Recitation: Rehearsing Wireless Packet Reception in Software

Zhenjiang Li, Yaxiong Xie, Mo Li
School of Computer Engineering
Nanyang Technological University
{lzjiang, yxie005, limo}@ntu.edu.sg

Kyle Jamieson
Department of Computer Science
University College London
k.jamieson@cs.ucl.ac.uk

ABSTRACT

This paper presents *Recitation*, the first software system that uses lightweight channel state information (CSI) to accurately predict error-prone bit positions in a packet so that applications atop the wireless physical layer may take the best action during subsequent transmissions. Our key insight is that although Wi-Fi wireless physical layer operations are complex, they are deterministic. This enables us to rehearse physical-layer operations on packet bits before they are transmitted. Based on this rehearsal, we calculate a hidden parameter in the decoding process, called error event probability (EVP). EVP captures fine-grained information about the receiver's convolutional or LDPC decoder, allowing Recitation to derive precise information about the likely fate of every bit in subsequent packets, without any wireless channel training. Recitation is the first system of its kind that is both software-implementable and compatible with the existing 802.11 architecture for both SISO and MIMO settings. We experiment with commodity Atheros 9580 Wi-Fi NICs to demonstrate Recitation's utility with three representative applications in static, mobile, and interference-dominated scenarios. We show that Recitation achieves 33.8% and 16% average throughput gains for bit-rate adaptation and partial packet recovery, respectively, and 6 dB PSNR quality improvement for unequal error protection-based video.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless Communication

Keywords

Wireless; channel state information; packet prediction; EVP; bit rate adaptation; unequal protection

1. INTRODUCTION

To achieve faster speeds in the face of increasing user demand, wireless local and wide area networks now turn to wideband transmission [44, 53]. For example, 802.11ac can now transmit over a full 160 MHz of bandwidth. As a result, different subcarriers now experience different subchannels [2, 30, 34]. Figure 1 illustrates these wide variations in (a) signal-to-noise ratio (SNR) and (b) bit error rate (BER). How do Wi-Fi and other wireless designs cope with this subcarrier diversity? The standard approach comprises two techniques, with the goal of equalizing the BER of all bits in a packet. First, the sender applies error control coding to produce a redundancy-containing stream of *coded bits* from the stream of *data bits* that make up a single packet. Then, the sender interleaves and pseudo-randomly spreads the coded bits over the different subcarriers of the wireless channel, before transmitting them over the air. We illustrate this process in Figure 2. Is this effective? Figure 1(c) shows that often, the answer is unfortunately no: note the visible difference in BER for data bits in different positions of the packet, ranging from 10^{-4} to almost 10^{-2}.

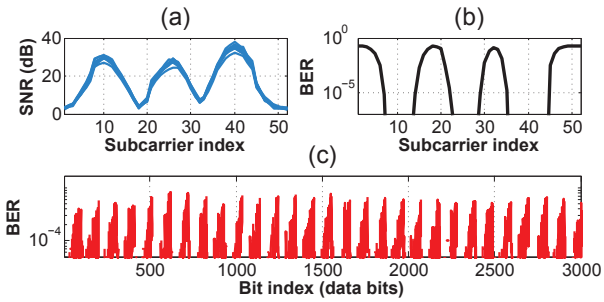

Figure 1: Data from 20 MHz 802.11n wireless channels. **(a)** Signal-to-noise ratio (SNR) across different subcarriers, measured over eight received packets; **(b)** Bit error rates (BERs) across different subcarriers; **(c)** The BER of decoded data bits, by position in the packet.

This forces existing wireless designs like Wi-Fi to take a conservative approach, applying enough coding to drive down BER across the packet so that even the highest BER is less than 10^{-6}. In this work we take a different position, exploring what we could do if we could construct a picture of a future packet like the one in Figure 1(c) simply by hearing the preamble of a recent packet. If we can construct such a picture, a number of possibilities emerge:

1. **Bit-rate adaptation.** Wireless channel quality is dynamic [8, 12, 52], and so senders using fixed bit-rates need to promptly select the rate at which they send information. Senders using rateless codes [11, 36] also benefit by learning when to stop sending rateless coded units.

2. **Partial packet recovery.** Receivers can leverage partially-received packets to improve throughput [14, 21]. More partially-received packets result when the sender selects data rates at or higher than the channel can support. Accurate per-bit BER pre-

MobiCom'15, September 7–11, 2015, Paris, France.

DOI: http://dx.doi.org/10.1145/2789168.2790126.

dictions allow the sender to decide whether the speed gains resulting from such higher data rates outweigh the consequent retransmission overheads.

3. **Unequal error protection.** Packet bits may have differing levels of importance, *e.g.*, key frames versus difference frames in video packets. Important bits are better placed at positions that are less likely to suffer from decoding errors [1, 19, 43].

In order to realize this vision, the wireless system needs to analyze each element of the physical layer design shown in Figure 2, most notably the error-controlling encoder and decoder, and the interleaving and de-interleaving. The missing piece is a way of analyzing how BER measured at each constituent subcarrier impacts the final confidence of each received data bit.

This paper presents *Recitation*, the first system that uses the channel state information (CSI) present in a packet's preamble as input, and based on this information alone, makes highly accurate packet error rate predictions so that a sender may select the best bit rate before the next transmission (Applications 1 and 2). Solely based on CSI data in the header, Recitation can also identify the error-prone bit positions in a packet so that the sender may rearrange packet bits based on application-level importance (Application 3).

To accomplish this, Recitation directly tackles the heretofore unclear relationship between diverse subcarrier SNRs and 802.11 decoding failures. Our key insight is that while the wireless physical layer is highly complex, it is (or can be designed to be) deterministic. This enables us to "rehearse" every operation on future received bits, for more accurate results than previous approaches such as Effective SNR (ESNR) [12] and SoftRate [52], as explained below in §6. We calculate the BER for each coded bit at point (a) in Figure 2 and carefully replay the operation of the decoder. Based on this "rehearsal" of the decoder, we calculate a quantity we term *error event probability* (*EVP*), a hidden parameter in the decoding process. EVP captures the probability of the decoder making a mistake at each step of its operation, thus giving extremely fine-grained decoding information. Based on EVP we then derive confidences of each data bit's correctness at point (b) in Figure 2, without requiring any wireless channel training. We demonstrate this approach in both 802.11's standard convolutional codes, as well as cutting-edge LDPC codes, establishing the generality of our approach.

Contributions. This paper makes following contributions. First, we formulate a novel metric that explicitly and precisely takes error control coding into account in its estimation of which packet data bits are likely to be correct (EVP). Second, we describe the design and implementation of Recitation, a system that integrates EVP with unequal error protection, partial packet recovery, and bit rate adaptation schemes. Third, we implement Recitation on the commodity Atheros chipset and present a comprehensive, testbed-based experimental evaluation that shows EVP's utility in both static and mobile scenarios with background traffic and interference.

Limitations. Recitation accurately calculates per-bit confidences of future packet receptions based on past receptions, and is thus most effective in the case of a channel where packet collisions are not the dominant factor limiting throughput, but in §4 we show promising throughput gains both in networks with typical levels of background traffic, and in interference-dominated networks.

We implement Recitation on commodity Atheros 9580 Wi-Fi network interface cards (NICs), and present experiments that validate the efficacy of the Recitation design. We demonstrate Recitation's utility in the three above applications. For each application, we compare Recitation with state-of-the-art approaches using a trace-driven GNU Radio evaluation. Recitation achieves significant performance improvements: a 33.8% average throughput gain over ESNR [12] for bit-rate selection, a 16% average throughput gain over Maranello [14] for partial packet recovery, and 6 dB PSNR video quality improvement over standard approaches for unequal error protection. We also evaluate the amount of computation Recitation requires to show that that it is practical. §4 concludes that Recitation is software-implementable and compatible with the existing 802.11 multiple-input and multiple-output (MIMO) physical layer without hardware modification.

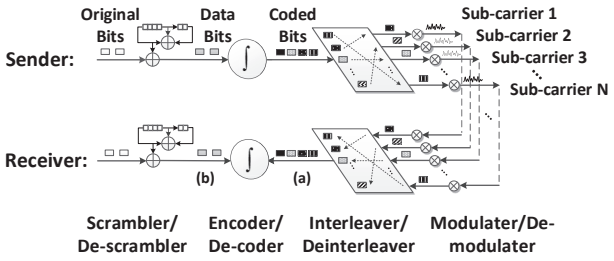

Figure 2: Scrambling, error control coding, spreading, and interleaving in the standard Wi-Fi architecture.

Roadmap. The rest of this paper is structured as follows: §2 presents our design, §3 our implementation, and §4 an experimental evaluation. §5 discusses future extensions to Recitation. §6 surveys related work before §7 concludes.

2. DESIGN

This section outlines the design of Recitation, starting with a primer on the relevant parts of the Wi-Fi physical layer in §2.1. Next we present the design of the EVP metric in Section 2.2, first in principle (§2.2.1), then in the context of Wi-Fi (§2.2.2), and then conclude with refinements that make its computation tractable in practice (§2.2.3).

2.1 Primer: Controlling errors in Wi-Fi

Figure 2 illustrates the 802.11 architecture [17] at a high level. At the sender, a scrambler deterministically whitens the original data bits. An encoder then maps the data bits into a larger number of partly-redundant coded bits. Orthogonal Frequency Division Multiplexing (OFDM) [26] divides the channel bandwidth of 20 or 40 MHz into 64 or 128 312.5 KHz *subcarriers*. The sender then interleaves the coded bits before mapping them to subcarriers and *modulating* them into *symbols*. Each subcarrier conveys independent symbols simultaneously on different frequencies. Together, the combined signal is called an *OFDM symbol*. At the receiver, each received OFDM symbol undergoes the complementary series of operations to recover the original data bits.

2.1.1 Scrambling, interleaving, and modulation

The scrambler performs bit-wise XORs between a pseudo random sequence and the original data bits. At the receiver, the original data bits are recovered by the same operation. As the random number sequence is specified in 802.11 standards [17], knowing the scrambled data bits is tantamount to knowing the original data bits. In the rest of the paper, we focus on the scrambled data bits and refer them to data bits for short, unless explicitly stated otherwise.

After encoding (discussed separately below in §2.1.2), the sender *interleaves* coded bits within each block of B bits in one OFDM symbol, to avoid consecutive coded bits being mapped onto faded subcarriers. The block interleaver uses two known and pseudo-random mappings, also specified by the 802.11 standard. After interleaving, the sender maps the data bits onto OFDM subcarri-

ers for transmission. Due to interleaving, long runs of consecutive coded bit errors are largely avoided. The interleaving, however, only re-allocates the positions of coded bits (their BERs are still diverse). The decoded data bit BERs thus remain diverse, as we can see from Figure 1(c). The periodical BER pattern across data bits is because interleaving is performed in the unit of OFDM symbols, *i.e.*, the operations within each OFDM symbol is the same and these operations are repeated over different symbols.

Although interleaving and modulation are complex, their operations are deterministic. We thus know the subcarrier over which each coded bit travels. Furthermore, subcarriers are narrow-band in 802.11, and so their SNRs can be computed from the CSI. We thus employ narrow-band SNR-BER relationships to calculate the BER of each coded bit (point (a) in Figure 2).

2.1.2 Encoding and decoding

Encoding. *Convolutional coding* is the default error control scheme in 802.11a/g/n/ac. Encoding can be seen as transitions of a state machine between a number of different states. At each step, the encoder inputs one data bit, triggers a state transition, and outputs two coded bits.[1] To illustrate the concepts, we consider in this section the simple ½-rate convolutional code with four states in Figure 3. At state a, if the input packet bit is "0", the state machine stays at this state after the i^{th} state transition and outputs two coded bits "00". It will transition to state c and output coded bits "11" when the input packet bit is "1". Suppose the incoming data bits were all zeros: then the encoder will remain at state a and all the output coded bits would be zero, corresponding to PATH-1 in Figure 3.

Low-Density Parity Check (LDPC) codes are recently appearing in 802.11n/ac [17]. They are block codes where the encoder divides data bits into k-bit data blocks, encoding each data block into n-bits by a matrix computation [31, 41]. The ratio between k and n, *i.e.*, k/n, defines the code rate.

Decoding. At the receiver, the received coded bits may contain errors. The decoder aims to find the most likely decoding result according to different criteria. The convolutional decoder approximates an optimal decoding by computing the number of differing bits, or *Hamming distance*, between the coded bits and the received bits, choosing the path that minimizes this Hamming distance. The best LDPC decoders typically adopt a variety of *belief propagation* (BP), which propagates information in a graph structure containing parity checks, stopping decoding when the propagated information (*beliefs*) satisfy all parity checks, or after a predefined maximum number of iterations.

2.2 Error event probability

A packet decoding fails exactly when excessive errors occur in the received coded bits, *i.e.*, the convolutional decoder selects an incorrect path (*e.g.* PATH-2 in Figure 3) or the LDPC decoder experiences a number of errors exceeding its error correction capability (ECC) in any coded block.[2]

In this event (an *error event*) the decoded data bits also contain errors. We denote the probability that an error event occurs the *error event probability* or EVP in this work. More precisely, EVP_i describes the probability that any faulty path diverges from the correct path in the i^{th} state transition for convolutional codes, which captures the likelihood of a decoding error at data bit i. For LDPC codes, EVP_i describes the probability that an error event occurs within data block i.

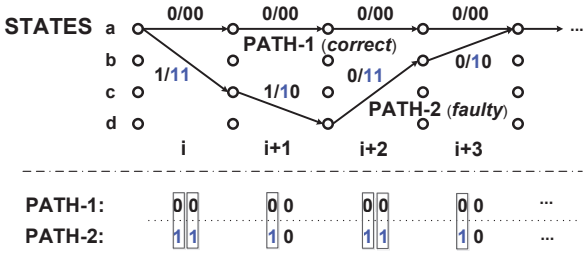

Figure 3: Convolutional coding. 1/11 along a branch means data bit "1" is decoded from the two coded bits "11".

2.2.1 EVP calculation in principle

We now explain how to compute EVP for convolutional and LDPC codes, showing that EVP can accurately predict packet error rate (PER) and bit error locations.

Convolutional codes. An incorrect path such as PATH-2 diverges from the correct path PATH-1 at some data bit i, as shown in Figure 3.[3] Since PATH-1 and PATH-2 differ by 6 coded bits, if more than three ($= \lfloor (6-1)/2 \rfloor + 1$) errors occur in the coded bit stream for PATH-1, these errors will result in an incorrect PATH-2 decoding. Therefore, in this example, the probability that the received sequence is decoded as PATH-2 can be calculated by the summation of the probabilities for all $\sum_{k=4}^{6} \binom{6}{k}$ cases. The probability of each case can be further calculated using the coded bit BERs. For instance, the probability in our example that only the first four highlighted bit positions are in error is $\prod_{j\in\mathcal{F}} p_j \cdot \prod_{j\in\overline{\mathcal{F}}}(1-p_j)$, where p_j is the BER of coded bit j, $\mathcal{F} = \{1,2,3,5\}$, and $\overline{\mathcal{F}} = \{6,7\}$.

PATH-2 is of course just one of many incorrect paths, so EVP_i is the probability that *any* path diverges from the correct path at packet bit i:

$$\text{EVP}_i \quad = \quad \sum\nolimits_{j\in\mathcal{P}_i} P_{\text{PATH-j}}, \tag{1}$$

where $\mathcal{P}_i$ is the set of all faulty paths diverging at packet bit i, and $P_{\text{PATH-j}}$ is the probability that the received coded bits are decoded as PATH-j.[4] $\mathcal{P}_i$ contains all possible faulty paths with different Hamming distances and thus grows exponentially in path length. Therefore we need a way of bounding the number of paths Recitation searches.

Faulty paths have *diverging segments* to the correct path, *e.g.*, from state transitions i to $i+3$ on PATH-2. The longer diverging segment a faulty path has, the more coded bit differences to the correct path, *i.e.*, a larger Hamming distance to the correct path, it tends to have. Error events caused by a faulty path with a longer Hamming distance are thus more unlikely to occur, because a longer Hamming distance requires more coded bit errors to occur together. In a classic paper [51], Viterbi *et al.* analytically derive the probability that an error event is caused by any faulty path with Hamming distance k for our example code of Figure 3.

[1] Other code rates are achieved using the puncturing technique [17] based on the ½ rate.

[2] Since convolutional codes are not block-based, instead processing data bits as a bit stream, the number of coded bit errors they can tolerate depends on the density of those errors [51]. Therefore, in contrast with LDPC codes, convolutional codes have no fixed ECC value, and their decoding performance must be explicitly analyzed.

[3] There may exist other paths that diverge from the correct path in the i^{th} state transition as well, which will also cause error events.

[4] Although we assume the transmitted coded bits are all zeros in Figure 3, the EVP definition in Eq. (1) can generalize to any coded bit sequence by symmetry, as when we focus on the difference between the transmitted and received coded bit sequences for PATH-1 and PATH-2 each "1" bit is an error, but this choice is arbitrary.

Their result shows that 98% of the error events are due to faulty paths with Hamming distance less than eight, suggesting considering only faulty paths whose Hamming distances are less than eight in the EVP calculation.

However, two difficulties arise that preclude a direct application of this analysis: first, the 802.11 specification convolutional code has 64 states instead of the four in Figure 3. Second, the analysis of Viterbi *et al.* assumes uniform coded bit BERs, which is not true in real wireless channels. Below in §2.2.2 we develop empirical approximations that reduce the amount of calculation required to practical levels while adopting the basic principle described above.

Before detailing this empirical calculation, we remind the reader of the two major uses of EVP:

1. Given the EVPs of each packet bit, the packet error rate (PER) can be estimated as $1-\prod_{i=1}^{N}(1-\text{EVP}_i)$, where N is the number of packet bits in a packet.
2. When the wireless channel is wideband, EVP varies within the packet, as we show in §4.1.3, and can identify the error-prone bit positions in the packet.

LDPC codes. Since LDPC codes are block codes, instead of analyzing the probability that each individual error event would occur in Eq. (1), we can rely on ECC to directly calculate EVP_i as:

$$\begin{aligned}\text{EVP}_i &= \text{Prob}\{\text{e}_i > \text{ECC}\},\\ &= \sum_{l=\text{EEC}}^{n_i}\sum_{\mathcal{F}_l\in\mathbf{F}_l}\prod_{j\in\mathcal{F}_l}p_j\cdot\prod_{j\in\overline{\mathcal{F}}_l}(1-p_j),\end{aligned}\quad(2)$$

where e_i is the number of bit errors in coded block i, n_i is the number of bits in coded block i, p_j is the BER of each coded bit j, $\mathcal{F}_l$ is a set of l coded bit errors, $\overline{\mathcal{F}}_l$ is $\mathcal{F}_l$'s complement, and $\mathbf{F}_l$ is the set of all possible $\mathcal{F}_l$s. The ECC can be calculated either theoretically or empirically: we take the latter approach for 802.11 LDPC codes in §4.1.2.[5]

2.2.2 EVP calculation in 802.11

In this section, we describe a practical algorithm to implement the EVP calculation for convolutional code. Our experiments guide the design of an algorithm to prune the search space of possible paths in Eq. (1) for the 802.11 convolutional code. We collect 10,000 real-world CSI measurements from Atheros 9580 NICs and feed them to an 802.11n specification-compliant transceiver on the GNU Radio, which features coding rates from 1/2 to 5/6 and modulations from BPSK to 64-QAM. Each CSI measurement reflects an instantaneous measurement of the wireless channel. The sender adapts the data rate with PER ≤ 20% to transmit packets (which may succeed or fail) over each channel realization. We record all data bits and coded bits of each 1,000-byte packet for ground truth, and all received coded bits and decoded data bits.

Error burst length. We first examine the distribution of the diverging segment lengths of faulty paths from the experiment. In fact, this length cannot be directly measured. For example, in Figure 3, the diverging segment length of PATH-2 is 8 coded bits. The length is known only when we can explicitly identify this path. To address this issue, we introduce the concept of *error burst length* to approximate the diverging segment length of a faulty path. We define an *error burst* as a sequence of consecutive coded bits, where the sequence starts and ends both from one coded bit error. In addition, the distance from any other coded bit error outside of the sequence to either the beginning or the end of the sequence should be sufficiently large, *e.g.*, 10 bits. For PATH-2 in Figure 3, the error burst length is seven (from the first coded bit error to the last one). The error burst length is always less or equal to the diverging segment length of a faulty path.

[5]Prior work [9] theoretically derives ECC for a family of simplified regular codes, but for irregular codes in general (*e.g.* the LDPC codes in 802.11), closed-form solutions are unknown.

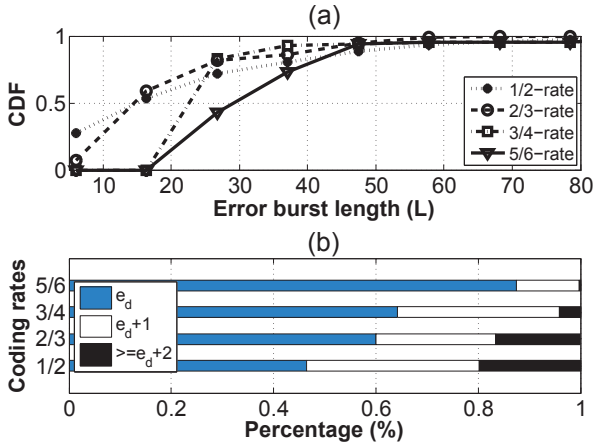

Figure 4: (a) CDF of the error burst length L; **(b)** Distribution of the number of coded bit errors, e, relative to the min. number of errors needed to cause a data bit error e_d.

For each received corrupted packet, we identify the error burst and feed the underlying coded bit error burst to the convolutional decoder to generate the decoded data bit errors. We then remove the data bit errors from the packet and continue searching for error bursts. Figure 4(a) depicts the CDF of the error burst lengths L. In general, L rarely exceeds about 40, and so we conclude that most faulty paths in 802.11 also have short diverging segment lengths. In other words, they have just a few bits differing with the correct path, *i.e.*, low Hamming distances, which is consistent with the conclusion made in §2.2.1 using the simple code.

Number of coded bit errors. A short Hamming distance k also implies that the number of errors in the error burst e is also small, since $\lfloor(k-1)/2\rfloor+1 \le e \le k$. In Figure 4(b), we further investigate the number of coded bit errors contained in error events under different code rates in 802.11. For different code rates, the minimum number of coded bit errors to cause an error event is different, denoted as e_d (although the EEC value for each specific convolutional code is not fixed, this value has a lower bound, represented by e_d [51]), where e_d ranges from five down to two for the 802.11 code rates from 1/2 to 5/6, respectively. Figure 4(b) shows that error events usually contain e_d or e_d+1 coded bit errors for all the code rates. In particular, e_d and e_d+1 together account for more than 80% of the error events for all four code rates.

Our conclusion is that we only need consider relatively few coded bit errors e that form relatively short bursts of length l in the EVP calculation. Based on the data, we consider only $e = e_d$ and e_d+1, and $l \le L = 40$, 32, 30, and 43 for the code rates from 1/2 to 5/6, capturing around 80$^{\text{th}}$ percentile burst length for each respective code rate. Therefore, these two parameters determined from Figure 4 can help eliminate unlikely cases from the EVP calculation. We will show in §4 that this error burst length reduction can make the computation in Recitation tractable on commodity devices, and achieves substantial performance gains.

Handling different bit error patterns. So far, one more issue remains. In Figure 3, to calculate the probability that the received sequence is decoded as PATH-2, we enumerate all possible cases that result in PATH-2. The reason we can do so is because we know

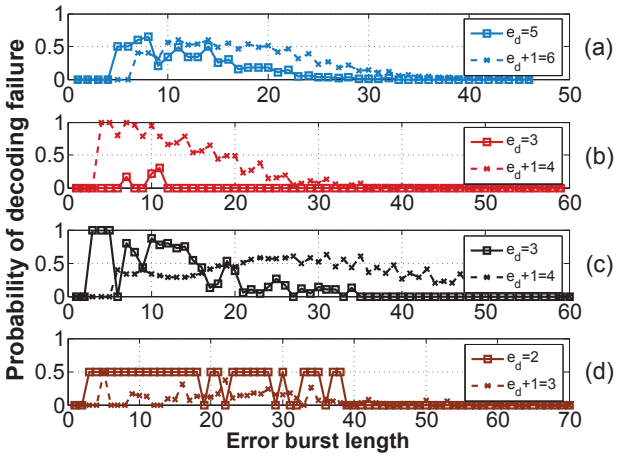

Figure 5: Weights of error combinations with different error numbers and burst lengths for **(a)** 1/2, **(b)** 2/3, **(c)** 3/4, **(d)** 5/6 code rates.

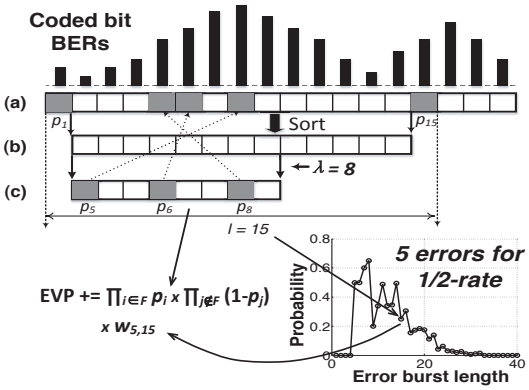

Figure 6: Recitation's error combination pruning technique, where coded bit errors occur at positions $\mathcal{F} = \{1, 5, 6, 8, 15\}$.

which coded bits on PATH-2 are different from the correct path. In general, when e errors form a l-bit burst, there are $\binom{l-2}{e-2}$ possible error combinations (a burst starts and ends from one coded bit error). For each combination, we need to tell whether it can cause a decoding failure even when we do not know which coded bits the faulty path are different from the correct path. To address this issue, we test all possible error combinations off-line in Figure 5. For each coding rate, Figure 5 provides the probability that an error combination, with e coded bit errors forming a l-bit burst, will cause a decoding failure. When $e = e_d + 1$, we eliminate the redundant combinations that are already calculated when $e = e_d$. We view Figure 5 as a look-up table for Recitation's EVP calculation. We note that the computation of the data in Figure 5 is just an one-time effort, which we have completed offline.

Recitation's EVP calculation. For each data bit i, we put the above ideas together in the following algorithm:

> **Step 1**. Use CSI to compute the BER of each coded bit, p_j and initialize $\text{EVP}_i \leftarrow 0$
> **Step 2**. For each error combination $\mathcal{F}$, update $\text{EVP}_i \leftarrow \text{EVP}_i + w_{e,l} \cdot \prod_{j\in\mathcal{F}} p_j \cdot \prod_{j\notin\mathcal{F}}(1-p_j)$
> **Step 3**. Return EVP_i

EVP_i is initialized as zero and the algorithm updates it when enumerating all possible error combinations, e.g., e coded bit errors form an error burst and the burst length l is no more than L. In Step 2, p_j is the BER of the coded bit j derived from CSI. Each error combination $\mathcal{F}$ indicates the positions of the coded bit errors. The term $\prod_{j\in\mathcal{F}} p_j \cdot \prod_{j\notin\mathcal{F}}(1-p_j)$ gives the probability that this error combination could occur, and the weight $w_{e,l}$ is the probability that this error combination will cause a decoding failure, which is obtained from Figure 5. Recitation then iterates the above EVP calculation for all data bits. As mentioned in §2.2.1, these calculated EVPs can be used to derive the packet error rate (PER) and indicate the error-prone bit positions in the packet.

In 802.11 MIMO transmissions, the sender encodes data bits to coded bits and then distributes them deterministically across different spatial streams. Our design Recitation is thus applicable to these transmission modes as well, with a similar EVP calculation.

Recitation's protocol summary. In Recitation, a sender uses the CSI from the previous transmission to predict the future packet reception under different configurations and then select the best action for the subsequent transmission. The CSI information is obtained by explicit CSI feedback, which is lightweight [12, 38].

2.2.3 *Reducing computational complexity*

The delay with which the sender can compute and act on the EVP metric gates Recitation's efficacy. For example, if we want to select a data rate for a packet using the PER predicted, the EVP computation has to be completed before that packet's transmission. Our experimental investigation with Atheros AR9580 NICs reports typical 2 *ms* delays for processing a packet of the maximum payload under the 802.11n data rates (*i.e.*, from 6.5 Mbit/s to 65 Mbit/s). It introduces a stringent timing requirement for the EVP computation. Recitation therefore employs two additional techniques to reduce computation complexity of the EVP calculation. We evaluate the algorithm in §2.2.2 in conjunction with the techniques proposed in this section in §4.

Error combination pruning. SNRs and thus the coded bit BERs can be highly diverse for wide-band transmissions (as shown in Figure 7), which makes the probability that each error combination occurs dramatically different. According to our experiments, such a difference can be as high as 10 orders of magnitude. Based on this observation, we only consider the combinations that consist of high coded bit BERs in the EVP calculation. We sort $l-2$ coded bits in the burst (except the first and last bits) according to their coded bit BERs in a decreasing order and only examine the first $\binom{e-2}{\lambda}$ combinations, where $\lambda \le l-1$. Figure 6 shows an example of the error combination pruning with $l = 15$ and $e = 5$. The five marked squares in line (a) of Figure 6 represent one error combination. Line (b) of Figure 6 depicts the sorted sequence (except the first and the last coded bits) in decreasing order according to their coded bit BERs. With $\lambda = 8$ and $e = 5$, we only consider $\binom{8}{3} = 56$ error combinations, instead of $\binom{17}{3} = 680$.

Periodicity of the EVP. The second technique we adopt is to leverage the periodic property of the EVP. We find that the EVPs of exhibit a period length related to the number of coded bits contained in one OFDM symbol. The periodic property exists because the interleaving and modulation operations are deterministic and the wireless channel is stationary for the duration of one frame at normal walking or even driving speeds [50]. The coded bits at the same position of their own OFDM symbols are transmitted over the same subcarrier, experiencing the same channel quality. The BERs of the coded bits are thus periodic, leading to the EVPs of the decoded packet bits having a periodical property. We experimentally verify this phenomenon in §4.1.3. With this observation,

we can compute EVPs for the data bits in one period only and reuse them for the following symbols in practice. As the period length in 802.11 is a small integer, by doing so, the computation complicity of Recitation is dramatically reduced and does not increase as the packet length.

3. IMPLEMENTATION

We implement Recitation on Atheros 9580 NICs that work on 802.11n 20 MHz channels with 64 subcarriers. Each 9580 NIC is connected to a Qualcomm System-on-Chip QCA9558 running an embedded Linux system, as shown in Figure 7(b). According to the 802.11 standard, the physical layer calculates CSI upon each correctly received packet when the sounding flag is set. We modify the driver to turn on this flag and let Atheros 9580 chipset export CSI, which causes no additional computation overhead. We also record both correct and corrupted packets at the receiver. Other useful information related to the packet transmission, *e.g.*, RSSI, noise floor, data rate, *et c.* is reported as well. This implementation forms the basis for all results reported in §4.

4. EVALUATION

We present a testbed-based experimental evaluation of Recitation. We begin in §4.1 with microbenchmarks evaluating Recitation's accuracy in predicting PER and error-prone bit positions, and measuring its computational complexity overhead. In §4.2 we present the results of experiments measuring end-to-end application performance. The parameter e, the number of coded bit errors to investigate, equals to e_d and $e_d + 1$, where e_d = 5, 3, 3, and 2 for 1/2, 2/3, 3/4, and 5/6 code rates, respectively. The maximum error burst length is set to 40, 32, 30, and 43 for rates from 1/2 to 5/6, respectively. These parameters have been investigated and determined in §2.2.2. We also implement other approaches used in three applications for the performance comparison.

4.1 Microbenchmarks

Methodology. We deploy Atheros 9580 nodes in our campus as access points (APs). These cover a path from a bus station to a research lab as shown in Figure 7 (a). We deploy another Atheros 9580 node as receiver to connect each AP to 50 random positions. At each position, the AP sends 10,000 UDP packets with 1,000-byte random payloads to the receiver. We iterate the transmissions for all eight rates using between one and three spatial streams. In each run, we fix the data rate and turn off link-layer retransmissions to measure the underlying PER. We thus obtain the PER statistics for all SISO and MIMO rates at each position. Throughout the experiment, the receiver records both correct and corrupted packets, and CSI from each correct packet. We thus obtain the decoding bit error patterns within each packet over lossy links. We then evaluate Recitation by comparing its prediction results with the measurement results.

4.1.1 Frequency selective fading

Our measurements in Figure 7 (a) cover typical 802.11 Wi-Fi environments, *e.g.*, an open-space hall, a parking lot surrounded by cars and stores, an office, etc, with different levels of frequency selective fading. Figure 7 (c) plots the CDF of the maximum subcarrier SNR difference from each CSI measurement. In 20 MHz channels, we observe a difference of at least 2.7 dB, nearly 10 dB for 50% of the links, and a maximum of 22 dB. From Figure 7 (c), we see that 40 MHz channels have even higher subcarrier diversity, from 7.6 dB up to 28 dB. As the trend of the 802.11 development is to further increase the bandwidth, we expect subcarrier diversity will increase in future Wi-Fi standards.

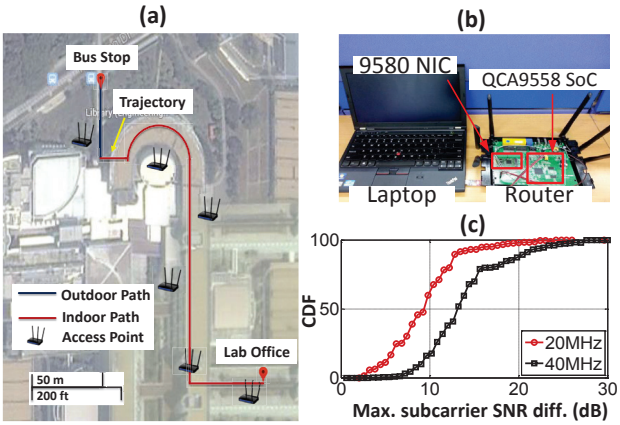

Figure 7: **(a)** Testbed map; **(b)** Atheros 9580 node used in the experiment; **(c)** CDF of the maximum subcarrier SNR difference (dB) in each CSI measurement.

4.1.2 PER prediction

Figure 8(a) and (b) depict the packet error rate (PER) prediction performance Recitation achieves with 802.11n convolutional codes. We also compare against the state-of-the-art approach Effective SNR (ESNR) [12].

802.11n SISO. Figure 8 (a) shows prediction accuracy for 802.11 single-antenna rates.[6] For all the links whose measured PERs fall into one bucket, we calculate the differences between the measured and predicted PERs, and plot the average prediction error as well as the error variance. From the figure, we see that Recitation's predictions are highly accurate in all buckets, with an average prediction error between 0.003 and 0.05, and variance less than 0.01. ESNR makes a prediction error of 0.22 on average. ESNR's inaccuracy arises because the PER versus SNR curve has a narrow transition range, and ESNR is a relatively coarse channel quality indicator whose predictions are more sensitive to uncertainty in the SNR measurement. ESNR thus provides mostly binary information about whether PER is less than a small threshold, *e.g.*, in bucket [0 ~ .01]. As Recitation can utilize all subcarrier SNRs in the EVP calculation, it accurately predicts PER even in the transition range.

802.11n MIMO. Figure 8 (b) depicts the PER prediction performance for 3 × 3 MIMO rates. Similar to Figure 8 (a), we divide the PER transition range into seven buckets. Due to larger throughput gaps between two consecutive MIMO rates, we obtain fewer measurements between 0.06 and 0.3 in the PER transition range. Therefore, the prediction errors are slightly higher than that in Figure 8 (a). Recitation's average prediction error varies between 0.002 and 0.07, while ESNR's average prediction error can be up to 0.34. Similar to SISO rates, ESNR achieves a small PER prediction error in the bucket [0 ~ .01]. In contrast, the average errors Recitation achieves are much smaller in all buckets.

LDPC codes. To evaluate Recitation with LDPC codes, we investigate the EECs of the 802.11 LDPC codes experimentally (different data rates employ different LDPC codes). We implement the 802.11 LDPC encoder/decoder and integrate them in our 802.11n specification-compliant transceiver introduced in §2.2.2. After a

[6]For clarity of presentation, we divide the PER range (0.0 to 0.3) into seven buckets. Most applications, like rate adaptation, partial packet recovery, *etc.*, work within this PER range.

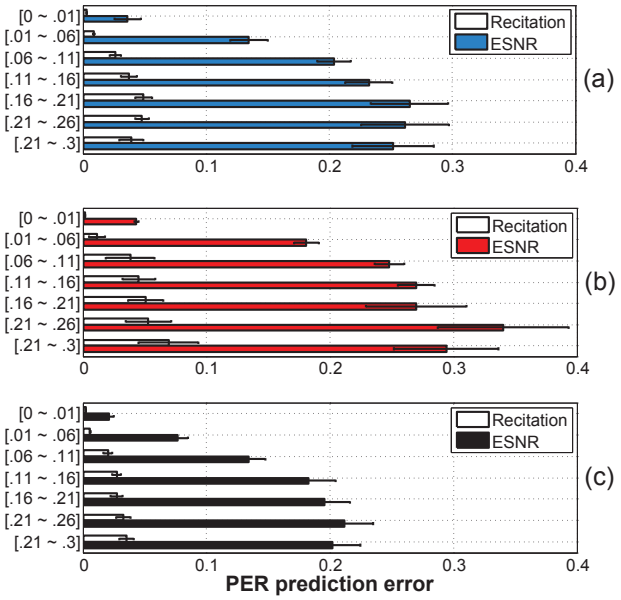

Figure 8: PER prediction comparison. **(a)** Convolutional codes, SISO rates (1×1); **(b)** Convolutional codes, MIMO rates (3×3); **(c)** LDPC codes, SISO rates (1×1).

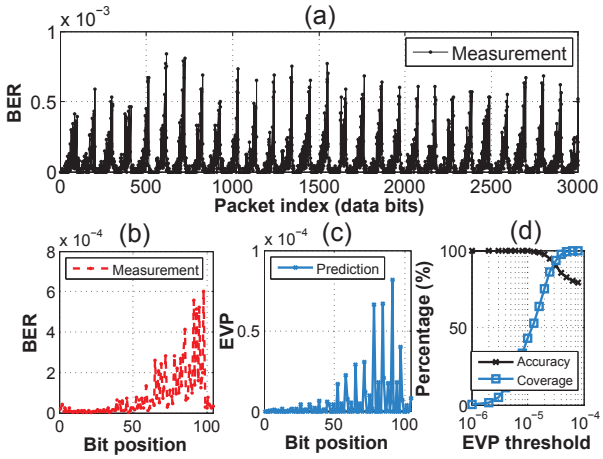

Figure 9: **(a)** BERs of each data bit in a packet; **(b)** Data bit BER in one period; **(c)** Data bit EVP in one period; **(d)** The prediction accuracy of error-prone bit positions.

LDPC encoder encodes a data packet, we manually flip coded bits to introduce coded bit errors. We then feed the *polluted* coded bits into LDPC decoders and repeat this process to evaluate the maximum number of coded bit errors each LDPC code can tolerate.

As the LDPC codes implementation on Atheros 9580 NICs as well as other NICs, like the widely used Intel 5300 NICs in the community, is not available to us, we use an 802.11n simulator to evaluate the PER prediction performance with LDPC codes. We adopt the CSI trace collected in Figure 7 as the ground truth for the channel, and transmit packets over the channel described by the CSIs. We then compare the predicted PER against the measured PER. Figure 8 (c) depicts that Recitation can also achieve an accurate PER prediction using LDPC codes. According to the statistics, the average prediction error varies from 0.002 to 0.035, and the variance is less than 0.01. The accuracy achieved in this figure is higher than that in Figure 8 (a) because simulations introduce less channel uncertainty. Similar to convolutional codes, the ESNR's average prediction error is much higher than Recitation, up to 0.21, and it performs well in the bucket [0 ~ .01] merely.

4.1.3 *Prediction of decoding error locality*

In this section we continue our evaluation of Recitation with convolutional codes on Atheros 9580 nodes. Figure 9 (a) plots the BERs of the data bits in one packet. We measure BER from a randomly-selected link in our test-bed using a $^1/_2$ code rate and QAM-16 modulation. For clarity, we show the BERs of the first 3,000 bits. The BERs of the remaining data bits follow similar patterns. From the figure, we observe (echoing similar observations by Miu *et al.* [33] and Han *et al.* [13]) that the BERs of each data bit are highly different, forming a pattern that has a period length equal to the size of the OFDM symbol. Therefore, we plot the average BER of the data bits in one period in Figure 9 (b), where the x-axis is each bit position in the period and the y-axis is the average BER over all the data bits with the same offset in each individual period. Figure 9 (b) clearly shows that some data bits are more likely to have decoding errors. In Figure 9 (c), we use Recitation to predict those error-prone positions. The predicted pattern matches the measured one, based on which we can identify all error-prone bit positions in one packet.

Since EVP does not describe the error dependence with other bits (§2.1), the EVP of each bit is smaller than the BER. Thus, to quantify prediction accuracy, we do not conduct a bitwise comparison. Instead, we divide each period into equal ranges, using thresholds to classify each into "safe" and "unsafe" clusters. All data bits in a safe range are considered to be safe and more important bits of a packet can be placed in the safe ranges.

Through our experiment, we find that it is sufficient to divide one period into 4 to 32 ranges for the eight data rates. For all the links with corrupted packets in our experiment, we calculate the range error rates and use an empirical threshold, *e.g.*, 0.04, to classify all the ranges into the "safe" and "unsafe" categories as ground truth. For each range, we then use Recitation to calculate its average EVP and adopt an EVP threshold to evaluate its prediction results. We divide our packet trace into training and testing sets. In the training set, Figure 9 (d) shows when the EVP threshold is small, Recitation can identify only a few safe ranges from the ground truth ("Coverage" in Figure 9 (d)), but identification accuracy is high. As the threshold increases, Recitation identifies more safe ranges but accuracy decreases slightly. We find that the EVP threshold 2.6×10^{-5} achieves a good tradeoff between the accuracy (92%) and the coverage (91%). In the testing set, Recitation achieves a similar performance. The setting is used for the application in §4.2.3.

4.1.4 *Computation and energy overhead*

Computation overhead. While Recitation is accurate in its PER and error-prone bit position predictions, it can be truly useful in practice only if its computational overhead is manageable. To this end, the parameter λ used in the "error combination pruning" technique controls the computation overhead and the EVP accuracy of Recitation. According to the 802.11 timing requirement in §2, an acceptable delay should be less than 0.25 *ms* such that the predictions for eight different data rates can be completed within two milliseconds. Figure 10 (a) depicts the average delay to predict PER for one PHY configuration when λ varies. When λ varies from 5 to 30, Figure 10 (a) shows that the computational delay of Recitation increases from 0.004 to 0.934 ms. For one such eligible λ

setting, e.g., $\lambda = 18$, Figure 10 (b) breaks down the computational overhead for individual data rates with the average delay at around 0.161 ms. The reason that the computation delays for rates 6.5, 13, 26, and 52 Mbit/s are larger is because they use the 1/2 and 2/3 code rates. One OFDM symbol thus contains more coded bits and there are more error combinations to calculate. It introduces much more computations than other three rates 19.5, 39, 58.5, and 65 Mbit/s, which adopts the 3/4 and 5/6 code rates.

In Figure 10(c), we evaluate the PER prediction accuracy under different choices of λ. For each link investigated in Figure 8, we vary λ to obtain different PER predictions and compare them with the measured results. Figure 10 (c) shows that when λ is small, e.g., $\lambda \leq 15$, the prediction error is large, *e.g.*, up to 0.23 on average. When λ is sufficiently large, Recitation achieves high accuracy, *e.g.*, the error decreases to 0.013 when $\lambda = 18$. As λ further increases, the accuracy improvement becomes marginal, while the computation overhead dramatically increases. Based on the results, we select $\lambda = 18$ as a default setting, which was used in both Figures 8 and 9. In Figure 10 (c), we also examine Recitation without the "error combination pruning" approximation. From the figure, we can see its average prediction error is 0.011, showing that this technique introduces a negligible accuracy loss, while reducing computational complexity dramatically.

Energy consumption. We measure the Recitation power consumption on Atheros 9580 nodes using an APPA 72 multimeter, by investigating its energy usage in comparison with the baseline when our design is not used. Since Recitation is a fully software-based system, power consumption mainly stems from CPU computation, which is lightweight. Our measurement shows that the additional power consumption caused by Recitation varies from 601 mW to 685 mW, which is comparable to or even longer than a variety of recent mobile applications [27, 60, 61], indicating that Recitation is energy efficient for mobile platforms. When a mobile device of Recitation communicates with a line-powered AP, the energy efficiency can be further improved: the AP can always compute the optimal configuration using the CSI measurement. For the downlink, the AP uses the optimal configuration to transmit packets; for the uplink, the AP sends the optimal configuration, instead of CSI [57], as feedback to avoid computation on the mobile device.

4.2 Applications

In this section, we demonstrate the utility of the accurate and fine-grained decoding predictions Recitation achieves through three representative applications: bit-rate adaptation, partial packet recovery, and unequal error protection-based video transmission. The first two applications aim to evaluate the accurate PER prediction of Recitation. In particular, the former one focuses on the PER prediction when PER is close to zero and the latter one focuses on the prediction with the PER transition range. The third application targets to evaluate the utility of the fine-grained error-prone bit position prediction. We use extensive trace-driven evaluations to compare against the state-of-the-art approaches.

Evaluation setup. We collect packet transmission and CSI traces in the laboratory (the bottom AP in Figure 7), considering both mobility and non-line-of-sight in the trace collection. We repeat the trace collection twice for 1×1 SISO and 3×3 MIMO settings, respectively. The AP continuously generates UDP packets. For each packet, AP sends out eight copies, iterating for all eight rates. The CSI from the eight copies are approximately the same since CSI is measured from the preamble, which does not depend on the rate. We thus view the transmissions of the eight copies as the ground truth for eight different rates when AP sends one packet over the current channel. During the AP's transmission, a receiver moves

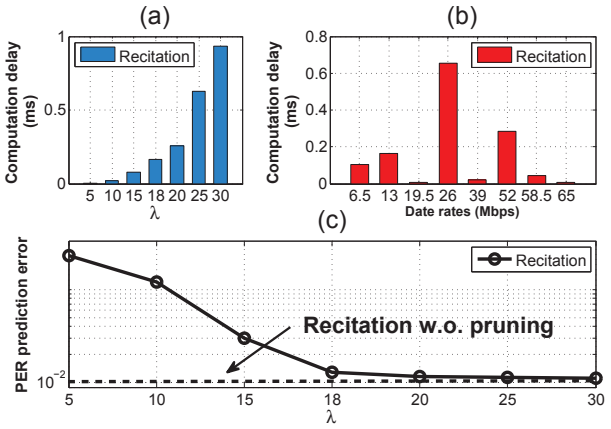

Figure 10: **(a)** Average Recitation computation delay for one rate when λ varies; **(b)** Recitation computation delay breakdown for different rates ($\lambda = 18$); **(c)** PER prediction errors when λ varies.

in a U-shaped route at a walking speed [15] in the laboratory and collects about 93,000 channel measurements. From CSI indices 0 to 30,000, the receiver moves far away from the AP and there is line-of-sight for most of time. From indices 30,000 to 50,000, the receiver takes a U-turn and traverses through a meeting room. The channel is hence relatively poor and non line-of-sight. In the rest of the CSI trace, the receiver moves towards the sender and there is no line-of-sight for most of time. In addition, the CSI feedback is included and its overhead is factored into the evaluation.

4.2.1 Wi-Fi rate selection

We evaluate the following bit-rate selection algorithms:

1. **Rect-Rate:** The sender uses the CSI from the receiver feedback to predict PER for each data rate by Recitation, and selects the highest rate with the predicted PER ≤ 0.1.
2. **ESNR:** The rate selection based on the proposed ESNR approach in [12]. ESNR is compared as it reports the best known throughput performance so far. ESNR has shown to outperform other recent approaches, *e.g.*, SoftRate [52]. We do not evaluate each of those individually.
3. **OPT:** An oracle selects the optimal data rate for each transmission, thus performance is an upper bound for all rate selection approaches in the experiment.

Some recent works, *e.g.*, [4, 38], propose to harness the frequency diversity to adapt rates, but they require a substantial redesign of the current 802.11 PHY with customized hardware. Therefore, we do not empirically compare Recitation with them. In this experiment, we feed the CSI trace into each approach. For each CSI, different approaches select different rates to transmit the packet, and the collected ground truth determines the transmission result for a particular scheme. We compare the throughput of each approach, where each reported throughput value is averaged in a window of 200 packets transmitted over 200 consecutive CSIs. Figures 11 and 12 plot average achieved throughput against time windows of 200 CSI readings.

802.11 SISO rates. Due to the inaccuracy of the PER prediction, ESNR selects suboptimal bit rates more frequently than Rect-Rate thus suffering a throughput penalty. Rect-Rate tracks the performance of OPT within 16% on average, while ESNR is only within 29%. Rect-Rate outperforms ESNR by 25.6% on average and 63.8% at most. To analyze the performance achieved by each approach, we characterize the rate selection accuracy of Rect-Rate

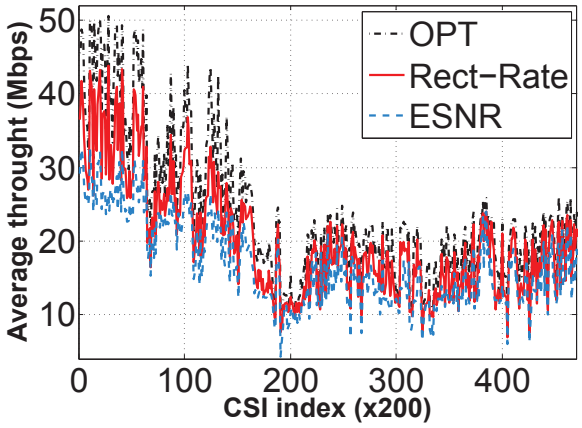

Figure 11: Throughput comparison for 802.11 SISO rates in human-speed mobility.

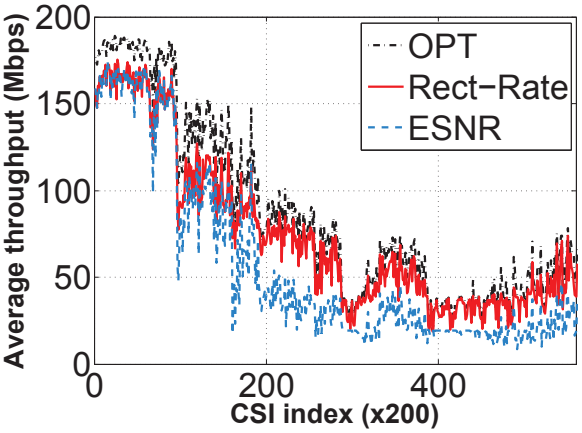

Figure 12: Throughput comparison for 802.11 MIMO rates in human-speed mobility.

and ESNR, with respect to OPT, for each transmitted packet. We classify the transmitted packets into three categories: over-selected, accurate, and under-selected data rates. Rect-Rate selects correct rates for about 70% of packets. The performance gap between Rect-Rate and OPT attributes to the 20% under-selected and 10% over-selected data rates. In comparison, ESNR has nearly half transmissions with under-selected data rates, which primarily limit its throughput.

802.11 MIMO rates. The performance of the two approaches with MIMO rates is similar to that in Figure 11. From the statistics, we find that the Rect-Rate outperforms ESNR by 33.8% on average and up to 78.2% occasionally. Referring to OPT, we observe that the fractions of the accurate data rate selection are 52% and 38% for Rect-Rate and ESNR, respectively. For 80% of transmissions, throughput gaps of Rect-Rate and ESNR (to OPT) are less than 17 Mbit/s and 35 Mbit/s, respectively. In addition, Rect-Rate tracks the performance of OPT within 14.3% on average, while ESNR is only within 34.4%.

Accelerated mobility. To evaluate each approach in different mobile environments, we sample the CSI trace, collected when the receiver moves at a walking speed, to accelerate the mobility of the sender. We sample the trace every c CSIs to form a new trace,

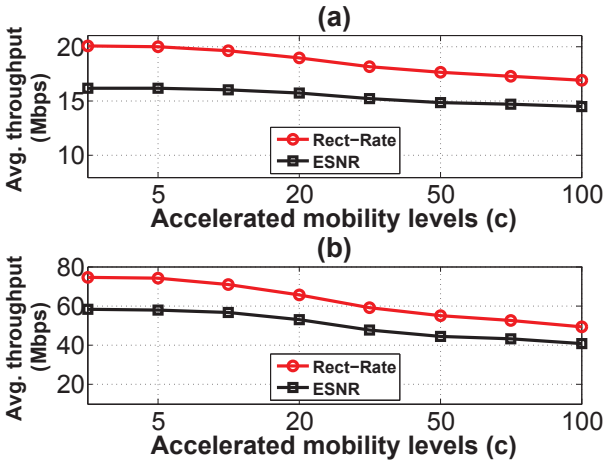

Figure 13: Throughput comparison with faster mobility for 802.11 **(a)** SISO; **(b)** MIMO rates.

which simulates a varying channel that the channel dynamics are accelerated c times. We evaluate the effect of mobility on each approach in Figure 13. Both approaches select the current data rate based on the past CSI measurement. They implicitly rely on the correlation between the current channel quality and the previous one. When the sampling interval c is increased to accelerate channel dynamics, the performance of both approaches decreases. OPT is not affected, and we omit it in Figure 13. But as c varies from one to 100, Rect-Rate can still outperform ESNR for both SISO and MIMO rates. Rect-Rate achieves 21% and 24% average throughput gains for SISO rates in Figure 13 (a) and for MIMO rates in Figure 13 (b), respectively.

Burst errors (interference). To evaluate the performance of each approach against burst errors [18, 25, 37], we repeat the CSI trace collection in the laboratory, but in addition to the background traffic, we introduce an external interferer close to the receiver and a hidden terminal to the sender. The interferer periodically generates interference packets without carrier sense, with an interval between two error bursts of 11-packets' worth of airtime. We vary the error burst duration from two to eight packets in the experiment. To deal with burst errors, the sender will use the next lower rate if CSI feedback is not received after certain packets transmitted [14], which is set at two in our evaluation.

Figure 14(a) illustrates a snapshot of the rate selections of two approaches when the burst duration is five. When burst errors occur, two approaches both gradually decrease their data rates. When such an interference disappears, after one packet gets successfully received by the receiver, the sender can promptly adjust its data rate according to the CSI feedback. The result indicates that the CSI-based rate selection approaches can promptly recover from burst errors. In Figure 14 (b), due to the accurate PER prediction, Rect-Rate outperforms ESNR by 30% on average when the burst error duration varies from two to eight packets.

4.2.2 *Partial packet recovery*

We evaluate the following approaches:

1. **Maranello:** It is the most recent partial packet recovery approach with the best reported performance [14]. It divides a packet into blocks and only retransmits erroneous blocks when the transmission fails.
2. **Rect-Maranello:** Before a packet transmission, Rect-Maranello uses Rect-Rate to select the data rate and further evaluates the

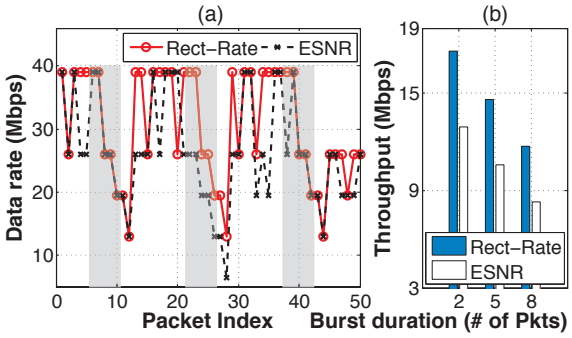

Figure 14: Throughput in presence of burst errors. Shadow area indicates burst errors. **(a)** Rate selection snapshot; **(b)** Average throughput under different burst error durations.

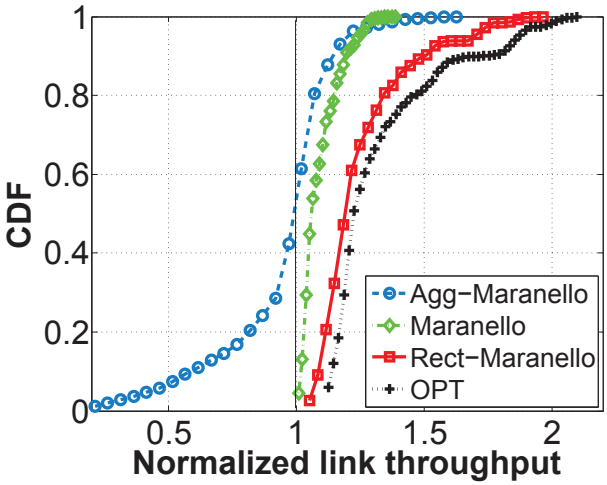

Figure 15: Throughput comparison for partial packet recovery.

PER of the next higher rate. If this PER is not sufficiently high, *e.g.*, < 0.3, Rect-Maranello transmits the packet with the augmented rate. It degrades to the original Maranello when the transmission fails. The data rate is not increased for retransmissions in the experiment.

3. **Agg-Maranello:** To understand the utility of the accurate PER prediction, we also compare against an aggressive design, Agg-Maranello, that always augments to the next higher data rate to harness more partial packets, suggested by μACK [58]. Similar to Rect-Maranello, the data rate is not increased for retransmissions neither.

We use the CSI trace collected in §4.1. Figure 15 depicts the throughput gains achieved by the three approaches, normalized to the throughput of Rect-Rate (without the data rate augmentation) in §4.2.1. As Rect-Rate can select the highest data rate with a low PER, the throughput gain of Maranello is limited, *e.g.*, 1.1× on average. Maranello has an advantage only when the transmission failure occurs occasionally. For Agg-Maranello, we find that it even performs worse than Rect-Rate in our experiment. It is because when the data rate is increased, the PER of the next rate could be moderate or very high. Without an accurate PER assessment, aggressively increasing the data rate may cause huge transmission overhead, which overwhelms the throughput gain from the higher rate. In our experiment, Agg-Maranello causes a 2% throughput loss. Benefiting from accurate PER prediction, Rect-Maranello can wisely increase the rate to make a full use of the channel bandwidth. According to the statistics, Rect-Maranello outperforms Maranello and Agg-Maranello by 16% and 30% on average, and 50% and 400% at most. Rect-Maranello performs within 5% of OPT. μACK in [58] can also harvest such a throughput gain with the help of extra hardware (additional antennas working on another frequency band). However, Figure 15 shows that without an accurate PER prediction, it is challenging to leverage partial packet recovery to fully utilize the channel bandwidth.

To further analyze the throughput each approach achieves, we find: Rect-Maranello selects correct rates for most packets. For Maranello, 37% of rates could be augmented. For Agg-Maranello, around half of the rates are over-selected.

4.2.3 Unequal bit protection

Packet bits may have different importance, and one of the most representative applications is media streaming [1, 43]. For example, in MPEG-4 video, the packet header is more important than the payload, and I-frames in payload in turn have a higher priority than other frames, *e.g.*, P and B frames. The media player can tolerate decoding errors in these frames, playing the video with a reduced quality. In contrast, when errors occur in I-frames, video quality drops significantly, and when errors occur in the header, the entire packet cannot contribute to the video playback. We evaluate the following approaches for this application:

1. **Rect-Video:** Each packet consists of multiple frames and a fixed-length header. Prior to the packet transmission, sender uses the EVP threshold to identify "safe" ranges in one data bit period introduced in Figure 9 and expand them to the entire packet. In our current design, we provide two safety levels: "safe" and "unsafe", where the EVP threshold used for this classification has been determined in §4.1.3. The packet header and I-frames are given a higher priority than other frames, which are placed in "safe" ranges sequentially. When "safe" ranges are occupied, the remaining bits are sequentially placed in "unsafe" ranges. In our current design, we use only one byte of extra overhead to record the original position of I-frames in the packet for the receiver to recover the original packet.
2. **Stan-Video:** Denotes the existing approach that treats every bit equally in the transmission. We adopt Rect-Rate to select rates for Stan-Video and focus on the performance comparison for the unequal error protection.

We adopt the standard metric, peak signal-to-noise ratio (PSNR), to measure the video quality [1]. PSNR is on the log scale, and a differences above 0.5 dB are visually detectable. When PSNR is above 37 dB, the video quality is considered to be "excellent" and "bad" when PSNR drops below 20 dB [8, 43]. We use the CSI trace collected in § 4.2.1 for evaluation. The sender generates a video streaming to the receiver at 30 frames per second. The video is MPEG-4 encoded and looped until the end of the experiment. In Figure 16, we plot the CDF of PSNRs achieved by the two approaches. We do not distinguish PSNRs that are greater than 40 dB since the video quality is excellent when it is above 37 dB. Similarly, we do not distinguish PSNRs less than 18 dB. From the figure, we see that half the time, the video quality in Rect-Video can be nearly 6 dB higher than Stan-Video, and the improvement can be up to 11 dB at most. The video quality gain of Rect-Video comes from the good protection of the packet header and I-frames.

In Figure 17, we further evaluate two approaches in three typical 802.11 WLANs environments: a open-space academic hall, a large library, and an indoor office, where the frequency selective fading levels increase. Figure 17 shows that Rect-Video outperforms Stan-

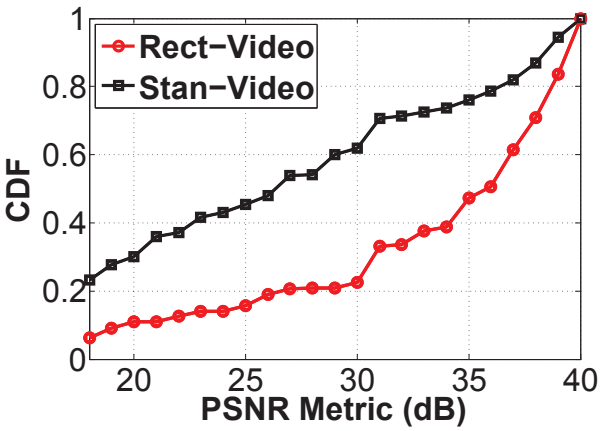

Figure 16: CDF of PSNRs achieved by two approaches.

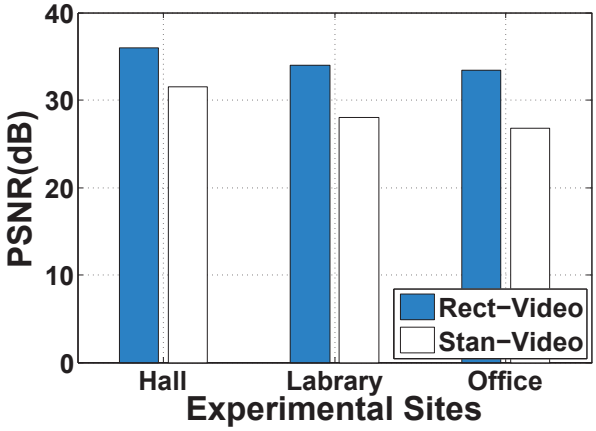

Figure 17: PSNR comparison at different positions.

Video in all three scenarios. The average PSNR gain over Stan-Video is at least 4.5 dB in hall and up to 6.6 dB in office.

5. DISCUSSION

Relationship with BER. One error event in the decoder usually results in a sequence of decoded data bit errors. The consequent "smearing" of probabilities across the data bits causes the BER of one data bit to be raised by errors from prior data bits. Hence, BER does not provide a direct measurement due to the dependence introduced among different packet bits by the encoding, *i.e.*, PER is not equal to $1 - \prod_{i=1}^{N}(1 - p_i)$, where p_i is the BER of each data bit i and N is the total number of packet bits. In contrast, there does exist a PER-EVP relationship (§2.2.1).

Packet prediction rationale. The prediction of packet reception is possible because the channel coherence time, within which the channel quality is relatively stable [10]. 802.11 Wi-Fi is mainly targeted at a static environment or an environment with walking-speed mobility, where the channel coherence time varies from tens of milliseconds to seconds. Since the transmission delay of one wireless packet is about one to two milliseconds, Recitation can confidently predict for future packets in the above environments.

Interfering packets. Recitation focuses on predicting wireless packet reception when packet collisions are not the dominant factor limiting throughput. While Recitation does not predict the effect of interfering packets [47, 48], the sender can promptly adjust its data rate upon receiving the CSI feedback from the receiver when interference disappears (see Figure 14). Hence, Recitation can act quickly against burst errors and achieve experimental gains in the presence of background traffic in unlicensed Wi-Fi spectrum (§4). Joint consideration of Recitation and interfering packets is possible, *e.g.*, exploring the SINR metric.

Scope of Recitation. The design of Recitation focuses on 802.11 WLANs, in which convolutional codes is the default coding scheme and LDPC codes is an emerging alternative. The packet bit processing in commodity Wi-Fi NICs is regulated by 802.11 standards and the sole input of our design is CSI. Recitation is thus generic for different Wi-Fi chipsets.

6. RELATED WORK

Subcarrier aggregation-based techniques. ESNR [12] has the advantage that it can be derived from the CSI in a Wi-Fi packet header and can translate a wideband channel to an equivalent narrow band channel to facilitate the upper-layer analysis. But as we show in §4, ESNR leaves performance on the table due to the approximation of channel quality with a single SNR value. Since the packet error rate (PER) versus SNR relationship has a very narrow transition range, a small inaccuracy in SNR measurement can cause a significant PER prediction error. In addition, ESNR does not provide per-bit BER confidences within each packet, making it unsuitable for our purposes. SoftRate [52] bit rate adaptation similarly aggregates BER across subcarriers in a similar computation to ESNR. Recitation emulates SoftRate's per-bit confidence computation with just the CSI information from the header. Error estimating codes (EEC) [8] adopt an advanced sampling technique to estimate the data bit BER, but only estimate the average packet BER.

Approximate communication. Apex [43] demonstrates the utility of unequal error protection for the purpose of media applications such as video or voice communication that can tolerate loss. While Apex demonstrates convincing experimental gains for video applications, it is designed and implemented only for 16-QAM and higher-order QAM modulations [43, Table 1, Table 2]. Recitation's design, in contrast, works with any QAM modulation. In additional, Recitation does not require any hardware modifications, unlike Apex—it is realizable on commodity Wi-Fi chipsets.

Softcast [19], FlexCast [1], and ParCast [29] recently propose novel joint source-channel coding schemes to deliver video quality that unequally protect video contents. Those works, however, either require additional physical layer information support (SoftPHY [20] in FlexCast) or a completely new physical design (Softcast and ParCast). On the contrary, Recitation enables unequal error protection via the CSI information from the packet header available on commodity Wi-Fi chipsets.

Partial packet recovery. Partial packet recovery approaches such as PPR [21], SOFT [55], Maranello [14], ZipTx [28] and Unite [56] minimize retransmission overhead. They have the advantage when packet collisions are the dominant factor limiting throughput or the channel is extremely lossy even the lowest data rate cannot reliably transmit a packet. Given the data rate selected bit-rate adaptation selects with a low PER, the throughput may still increase when the bit-rate is further augmented, if the throughput gain from the higher data rate outweighs the retransmission overheads. The accurate PER Recitation predicts enables such an assessment to fully utilize the channel.

Bit-rate adaptation. A variety of SNR-based bit-rate adaptation approaches exist, *e.g.* PBAR [16], OAR [42], CHARM [22], BlockRate [49], and Medusa [44]. Due to frequency selective fading,

Zhang *et al.* [59] and Camp *et al.* [6] observe that the SNR measurement needs careful calibration. To overcome the SNR measurement issue, many solutions utilize packet reception statistics, *e.g.*, SampleRate [5], RRAA [54], and MIMO-Rate [35]. Shen *et al.* [46] further studies the rate adaptation in multi-user MIMO networks. [3, 7, 24] focus on the rate adaptation analysis, [23, 39] concentrate on the energy/power control, and [32, 62] explore to correlate the link behaviors to rate adaptation in a target network. Recent studies further propose to adapt data rates by directly measuring coded BER [52] or symbol-level dispersion in AccuRate [45], but require specialized hardware. EEC [8] can adapt rates based on the estimated data bit BER without the hardware modification. Although most of above works are CSI-agnostic, they mainly adapt, instead of selecting, bit rates. The adaptation is less effective when channels change faster and the configuration space is more complex (*e.g.*, MIMO). The most recent approach [12] uses ESNR for the bit-rate selection. Benefiting from an accurate PER prediction, Recitation outperforms the best reported performance for both 802.11 SISO and MIMO rates.

Other physical-layer designs. Bhartia *et al.* [4] proposes a smart interleaver to directly map data bits to different subcarriers based on the bit importance. FARA [38] designs independent modulation and interleaving for each subcarrier to maximize throughput on each subcarrier. Both approaches harness frequency selective fading by substantially redesigning the current 802.11 PHY. However, the hardware-level implementation and transceiver coordination difficulties are the two major hurdles that prohibit them being adopted in practice. Recent rateless codes such as Strider [11] and Spinal [36] can avoid the channel measurement. They however require either a new PHY design [11] or dedicated hardware to accelerate the rateless decoding [36], not applicable to the 802.11 PHY. COLLIE [40] is a system that statistically discriminates between loss due to collision and loss due to a weak received SNR. It is thus complementary to Recitation in much the same way SoftRate is.

7. CONCLUSION

We have described Recitation, a fully software-based system that uses lightweight CSI information from Wi-Fi NICs to make precise predictions about the number and location of bit errors in subsequent packets. It works with 802.11's standard convolutional codes, and recent LDPC codes. We have shown significant performance improvements for three different applications: bit rate adaptation, partial packet recovery, and unequal protection-based video streaming.

Acknowledgments

We would like to thank the anonymous reviewers and shepherd for their useful feedback. This work is supported by Singapore MOE AcRF Tier 2 Grant MOE2012-T2-1-070 and NTU Nanyang Assistant Professorship (NAP) Grant M4080738.020. The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement no. 279976.

References

[1] S. Aditya and S. Katti. FlexCast: Graceful wireless video streaming. In *Proc. of ACM MobiCom*, 2011.

[2] A. Akella, G. Judd, S. Seshan, and P. Steenkiste. Self-management in chaotic wireless deployments. In *Proc. of ACM MobiCom*, 2005.

[3] M. Y. Arslan, K. Pelechrinis, I. Broustis, S. V. Krishnamurthy, S. Addepalli, and K. Papagiannaki. Auto-configuration of 802.11 n wlans. In *Proc. of ACM CoNEXT*, 2010.

[4] A. Bhartia, Y.-C. Chen, S. Rallapalli, and L. Qiu. Harnessing frequency diversity in Wi-Fi networks. In *Proc. of ACM MobiCom*, 2011.

[5] J. Bicket. Bit-Rate Selection in Wireless Networks. Master's thesis, Massachusetts Inst. of Technology, 2005.

[6] J. Camp and E. Knightly. Modulation rate adaptation in urban and vehicular environments: Cross-layer implementation and experimental evaluation. In *Proc. of ACM MobiCom*, 2008.

[7] P. Chaporkar, A. Proutiere, and B. Radunoviac. Rate adaptation games in wireless LANs: Nash equilibrium and price of anarchy. In *Proc. of IEEE INFOCOM*, 2010.

[8] B. Chen, Z. Zhou, Y. Zhao, and H. Yu. Efficient error estimating coding: Feasibility and applications. In *Proc. of ACM SIGCOMM*, 2010.

[9] S. K. Chilappagari and B. Vasic. Error-correction capability of column-weight-three ldpc codes. *IEEE Transactions on Information Theory*, 2009.

[10] A. Goldsmith. *Wireless communications*. Cambridge university press, 2005.

[11] A. Gudipati and S. Katti. Strider: Automatic rate adaptation and collision handling. In *Proc. of ACM SIGCOMM*, 2011.

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

[13] B. Han, L. Ji, S. Lee, B. Bhattacharjee, and R. R. Miller. Are all bits equal?: experimental study of IEEE 802.11 communication bit errors. *IEEE/ACM Transactions on Networking*, 2012.

[14] B. Han, A. Schulman, F. Gringoli, N. Spring, B. Bhattacharjee, L. Nava, L. Ji, S. Lee, and R. Miller. Maranello: Practical partial packet recovery for 802.11. In *Proc. of USENIX NSDI*, 2010.

[15] J. Han, C. Qian, X. Wang, D. Ma, J. Zhao, P. Zhang, W. Xi, and Z. Jiang. Twins: Device-free object tracking using passive tags. In *Proc. of IEEE INFOCOM*, 2014.

[16] G. Holland, N. Vaidya, and P. Bahl. A rate-adaptive MAC protocol for multi-hop wireless networks. In *Proc. of ACM MobiCom*, 2001.

[17] IEEE Standard for Information Technology—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer Specifications, 2012.

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

[19] S. Jakubczak and D. Katabi. A cross-layer design for scalable mobile video. In *Proc. of ACM MobiCom*, 2011.

[20] K. Jamieson. *The SoftPHY Abstraction: from Packets to Symbols in Wireless Network Design*. PhD thesis, MIT, 2008. http://hdl.handle.net/1721.1/41857.

[21] K. Jamieson and H. Balakrishnan. PPR: Partial packet recovery for wireless networks. In *Proc. of ACM SIGCOMM*, 2007.

[22] G. Judd, X. Wang, and P. Steenkiste. Efficient channel-aware rate adaptation in dynamic environments. In *Proc. of ACM MobiSys*, 2008.

[23] M. O. Khan, V. Dave, Y.-C. Chen, O. Jensen, L. Qiu, A. Bhartia, and S. Rallapalli. Model-driven energy-aware rate adaptation. In *Proc. of ACM MobiHoc*, 2013.

[24] R. Kumar, S. Tati, F. De Mello, S. V. Krishnamurthy, and T. L. Porta. Network coding aware rate selection in multi-rate IEEE 802.11. In *Proc. of IEEE ICNP*, 2010.

[25] L. E. Li, R. Alimi, D. Shen, H. Viswanathan, and Y. R. Yang. A general algorithm for interference alignment and cancellation in wireless networks. In *Proc. of IEEE INFOCOM*, 2010.

[26] L. E. Li, K. Tan, Y. Xu, H. Viswanathan, and Y. R. Yang. Remap decoding: Simple retransmission permutation can resolve overlapping channel collisions. In *Proc. of ACM MobiCom*, 2010.

[27] R. LiKamWa, B. Priyantha, M. Philipose, L. Zhong, and P. Bahl. Energy characterization and optimization of image sensing toward continuous mobile vision. In *Proc. of ACM MobiSys*, 2013.

[28] K. C.-J. Lin, N. Kushman, and D. Katabi. Ziptx: Harnessing partial packets in 802.11 networks. In *Proc. of ACM MobiCom*, 2008.

[29] X. L. Liu, W. Hu, Q. Pu, F. Wu, and Y. Zhang. Parcast: Soft video delivery in MIMO-OFDM WLANs. In *Proc. of ACM MobiCom*, 2012.

[30] F. Lu, G. M. Voelker, and A. C. Snoeren. SloMo: Downclocking Wi-Fi communication. In *Proc. of USENIX NSDI*, 2013.

[31] D. J. MacKay. Good error-correcting codes based on very sparse matrices. *IEEE Transactions on Information Theory*, 1999.

[32] M. Mirza, P. Barford, X. Zhu, S. Banerjee, and M. Blodgett. Fingerprinting 802.11 rate adaption algorithms. In *Proc. of IEEE INFOCOM*, 2011.

[33] A. Miu, H. Balakrishnan, and C. E. Koksal. Improving loss resilience with multi-radio diversity in wireless networks. In *Proc. of ACM MobiCom*, 2005.

[34] J. Pang, B. Greenstein, M. Kaminsky, D. McCoy, and S. Seshan. WiFi-Reports: Improving wireless network selection with collaboration. In *Proc. of ACM MobiSys*, 2009.

[35] 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.

[36] J. Perry, P. Iannucci, K. Fleming, H. Balakrishnan, and D. Shah. Spinal codes. In *Proc. of ACM SIGCOMM*, 2012.

[37] L. Qiu, Y. Zhang, F. Wang, M. K. Han, and R. Mahajan. A general model of wireless interference. In *Proc. of ACM MobiCom*, 2007.

[38] H. Rahul, F. Edalat, D. Katabi, and C. G. Sodini. Frequency-aware rate adaptation and MAC protocols. In *Proc. of ACM MobiCom*, 2009.

[39] K. Ramachandran, R. Kokku, H. Zhang, and M. Gruteser. Symphony: synchronous two-phase rate and power control in 802.11 WLANs. In *Proc. of ACM MobiSys*, 2008.

[40] 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. of IEE INFOCOM*, 2008.

[41] T. J. Richardson and R. L. Urbanke. Efficient encoding of low-density parity-check codes. *IEEE Transactions on Information Theory*, 2001.

[42] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly. Opportunistic media access for multirate ad hoc networks. In *Proc. of ACM MobiCom*, 2002.

[43] S. Sen, S. Gilani, S. Srinath, S. Schmitt, and S. Banerjee. Design and implementation of an "approximate" communication system for wireless media applications. In *Proc. of ACM SIGCOMM*, 2010.

[44] S. Sen, N. K. Madabhushi, and S. Banerjee. Scalable Wi-Fi media delivery through adaptive broadcasts. In *Proc. of USENIX NSDI*, 2010.

[45] S. Sen, N. Santhapuri, R. R. Choudhury, and S. Nelakuditi. AccuRate: Constellation based rate estimation in wireless networks. In *Proc. of USENIX NSDI*, 2010.

[46] W. Shen, Y. Tung, K. Lee, K. C. Lin, S. Gollakota, D. Katabi, and M.-S. Chen. Rate adaptation for 802.11 multiuser MIMO networks. In *Proc. of ACM MobiCom*, 2012.

[47] K. Srinivasan, M. Jain, J. I. Choi, T. Azim, E. S. Kim, P. Levis, and B. Krishnamachari. The κ factor: inferring protocol performance using inter-link reception correlation. In *Proc. of ACM MobiCom*, 2010.

[48] K. Srinivasan, M. A. Kazandjieva, S. Agarwal, and P. Levis. The β-factor: measuring wireless link burstiness. In *Proc. of ACM SenSys*, 2008.

[49] X. Tie, A. Seetharam, A. Venkataramani, D. Ganesan, and D. L. Goeckel. Anticipatory wireless bitrate control for blocks. In *Proc. of ACM CoNEXT*, 2011.

[50] D. Tse and P. Viswanath. *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[51] A. J. Viterbi. Convolutional codes and their performance in communication systems. *IEEE Transactions on Communication Technology*, 1971.

[52] M. Vutukuru, H. Balakrishnan, and K. Jamieson. Cross-layer wireless bit rate adaptation. In *Proc. of ACM SIGCOMM*, 2009.

[53] Y. Wang, J. Liu, Y. Chen, M. Gruteser, J. Yang, and H. Liu. E-eyes: Device-free location-oriented activity identification using fine-grained WiFi signatures. In *Proc. of ACM MobiCom*, 2014.

[54] S. H. Wong, H. Yang, S. Lu, and V. Bharghavan. Robust rate adaptation for 802.11 wireless networks. In *Proc. of ACM MobiCom*, 2006.

[55] G. R. Woo, P. Kheradpour, D. Shen, and D. Katabi. Beyond the bits: cooperative packet recovery using physical layer information. In *Proc. of ACM MobiCom*, 2007.

[56] J. Xie, W. Hu, and Z. Zhang. Revisiting partial packet recovery in 802.11 wireless lans. In *Proc. of ACM MobiSys*, 2011.

[57] X. Xie, X. Zhang, and K. Sundaresan. Adaptive feedback compression for MIMO networks. In *Proc. of ACM MobiCom*, 2013.

[58] J. Zhang, H. Shen, K. Tan, R. Chandra, Y. Zhang, and Q. Zhang. Frame retransmissions considered harmful: improving spectrum efficiency using micro-acks. In *Proc. of ACM MobiCom*, 2012.

[59] J. Zhang, K. Tan, J. Zhao, H. Wu, and Y. Zhang. A practical SNR-guided rate adaptation. In *Proc. of IEEE INFOCOM*, 2008.

[60] L. Zhang, X.-Y. Li, W. Huang, K. Liu, S. Zong, X. Jian, P. Feng, T. Jung, and Y. Liu. It starts with igaze: Visual attention driven networking with smart glasses. In *Proc. of ACM MobiCom*, 2014.

[61] X. Zhang and K. G. Shin. E-MiLi: Energy-minimizing idle listening in wireless networks. In *Proc. of ACM MobiCom*, 2011.

[62] X. Zhou, J. Zhao, and G. Yang. Correlation based rate adaptation via insights from incomplete observations in 802.11 networks. In *Proc. of IEEE ICC*, 2007.