP-based, publicly available adaptive video streaming tool called QStream [14], [19]. Briefly, QStream uses a scalable video compression format called SPEG (an easier to implement variant of MPEG-4 FGS) and follows a priority drop strategy



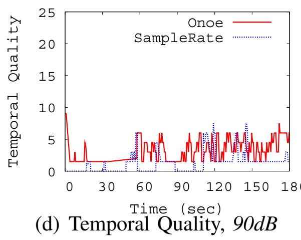
(a) Spatial Quality, *82dB*



(b) Spatial Quality, *90dB*



(c) Temporal Quality, *82dB*



(d) Temporal Quality, *90dB*

Fig. 4. Performance of QStream (adaptive video streaming tool) with Onoe and SampleRate (TGn channel model D, *82dB* and *90dB* path loss, respectively).

for application-level adaptation. In particular, it prioritizes video data units based on their relative importance and drops low priority units right at the sender when available network bandwidth falls (as reported by the underlying congestion control protocol, TCP in this case)

QStream supports two measures for assessing video quality corresponding to spatial and temporal dimensions. Spatial quality represents the spatial resolution of a video frame; SPEG codec (used by QStream) provides four spatial quality levels. Temporal quality is the frame rate in terms of the number of frames displayed per second (fps).

We studied QStream performance with Onoe and SampleRate across different channel models and a wide range of path loss values for each model. However due to space constraints, we present only a small subset of that data to illustrate our main point that neither rate adaptation mechanism outperforms the other for all channel models and path loss values. Fig. 4 shows the QStream performance observed during a three minute period (when a stored video is streamed between the two laptops in our testbed) for two path loss values (82dB and 90dB, respectively) using TGn channel model D.

With a low path loss value (82dB), SampleRate exhibits superior average and peak performance over Onoe across both measures (Fig. 4(a),(c)). We notice a reversal in relative performance when the channel quality gets worse with increase in path loss value to 90dB (Fig. 4(b),(d)); this is evident from the repeated pauses seen with SampleRate across both measures. We also observe that SampleRate performance shows wide variability in cases where it performs well, whereas Onoe has relatively steady albeit poor performance (Fig. 4(a),(c)). The above differences in performance behaviors with SampleRate and Onoe can be traced back to their differences in throughput, loss rate and jitter performance seen earlier with UDP traffic, and the effect of loss on the TCP sending rate. Further, the poor performance with both mechanisms (although at different

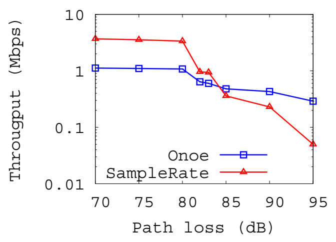


Fig. 5. Throughput for file transfer traffic with Onoe and SampleRate (TGn channel model D).

operating regions) highlights their ineffectiveness in providing good overall performance; this is rooted in their inability to tune the adaptation strategy in response to the channel quality *and* application characteristics.

The above performance issue can be addressed with a combination of techniques at different layers (e.g., unequal error protection at all layers; use of FEC at application/link layers; loss differentiation at transport layer; use of incremental redundancy, and judicious choice of frame sizes and retransmission limits at the link layer) along with inter-layer awareness. An alternative and potentially more promising approach would be to more tightly integrate all layers and jointly optimize them toward best application performance. Thorough comparison of these different approaches is left for future work.

C. File Transfer Application

In this subsection, we focus on the more traditional application of file transfer; it also shares the characteristics of file downloads in emerging P2P applications. A key characteristic of this application is its zero tolerance to packet losses, which necessitates the use of a reliable transport protocol like TCP. The application layer throughput is the primary measure of performance for file transfer. We use the well-known Netperf tool [20] for generating a large TCP workload to reflect file transfer traffic.

Fig. 5 shows the file transfer performance (throughput) with Onoe and SampleRate for TGn channel model D for a wide range of path loss values. We can observe that SampleRate performs much better than Onoe when the channel quality is good (i.e., low path loss values) while it is the opposite at higher path loss values. For instance, *the throughput with SampleRate is a factor of five greater than Onoe at 70dB path loss, whereas it is one-tenth that of Onoe for 95dB path loss.* This is interesting given that the main motivation behind SampleRate design is to improve throughput under lossy conditions [4]. As noted before, the higher loss rate with SampleRate at high path loss values (as clearly seen with UDP traffic) has a dominating impact on the sending rate of TCP, hence the poor application throughput. The results in this section reiterate the need for application-aware PHY rate adaptation.

D. Web Traffic

Finally we consider web (HTTP) traffic which dominates the traffic both in the Internet as well as 802.11 networks [15].

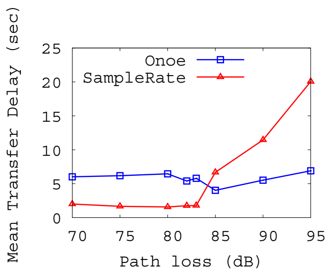


Fig. 6. Mean transfer delay for web traffic with Onoe and SampleRate (TGn channel model D).

Even though web traffic is also based on TCP like file transfer, it has two unique characteristics that make it quite different. First, it is characterized by short-lived flows. Second, it is more interactive in nature. The second characteristic directly relates to the key performance measure of interest (from a user viewpoint) when evaluating web traffic, i.e., the time it takes to complete the transfer of an object from the point a request is made (transfer delay). We generate the web traffic using the SURGE tool [21], which is based on data collected from empirical measurements. Specifically, we setup a web server on the AP in our testbed and have the other wireless host act as a user making web requests (of varying file sizes). We use the default parameter settings of SURGE with HTTP/1.1. As seen from Fig. 6, the web traffic performance (mean transfer delay) exhibits similar relative performance behaviors between Onoe and SampleRate as before with similar underlying causes.

Comparing results in Fig. 5 and Fig. 6 shows that it is important to consider application-level characteristics for realistic evaluations. Even though both file transfer and web traffic are based on TCP, application performance with Onoe and SampleRate in both cases is quite different. For file transfer traffic, SampleRate is 5x better than Onoe at 70dB path loss, whereas it is 10x worse than Onoe at 95dB path loss; for web traffic, on the other hand, performance differentials are around factor of three at both those path loss values.

V. Conclusions

In this paper, we have experimentally studied the interplay between the performance of real applications and the design of PHY rate adaptation mechanisms over 802.11 wireless links under different channel environments. A key aspect of our study is the use of highly realistic testbed based on a real-time hardware channel emulator. Using this testbed, we have evaluated a wide range of realistic application workloads (including adaptive video streaming and web traffic), some of them having multiple metrics, over two representative 802.11 rate adaptation mechanisms, namely Onoe and SampleRate.

Across all our workloads, we have found that neither rate adaptation mechanism consistently outperformed the other. Further, their performance can be significantly different with the relative performance dependent on the channel characteristics and the performance measure. This is in sharp contrast to prior evaluations based on raw throughput performance using backlogged UDP traffic. Fundamentally, these differences are rooted in the way different mechanisms tradeoff loss rate when optimizing link layer throughput without consideration for application characteristics. Thus, our results specifically highlight the importance of application-awareness in determining the adaptation strategy for rate selection in response to time-varying channel conditions, and more generally the need for cross-layer optimization.

Our future work will focus on extending this evaluation study to include other types of rate adaptation mechanisms, and developing more effective link adaptation mechanisms based on cross-layer awareness.

Acknowledgment

We thank Weijun Zhu, David Browne and Prof. Mike Fitz for help with the channel emulator.

References

[1] IEEE Std. 802.11, “Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,” 1999.
[2] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, “Architecture and Evaluation of an Unplanned 802.11b Mesh Network,” in *Proc. IEEE/ACM MobiCom*, 2005.
[3] M. Lacage, M. H. Manshaei, and T. Turletti, “IEEE 802.11 Rate Adaptation: A Practical Approach,” in *Proc. ACM MSWiM*, 2004.
[4] J. Bicket, “Bit-rate Selection in Wireless Networks,” Master’s thesis, MIT, Feb 2005.
[5] Z. Wu, S. Ganu, I. Seskar, and D. Raychaudhuri, “Experimental Investigation of PHY Layer Rate Control and Frequency Selection in 802.11-based Ad-Hoc Networks,” in *Proc. Sigcomm 2005 Workshop on Experimental Approaches to Wireless Network Design and Analysis (E-WIND’05)*, 2005.
[6] I. Haratcherev, J. Taal, K. Langendoen, R. Lagendijk, and H. Sips, “Automatic IEEE 802.11 Rate Control for Streaming Applications,” *Wireless Communications and Mobile Computing Journal*, June 2005.
[7] G. Judd and P. Steenkiste, “Using Emulation to Understand and Improve Wireless Networks and Applications,” in *Proc. 2nd Symposium on Networked Systems Design & Implementation (NSDI)*, 2005.
[8] PropSim, “Propsim C8 - Wideband Multichannel Simulator,” http://www.propsim.com/.
[9] IEEE 802.11 Task Group n, “Channel Models,” May 2004.
[10] “Madwifi: Multiband Atheros Driver for WiFi,” http://madwifi.sourceforge.net/.
[11] A. Kamerman and L. Monteban, “WaveLAN-II: A High-Performance Wireless LAN for the Unlicensed Band,” *Bell Labs Technical Journal*, 1997.
[12] G. Holland, N. Vaidya, and P. Bahl, “A Rate-Adaptive MAC Protocol for Multi-Hop Wireless Networks,” in *Proc. IEEE/ACM MobiCom*, 2001.
[13] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly, “Opportunistic Media Access for Multirate Ad Hoc Networks,” in *Proc. IEEE/ACM MobiCom*, 2002.
[14] C. Krasic, J. Walpole, and W. Feng, “Quality-Adaptive Media Streaming by Priority Drop,” in *Proc. ACM NOSSDAV*, 2003.
[15] T. Henderson, D. Kotz, and I. Abyzov, “The Changing Usage of a Mature Campus-Wide Wireless Network,” in *Proc. IEEE/ACM MobiCom*, 2004.
[16] S. Sen, O. Spatscheck, and D. Wang, “Accurate, Scalable In-network Identification of P2P Traffic Using Application Signatures,” in *Proc. 13th International World Wide Web (WWW) Conference*, 2004.
[17] MGEN, “The Multi-Generator Toolset,” http://mgen.pf.itd.nrl.navy.mil/.
[18] J. van der Merwe, S. Sen, and C. Kalmanek, “Streaming Video Traffic: Characterization and Network Impact,” in *Proc. 7th International Web Content Caching and Distribution Workshop*, 2002.
[19] QStream Project., “Quality Adaptive Media Streaming,” http://qstream.org.
[20] “Netperf: A Network Performance Benchmark,” http://www.netperf.org/netperf/NetperfPage.html.
[21] P. Barford and M. Crovella, “Generating Representative Web Workloads for Network and Server Performance Evaluation,” in *Proc. ACM Sigmetrics*, 1998.