# Augmenting Wide-band 802.11 Transmissions via Unequal Packet Bit Protection

Yaxiong Xie*, Zhenjiang Li†, Mo Li*, Kyle Jamieson‡§
*School of Computer Engineering, Nanyang Technological University
†Department of Computer Science, City University of Hong Kong
‡Department of Computer Science, University College London
§Department of Computer Science, Princeton University

***Abstract*—Due to frequency selective fading, modern wide-band 802.11 transmissions have unevenly distributed bit BERs in a packet. In this paper, we propose to unequally protect packet bits according to their BERs. By doing so, we can best match the effective transmission rate of each bit to channel condition, and improve throughput. The major design challenge lies in deriving an accurate relationship between the frequency selective channel condition and the decoded packet bit BERs, all the way through the complex 802.11 PHY layer. Based on our study, we find that the decoding error of a packet bit corresponds to dense errors in the underlying codeword bits, and the BER can be truthfully approximated by the codeword bit error density. With above observation, we propose UnPKT, scheme that protects packet bits using different MAC-layer FEC redundancies based on bit-wise BER estimation to augment wide-band 802.11 transmissions. UnPKT is software-implementable and compatible with the existing 802.11 architecture. Extensive evaluations based on Atheros 9580 NICs and GNU-Radio platforms show the effectiveness of our design. UnPKT can achieve a significant goodput improvement over state-of-the-art approaches.**

## I. Introduction

Modern 802.11 WLANs (Wireless Local Area Networks) work in wide-band wireless channels, e.g., 20/40MHz channels in 802.11n [1], [2], [3]. Due to frequency selective fading [4], [5], [6], codeword bits, i.e., packet bits after channel coding, experience different channel qualities as they are transmitted over different sub-carriers. 802.11 provides a uniform protection within each transmitted packet, using the same channel coding rate and modulation order across all bits. The BERs (Bit Error Rate) of decoded packet bits are hence highly *uneven*—decoding errors are more likely to occur in certain bit positions of a packet—which is also observed in the prior measurements [7], [8]. Error-prone bits inherently dominate transmission failures and impair throughput. The default 802.11 rate selection mechanism can help to alleviate the error-prone bits [9], [10]. With a conservatively-selected rate, all packet bits will experience low BERs, which however under-utilizes the channel bandwidth and reduces the throughput. Simply relying on rate selection cannot fundamentally address the problem.

To ensure full utilization of the channel capacity and successful packet delivery, accurate protection should be given to packet bits "unequally" according to their BER level and BER distribution. In such a way, the effective transmission rate of each packet bit can best match its experienced channel condition, and the overall packet bit BERs become even and low. Transmission failures thus can be prevented, and the throughput improves. PHY layer "unequal" bit protection has been explored in the previous literature [6], [11]. The basic idea is to break the existing 802.11 uniform bit protection, and provide codeword bits with the most appropriate redundancies in the PHY layer according to the sub-carrier quality or importance of packet bits. However, those approaches need to redesign the PHY layer, which causes excessive development overhead and is not compatible to the existing 802.11 devices.

To comply with the 802.11 framework, in this paper, we propose to supplement the MAC-layer FEC (Forwarding Error Correction) redundancies to packet bits before they are sent to the 802.11 PHY layer. Different redundancies are calculated and provided according to the BERs of different segments of packet bits. By doing so, we can provide effective and accurate packet protection without altering 802.11. The major design challenge lies in deriving the accurate relationship between the heterogeneous subcarrier quality and the decoded packet BERs, all the way through the entire 802.11 PHY layer. In particular, codeword bits traverse different sub-carriers and undergo complicated PHY-layer operations at the receiver before decoding, e.g., demodulation, deinterleaving, etc. However, the estimation of the final packet bit BERs from the codeword errors could be very difficult due to the arbitrarily distributed codeword bit errors as a result of frequency selective fading. Existing 802.11 packet bit BER analysis considers a homogenous codeword error distribution [12], which can be fully described in mathematics and greatly simplifies the analysis, but only works over the flat-fading channel. The decoded packet bit BERs with an arbitrary codeword bit error distribution is still unknown. Recitation [8] considers frequency selective fading channels and proposes the EVP metric to indicate the error-prone positions within a packet. Nevertheless, the EVP metric represents the likelihood of decoding error events [8], which cannot directly map to packet bit BERs.

A straightforward solution to sidestep such a challenge is to estimate an averaged BER of packet bits, according to which, a uniform protection (with a long FEC block length) is provided to the entire packet. Using an averaged BER, however, leads to an inaccurate estimation of protection needed since not only the amount of bit errors but their distribution will affect the

978-1-4673-9953-1/16/$31.00 ©2016 IEEE

number of block errors and thus the amount of protection. On the other hand, the encoding/decoding overhead of block codes (e.g., RS codes, LDPC codes, etc.) significantly grows with the block size, making it computationally infeasible to treat the whole packet as one block.

In this paper, we study the 802.11 decoding process and observe that the decoding error of a packet bit corresponds to a series of dense errors in a group of underlying codeword bits. We find that the probability of dense codeword bit errors, together with the error density, can well approximate the packet bit BERs. In summary, the contributions of this paper are as follows.

- We propose an unequal packet bit protection approach, UnPKT, for 802.11 WLANs. UnPKT protects a packet based on the BERs of different segments of the packet. Transmission failures are largely prevented while throughput improves. UnPKT is implemented solely in software.
- We address the fine-grained packet bit BER estimation issue. We propose a BER estimator, which can estimate the bit-wise packet bit BERs. Our estimator is computationally efficient and takes CSI as the sole input.
- We use commodity Atheros 9580 Wi-Fi NICs to validate the effectiveness of the UnPKT design. We further compare UnPKT with the state-of-the-art approaches using the trace-driven evaluation on the GNU Radio platform. The results show that the goodput gain achieved by UnPKT is significant, ranging from 12.2% to 200%.

In the rest of this paper: Section II gives the literature review. Section III states the motivation, and observation of UnPKT. We detail its design in Section IV, and introduce the implementation and evaluation results in Section V. We finally conclude in Section VI.

## II. Related Work

**Frequency Diversity**. Due to frequency selective fading, the subcarriers of OFDM could exhibit highly different channel quality [11], [6], which has fundamental impact on the performance of 802.11 networks. To mitigate the impacts of frequency selective fading, FARA [11] introduces separated modulations and channel coding for each sub-carrier. [6] harnesses this frequency diversity by sending bits with higher priority to subcarriers with better channel quality. Apex [13] leverages the different reliabilities of codeword bits of constellation symbols to achieve unequal bit protection for video or voice communications. These designs require a complete redesign of 802.11 PHY layer, including the channel coding, interleaving and modulation modules. UnPKT, on the contrary, completely complies with the 802.11 framework and the existing 802.11 hardware design. UnPKT explicitly addresses the challenge of predicting diverse packet bit BERs within 802.11 framework.

**Cross-Layer FEC**. Cross-layer FEC provides additional packet bit protection above the PHY layer. It is mainly applied to the APP layer for the video/audio streaming by protecting certain key information in the streams, e.g., key

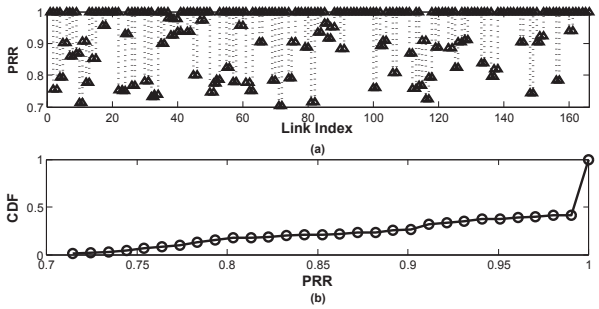


Fig. 1. (a) PRRs of all 166 links in three test-beds; (b) CDF of the PRR.

video frames, meta header information, etc., with additional redundancies [14], [15]. The redundancies can be equally or unequally applied to the protection objects. Those approaches are however content-aware and limited to specific applications. On the contrary, UnPKT is content-oblivious and serve in a general purpose for all 802.11 wide-band transmissions. On the MAC-layer, [16] proposes to add equal MAC-layer FEC to battle packet bit errors in narrow-band channels. The redundancy is empirically added to each packet. Compared with [16], UnPKT is designed for modern 802.11 WLANs working in wide-band channels and solves the unique issue of the uneven packet bit BER estimation. Our experiments show that the design [16] works poorly in wide-band channels. In contrast, UnPKT achieves much better performance.

**Hybrid ARQ**. With hybrid ARQ, packet bits are encoded, e.g., by the convolutional code, before transmission, and the whole packet or additional coded bits are retransmitted if the original transmission fails [17]. 802.11 essentially follows the hybrid ARQ principle. However, retransmission causes non-negligible overhead, e.g., transmission delay and MAC-layer overhead. A variety of partial packet retransmission schemes [18], [19], [20] have been proposed to improve the retransmission efficiency, which reduce the transmission delay of each retransmission, but still suffer from the MAC-layer overhead. Different from those approaches, UnPKT predictively protects the packet bits of each transmission and completely eliminates most retransmission overhead. Thus, UnPKT can better utilize the channel bandwidth and achieve higher throughput.

**BER Estimation.** UnPKT needs to estimate the packet bit BERs from CSI. There have been some recent works tailored for measuring the packet bit BER in wide-band channels. Error Estimation Code (EEC) [21] adopts an advanced sampling technique, but it can only estimate the average BER of a packet. SoftPHY [22] can estimate the bit-wise BER using the PHY-layer hints, but it requires special hardware support, namely soft-decoder, which cannot be directly applied to the existing 802.11 NICs. Recitation [8] is compatible to 802.11 framework and proposes the EVP metric to capture the likelihood of error event during decoding. EVP cannot tell BERs of packet bits and thus cannot indicate the required level of protection to them. Han *et. al.* [7] have recently observed the periodical packet bit BER distribution through measurements, but they do not explore this phenomenon to augment the 802.11 transmissions.

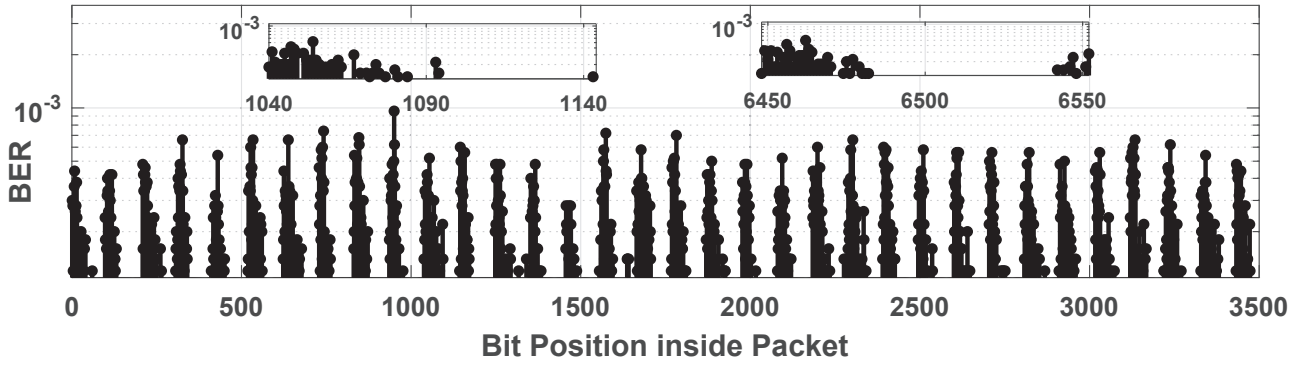


Fig. 2. Measured BER of each bit position within a 1000-byte packet. We depict the first 3500 packet bits for the sake of a clear presentation.

## III. Motivation and Observations

### A. Measuring packet bit BERs

We investigate wireless links in all three test-beds described in Section V and measure the packet bit BERs over lossy links. In theory, the BER distribution should be measured when the link quality is fixed. In practice, the link quality varies all the time. To minimize measurement error, we conduct experiments at night with minimum dynamics in the environment. In addition, we scan the frequency band before each experiment to make sure there is no noticeable interference from other wireless sources. We disable packet retransmission for the measurements. For each link, the sender transmits packets to the receiver indexed by the sequence number. The receiver records both correct and corrupted packets. The payload of each packet is random content of 1000 bytes. For each corrupted packet that fails in the CRC check, we can identify the decoded bit errors referring to the ground truth at the sender side.

**Measurements**. Figure 1(a) plots the PRR (Packet Reception Ratio) of all 166 links in the three test-beds. Due to the data rate uniformity for all the packet bits and the limited rate choices in the 802.11 standards, it is unlikely that one selected data rate can perfectly match the frequency selective channel quality [23]. To maximize the throughput, transmissions are usually over marginal links, e.g., $70\% <$ PRR $< 98\%$, and transmission failures are inevitable [24]. The performance is thus limited by the retransmission overhead of corrupted packets. Figure 1(b) depicts the CDF of all the PRR values. From the figure, we observe that about 50% works on the marginal links with the PRRs. If the PRR of a link is sufficiently low, e.g., smaller than 70%, a lower data rate needs to be selected to match the channel quality.

### B. Observations on the decoded bit BERs

**Periodic packet bit BER distribution**. For each marginal link, we examine the decoded packet bit BER distribution. Figure 2 plots the packet bit BERs measured from a randomly selected link with convolutional coding rate 5/6 and 64-QAM modulation. Although not shown here, these results generalize for other links as well. The *x*-axis represents the bit position, and for each bit position, the *y*-axis indicates the measured BER over this link. Figure 2 shows that the BER distribution is highly uneven due to frequency selective fading across subcarriers. In addition, the BER distribution of the decoded packet bits have a strong periodic property where the period equals the number of packet bits within one OFDM symbol. The result is consistent with the observations in the literature [8], [7], which is fundamentally different from the BER distribution of flat-fading narrow band channels. Figure 2 implies that given a wide-band channel, the location of decoding errors biases to certain bit positions. We can explicitly protect those error-prone bits with better redundancy to prevent transmission failures and improve the throughput.

In Figure 2, we further zoom in two different periods and observe that the packet bit BERs are distributed similarly within the two windows. As a matter of fact, the BER distribution is similar in any two different periods. To show that, we plot the average BER distribution of decoded packet bits in one OFDM period in Figure 3(a), where the *x*-axis represents each bit position in the period, the *y*-axis is the average of the BERs over bits of the same position in OFDM symbols. The reason of the periodical property is as follows. The packet bits are coded into codeword bits and transmitted with OFDM symbols. The codeword bits in the same position of their own OFDM symbols are interleaved to the same sub-carrier, experiencing the same channel quality. The BER distribution of the codeword bits is thus periodical, leading to the BER of the decoded packet bits of a periodical property.

The above observation reveals that the packet-level BER diversity attributes to the uneven BER distribution in each individual period. As the BER distributes similarly among different periods, we can focus on identifying error-prone bit positions in one period. All error-prone bit positions from other periods are then equivalently obtained. The overhead to analyze the BER distribution in one period is minor compared with the overhead of analyzing the entire packet. In addition, such an overhead is merely determined by the period length, which is oblivious to the packet length.

**Predictable BER distribution within a period**. We find that the BER distribution in each period is predictable, which strongly relates to the density of errors that could occur in the underlying codeword bits.

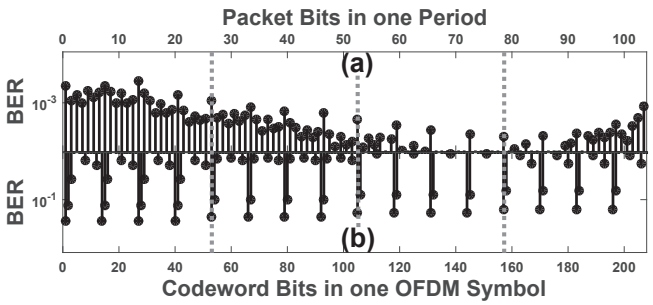


Fig. 3. (a) Average BER of each bit position in one period; (b) BER of the codeword bits in one OFDM symbol.

In Figure 3(b), we plot the BER of each codeword bit in one OFDM symbol. The codeword bit BER values can be calculated using the channel state information (CSI) (see Section IV). As the channel coding rate is $\frac{1}{2}$, one packet bit is encoded into two codeword bits. We align each packet bit with its codeword bits in Figure 3, and observe that the packet bit positions of higher BERs usually correspond to codeword bits of high BERs. For example, within the codeword bit region $[1, 53]$ in Figure 3(b), for every 13 codeword bits, the BERs of three of them are greater than $4.04 \times 10^{-2}$ and the maximal BER equals to $2.56 \times 10^{-1}$. In convolutional codes, ECC (Error Correcting Capability) stands for the maximum consecutive errors that a code can tolerate. For example, EEC equals 4 when the coding rate is $\frac{1}{2}$ [1]. In Figure 3(b), it is easier to have more than 4 (EEC=4) concurrent codeword bit errors in the region $[1, 53]$. The packet bits, within this region, are thus more likely to be erroneous. Essentially, the codeword bit BER distribution is determined by both the frequency selective channel condition (CSI) and the PHY-layer operations, whose diversity leads to the uneven packet bit BERs after decoding.

**Summary**. The 802.11 packet bit BER distribution is not equal, which relates to the occurrence of underlying codeword bit errors. In the next section, we show that the channel CSI determines how likely the codeword bit errors could occur, which can be used to approximately detect the error-prone bit positions in a packet and estimate their BERs. After that, packet bits of different BERs can be protected using the most appropriate redundancies.

## IV. Design

In this section, we describe our unequal packet bit protection design, *i.e.*, UnPKT. In UnPKT, packet bits are protected based on their estimated BERs to prevent transmission failure and improve the throughput. The UnPKT design is encapsulated into a Protection (PTC) layer with a clean abstraction integrated in the existing 802.11 Wi-Fi network stack. The PTC layer is built atop the PHY layer and interacts with the MAC layer. Prior to a packet transmission, PTC layer intercepts the packet from the upper layer and returns the protected packet to MAC layer for transmission. PTC layer is purely software-implementable without any extra hardware support.

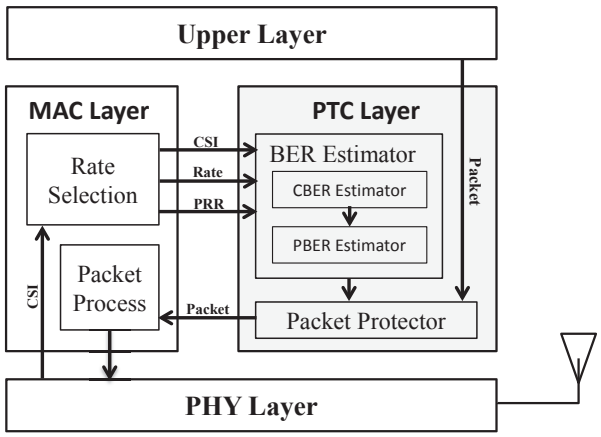


Fig. 4. All the UnPKT operations are abstracted in the PTC layer.

### A. Design Overview

Figure 4 depicts the architecture of UnPKT. At the sender side, the PHY layer measures the channel CSI for the MAC-layer to select a data rate for the next transmission (Section IV-B). The PTC layer takes as input the selected data rate, the estimated PRR, and the measured CSI. If the estimated PRR indicates the next transmission to be over a marginal link, e.g., PRR$< 98\%$, PTC layer provides additional protection for the packet; Otherwise, the packet is transmitted directly. To enable the PTC-layer protection, the codeword BER (CBER) estimator module calculates the BER of each codeword bit given the selected data rate and measured CSI (Section IV-C). The packet bit BER (PBER) estimator then estimates the decoded packet bit BERs (Section IV-D), which are further used by the packet protector to add appropriate redundancies. Afterwards, the protected packet is returned to the MAC layer for transmission. In UnPKT, we also provide the option to augment the data rate if the estimated PRR (without protection) for the current data rate is high, e.g., PRR$> 98\%$, and the estimated PRR (without protection) for the next higher data rate is sufficiently high as well, e.g., PRR$> 70\%$. PTC-layer protection can be performed for the augmented data rate. How the redundancies are added to the packet bits is recorded by a protection field in the packet header with 4-byte fixed overhead (Section IV-D).

Upon receiving a packet, if the packet is correctly decoded, the receiver delivers it to the upper layer; Otherwise, the receiver extracts the protection field information from the header and tries to recover the original packet. If the decoding still fails, the receiver explicitly requests for the retransmission.

### B. Data rate selection

In UnPKT, we apply a state-of-art data rate selection protocol, Recitation [8], on the MAC layer. We adopt its implicit CSI feedback scheme to measure the channel using the reverse ACK packets from the receiver to approximate the forward channel quality. We can then select the data rate and estimate the PRR for the next transmission.

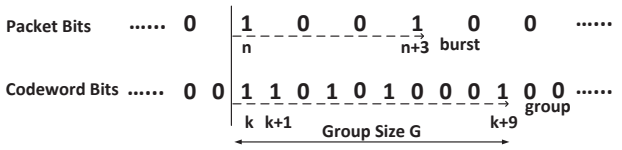


Fig. 5. Illustration of the decoding bit errors and the codeword bit errors.

### C. BER estimator

To transmit a packet, all packet bits $b_n, n = 1 \cdots N$, are scrambled[1] and then encoded into a longer codeword bit sequence $c_k, k = 1 \cdots N'$, where $N$ and $N'$ are the lengths of the packet bits and codeword bits, respectively. To facilitate the discussion, we first focus on a $\frac{1}{2}$ coding rate to introduce the estimator design (i.e., $N' = 2N$), and postpone how the estimator can be applied to all other coding rates until the end of this subsection.

Codeword bits $c_k$ are interleaved and modulated at the sender before transmission. The received codeword bits are demodulated and de-interleaved by the receiver to reassemble the original codeword bits before decoding, whereas errors could occur in the reassembled codeword bits, which are denoted as $\widetilde{c}_k$. The BER estimator is composed of two parts, the codeword bit BER estimator (CBER) and the packet bit BER estimator (PBER), where the CBER estimator first estimates the BERs of the codeword bits $\widetilde{c}_k$, denoted as $e_k, k = 1 \cdots N'$. The PBER estimator further estimates the BERs of the decoded packet bits, denoted as $p_n, n = 1 \cdots N$, taking each $e_k$ as input.

*1) CBER estimator:* The 802.11 standard employs a block interleaver of the size equal to the number of codeword bits in one OFDM symbol [1]. The goal of interleaving is to randomize the codeword bit order during the transmission such that long runs of low reliable codeword bits can be avoided. For any codeword bit $c_k$, the interleaver will interleave it to the position $k_2 = s \times \lfloor \frac{k_1}{B} \rfloor + (k_1 + B - \lfloor 13 \times \frac{k_1}{B} \rfloor) \bmod s$, where $s = max\{1, Q/2\}$, Q is the size of one constellation point, $B$ is the number of codeword bits contained in one OFDM symbol, and $k_1 = \frac{B}{13} \times (k \bmod 13) + \lfloor \frac{k}{13} \rfloor$. After this permutation, the reordered codeword bits are further grouped into four clusters in sequence for the modulation. All the codeword bits in cluster one will be transmitted over sub-carrier $j$ mod 52 in order, where $j = 1, 5, 9, \cdots$, and 52 is the total number of sub-carriers. In general, the codeword bits in cluster $i$ are transmitted over sub-carrier $j$ mod 52, where $j = i, i+4, i+8, \cdots$. Figure 3 illustrates how the codeword bits fall into the four clusters in one OFDM symbol.

The subcarrier over which codeword bits are transmitted is fixed, similar to [8], hence we know the channel quality that each codeword bit experiences during the transmission from the CSI, e.g., the subcarrier SNR. In addition, subcarriers are narrowband in 802.11, e.g., 312.5 kHz in 802.11n. The CBER estimator can employ a classical narrowband SNR-BER relation to estimate codeword bit BERs, $e_k, k = 1 \cdots N'$, which serve as the input of the PBER estimator. In Figure 3(b), we have plotted the codeword bit BERs in one OFDM symbol. The four highlighted regions correspond to the four clusters formed during the modulation. In each region, the four repeated codeword bit BER patterns are because the transmission of the codeword bits in each cluster circulates among 13 (=52/4) fixed sub-carriers.

[1]As the scrambler performs bit-wise XOR between the original packet bits and a scrambling sequence, which is specified by 802.11 standards, knowing the scrambled packet bits is equivalent to knowing the original packet bits. In this paper, we refer to the scrambled packet bits as packet bits for short.

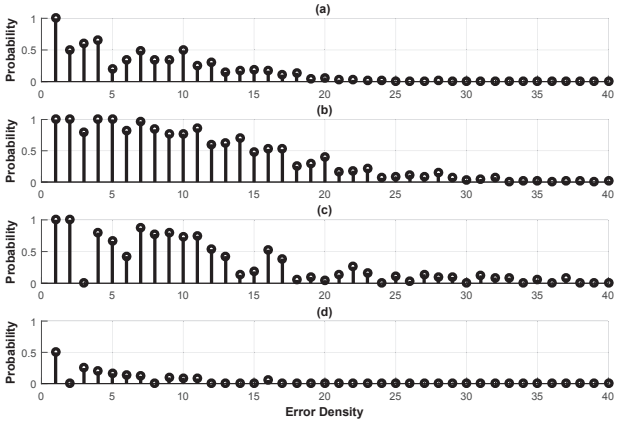


Fig. 6. $P_{group}(k, \eta)$ changes with $\eta$ for the different convolutional coding rates (a) 1/2; (b) 2/3; (c) 3/4; (d) 5/6.

*2) PBER estimator:* As mentioned in Section III, if the errors occur in a group of nearby codeword bits, the error density may exceed the protection capability of the convolutional code and lead to decoding bit errors. Figure 5 depicts such an instance, where "0" and "1" bits indicate the bit's correctness, e.g., they are for correct and erroneous bits, respectively. In general, the BER of packet bit $n$ is composed of two parts:

$$p_n = \sum\nolimits_{\eta} P_{group}(k, \eta) \times P_{fail}(\eta), \tag{1}$$

where $P_{group}(k, \eta)$ denotes the probability to form a codeword bit group with errors starting from codeword bit $c_k$ of error density $\eta$, and $P_{fail}(\eta)$ denotes the probability that such a group could cause decoding bit errors. We now detail the calculation of Eq. (1).

$P_{group}(k, \eta)$ **calculation**. We consider the example in Figure 5 to introduce the calculation of $P_{group}(k, \eta)$. To facilitate the calculation, we define a *codeword bit group* with errors both starts from and ends at codeword bit errors, e.g., codeword bits $[k, k+5]$ and $[k, k+9]$ in Figure 5. We now focus on the latter group $[k, k+9]$. In this group, $\eta = \frac{n_e}{G} = \frac{5}{10}$, where $n_e$ is the number of errors and $G$ is the group size. Given $n_e$ and $G$, in principle, there are $C_8^3$ different group instances. Except the two errors stay at the beginning and the end of the group, the three $(= n_e - 2)$ remaining errors could occur in the middle eight $(= G - 2)$ codeword bits. Figure 5 depicts one such instance.

From the CBER estimator, we have obtained the codeword bit BER $e_k$. $P_{group}(k, \eta)$ in principle equals to the summation of the probabilities that each of the $C_8^3$ group instances will occur. However, it is computational intensive to enumerate

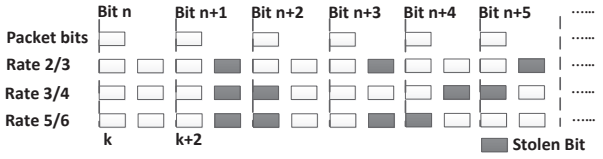


Fig. 7. Puncturing technique to achieve different coding rates in 802.11.

all those combinations. To address this issue, we propose to use the average codeword bit BER ($\overline{e}$) of the eight middle codeword bits, $e_2$ to $e_9$, to simplify the calculation as follows.

$$P_{group}(k, 5/10) = e_k \cdot e_{k+9} \cdot C_8^3 \cdot \overline{e}^3,$$

where $\overline{e} = (\sum_{x=k+1}^{k+8} e_x)/8$ and $e_k$ is the BER of codeword bit $\widetilde{c}_k$. For $\eta$ equals to $\frac{n_e}{G}$ in general, $P_{group}(k, \eta)$ is calculated by the following equation:

$$P_{group}(k, n_e/G) = e_k \cdot e_{k+L-1} \cdot C_{G-2}^{n_e-2} \cdot \overline{e}^{n_e-2}, \qquad (2)$$

where $\overline{e} = (\sum_{x=k+1}^{k+L-2} e_x)/(L-2)$.

$P_{fail}(\eta)$ **calculation**. Given an arbitrary codeword bit group of size $G$ with $n_e$ errors, the decoding bit errors occur only when $\eta$ is sufficiently high. Prior studies [25] have found that decoding bit errors occur mainly when $n_e$ of a group equals to EEC + 1, where EEC is the protection capability of the convolutional code. EEC $= 4, 3, 2, 1$ for the convolutional coding rates $\frac{1}{2}$, $\frac{2}{3}$, $\frac{3}{4}$, and $\frac{4}{5}$, respectively. Thus, we focus on $\eta = \frac{n_e}{G} = \frac{\text{EEC}+1}{G}$ in the $P_{fail}(\eta)$ calculation. Given the group error density $\eta$, there are $C_{G-2}^{n_e-2}$ different group instances. Although not all the instances lead to decoding bit errors, the probability that a group instance could cause decoding bit errors can be off-line tested as shown in Figure 6. By doing so, $P_{fail}(\eta)$ can be directly obtained from Figure 6.

So far, we have obtained $P_{group}(k, \eta)$ and $P_{fail}(\eta)$. Then $p_n$ is simply the summation of $P_{group}(k, \eta) \times P_{fail}(\eta)$ for all possible $\eta$s. According to Figure 6, we find that when $\eta$ is sufficiently small, $P_{fail}(\eta)$ is close to zero. Hence only a small number of $\eta$s are involved in the $p_n$ calculation.

On the other hand, as the packet bit $n$ is encoded into two codeword bits, as shown in Figure 5, $c_k$ and $c_{k+1}$, the decoding bit error at $n$ can also be caused by the codeword bit groups starting from $c_{k+1}$. As a result, $p_n = \sum_\eta P_{group}(k, \eta) \times P_{fail}(\eta) + \sum_\eta P_{group}(k+1, \eta) \times P_{fail}(\eta)$.

$p_n$ **for other coding rates**. We now extend the $p_n$ calculation to all other coding rates. According to the 802.11 standard, all other coding rates, $\frac{2}{3}$, $\frac{3}{4}$, and $\frac{5}{6}$, are implemented based on the $\frac{1}{2}$ rate, using a puncturing technique as shown in Figure 7. For example, to achieve the $\frac{2}{3}$ coding rate, i.e., two packet bits are encoded into three codeword bits, two packet bits (e.g., $b_n$ and $b_{n+1}$) are first encoded into four codeword bits, but the last codeword bit will not be transmitted, which we referred to as a *stolen bit*. The positions of the stolen bits are periodic, and specified in the 802.11 standard. To calculate $p_n$, if one of its encoded codeword bit is a stolen bit, we simply ignore it in the calculation. For instance, $p_{n+1} = \sum_\eta P_{group}(k+2, \eta) \times P_{fail}(\eta)$.

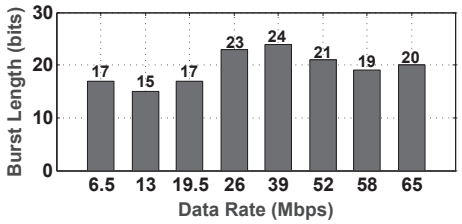


Fig. 8. Burst length (bits) under different data rates.

**Decoding error burst**. We can now calculate the BER of any packet bit $n$. However, as introduced in [26], a decoding failure of the convolutional code usually results in a set of packet bit errors, also known as error bursts. For instance, the burst in Figure 5 contains the errors in positions $n$ and $n+3$. So far, $p_n$ only calculates the probability when the error burst starts from packet bit $n$. $p_n$ should also contain the probability that bit $n$ is an error, but the error burst starts from prior packet bits. The statistical error burst size has been well studied. Existing works [26] found that the error burst length follows the exponential distribution and the distribution parameters are determined by the average BER of the codeword bits. Considering the error burst effect, we compute the excepted error burst size $l$. After calculating $p_n$ in Eq. (1), we add $p_n$ to the BERs of the following $l$ packet bits. Thus, for each packet bit, the output of the PBER estimator is the summation of the BER calculated from Eq. (1) and the BERs of the previous $l$ packet bits.

### *D. Packet protector*

The packet protector module provides unequal packet bit protection according to the BERs. A lightweight *cluster-based* method is used to provide unequal but appropriate redundancies to different packet segments.

**Cluster-based protection.** The interleaving operation of 802.11n standard divides coded bits in OFDM symbol into four consecutive clusters. Coded bits in different clusters are mapped to different sets of non-overlapping subcarriers. Such a process repeats $\kappa$ times when $2^\kappa$-QAM is used. As a result, coded bits in each cluster tend to have similar BERs. Convolutional coding induces error bursts and thus correlates BERs of nearby packet bits, so we can group sequential packet bits into four or $4 \times \kappa$ clusters and provide unequal protection based on their BERs. UnPKT sets the cluster number to four for all modulation choices and our experimental results in Section V demonstrate that it achieves good balance between the computational overhead and accuracy in estimation of redundancy requirement.

**RS (Reed-Solomon) code.** UnPKT employs RS code to provide protection for packet bits. RS code is efficient at correcting burst errors [4], which are common for the decoded packet bits after convolutional code. In RS code, one codeword consists of $u$ RS symbols and each symbol consists of $v$ bits. The $u$ data symbols are composed of $u - w$ data symbols and $w$ symbols as redundancy. As a convention, the RS code can be denoted as $RS(u, w)$. Any $RS(u, w)$ can correct up to $w/2$ symbol errors, which occurs when any number of bits in this

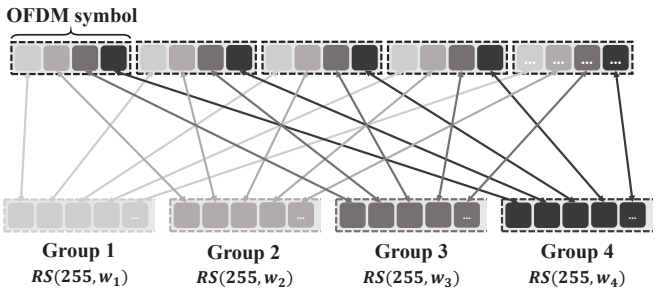


Fig. 9. Clusters from multiple OFDM symbols are grouped together for BER estimation and protected using RS code.

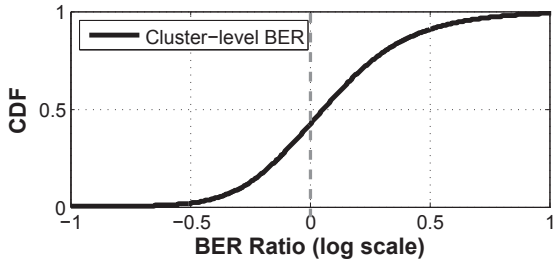


Fig. 10. Cluster-level BER ratio between estimations and measurements.

symbol get wrong. UnPKT adopts the $RS(255, w)$ where each symbol consists of eight bits. This type of RS code has efficient software implementation and is widely used as 802.11 MAC-layer FEC [19], [18], [20], [27]. Figure 8 depicts bit error burst statistics from the corrupted packets measured in Section III. The results show that only three to four $RS(255, w)$ symbols of redundancy are required to correct one error burst as the length of such bursts only last for 15 to 24 bits on average.

**Encoding and decoding.** In UnPKT, the packet protector groups clusters from the same portion across different OFDM symbols together and protects them using an RS code as shown in Figure 9. As a result, the clusters in one group manifest similar BERs. UnPKT uses the average BER of each group to calculate the redundancies required according to theory [4]. Note that the packet bits are not physically moved to form the groups of clusters. After determining the optimal $w$, the sender appends the RS parity symbols sequentially for each group and at the end of the packet. The value of $w$ is encoded into the MAC header (a four byte overhead).

A simpler alternative of protecting packet bits is to encode the entire packet into a single RS codeword and estimate the redundancy based on the average BER of a packet. Using merely a packet level BER and ignoring the bit distribution, however, leads to inaccurate protection, *e.g.*, the distribution of eight bit errors in 1 RS symbol or in eight RS symbols requires very different level of protection, albeit they have the same average BERs. On the other hand, using longer codeword significantly increases the encoding/decoding complexity of RS code. If we encode the entire packet into one RS codeword, the computation it takes will be 64 times what of using $RS(255, w)$ [28].

Upon receiving the packet, the receiver first separates the redundancies from the payload (using the payload length and value of redundancy number $w$ in the header) and then performs error recovery with the redundancy if the packet fails the CRC check. When the error recovery succeeds, the decoded packet is passed to the upper layer. Otherwise, the receiver explicitly requests a retransmission.

## V. Evaluation

In this section, we first use our test-beds to experimentally evaluate our BER estimator, which is the prerequisite that UnPKT can perform well. We then compare UnPKT with the state-of-the-art approaches using extensive trace-driven evaluations.

**Test-beds.** We use Atheros 9580 NICs that support 802.11n 20/40MHz channelization and operate in both 2.4GHz and 5GHz frequency band. The Wi-Fi NIC is configured to report CSI value for every non-empty subcarrier, *i.e.*, 56/114 subcarrier using 20/40MHz channels. Other information associated with the received packet, including the payload, RSSI, data rate and time-stamp, is recorded together with reported CSI [29][2]. We have developed and released an open-source toolkit that works with Ubuntu system [30].

We deploy Atheros 9580 nodes in three different test-beds in campus—an indoor office with 16 experimental locations, a parking lot surrounded by the cars and stores with nine experimental locations, and an open lecture hall with five experimental locations. The three test-beds are typical indoor 802.11 network environments with different degrees of frequency selective fading.

### A. BER estimation evaluation

In Section III-A, we maintained a stable experimental environment to measure the packet bit BERs over each marginal link in the three test-beds. Corrupted packets are collected from 76 marginal links and the payload of each packet is 1000-bytes. We hence have 608,000 ($= 76 \times 1000 \times 8$) bit-wise BER estimations. As the packet protection of UnPKT is performed in the granularity of clusters, for each corrupted packet, we calculate the BER of each cluster, i.e., the average BER of all the bits in the same cluster. We then calculate the ratio between the estimated BER and the measured BER. Figure 10 plots the CDF of the BER ratio. The optimal estimation result yields to the ratio always being one, and so we see that the estimation of UnPKT in general is accurate. According to the statistics, we observe that about 80% and 50% of BER estimations are within 0.5 and 0.25 of one order of the magnitude compared with the BER measurements.

### B. Trace-driven evaluation

In this subsection, we evaluate the overall performance of UnPKT using trace-driven simulations and compare it with state-of-the-art approaches.

[2]We develop the 802.11n CSI tool, instead of using the existing one based on Intel 5300 NICs [5], as corrupted packets are not accessible with Intel 5300 NICs, which are however useful for the evaluation in this paper.

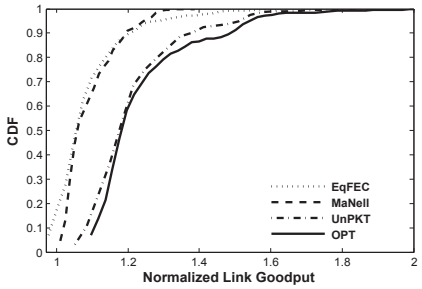

Fig. 11. CDF of the normalized per-link goodput.

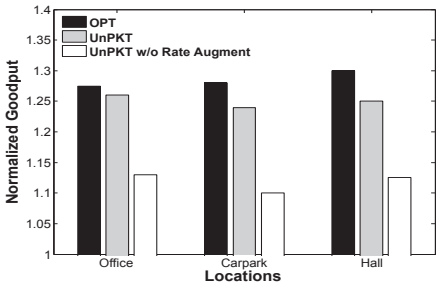

Fig. 12. Goodput gain breakdown of UnPKT.

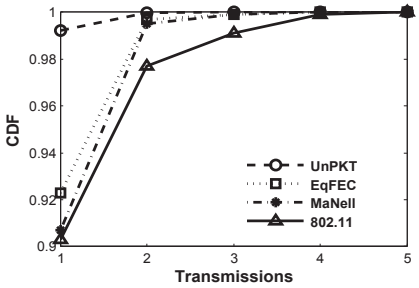

Fig. 13. CDF of transmissions to delivery a packet.

*1) Setup:* The simulator is built on the GNU Radio platform based on the 802.11n PHY-layer specification, including the convolutional code, block interleaver, and OFDM modulation. We implement the convolutional coding rates from $\frac{1}{2}$ to $\frac{5}{6}$, and modulations from BPSK to QAM-64. For the data rate selection, we implement the most recent scheme ESNR [5]. We also implement the RS code encoder and decoder. The link quality between each pair of transceiver is directly from the CSI measurement of all the 166 links in our three test-beds collected in Section III. Each CSI contains 52 sub-carriers, serves as the ground truth for the link quality, and is feeded to the simulator. We include the non-marginal links in the evaluation because we implement the full version of UnPKT which can augment the data rate (Section IV-A). In the evaluation, the sender transmits packets to the receiver and we measure the goodput achieved over each link.

*2) Approaches for comparison:* In addition to UnPKT, we also implement the following approaches for comparison.

**802.11**. The default 802.11 transmissions, which retransmits at most 7 times after the original transmission fails.

**EqFEC**. EqFEC [16] empirically adds a MAC-layer FEC to protect packets in narrow-band channels. For a fair comparison, we provide the packet-level BER to EqFEC. We also allow EqFEC to augment data rate the same as UnPKT.

**MaNell**. MaNell is short for Maranello in [18], which is a partial packet recovery approach with the best reported performance. Therefore, we do not compare UnPKT with other partial packet recovery approaches, e.g., ZipTx [19], explicitly. MaNell divides a packet into blocks and only retransmits erroneous blocks after a transmission fails.

**OPT**. OPT adds the most appropriate MAC-layer FEC to each packet and completely avoids transmission failure.

### C. Experimental Results

**Goodput gain**. Figure 11 examines the per-link goodput gains of EqFEC, MaNell, UnPKT, and OPT, normalized to the goodput achieved by 802.11 in the three test-beds. As the packet-level BER cannot fully represent the diverse packet bit BER distribution in the wide-band transmissions, the goodput gain of EqFEC is only 9.5% on average. In particular, it may perform slightly worse than 802.11, if the redundancies cannot recover the corrupted packets, especially when the data rate is augmented, while it introduces communication overhead to each packet. MaNell achieves 11.2% median and 40% maximal goodput gains over 802.11. The goodput of MaNell is limited mainly because the number of retransmissions in MaNell is still high. Benefiting from the appropriate unequal packet bit protection, the performance of UnPKT is within 4% of OPT. According to statistics, UnPKT outperforms 802.11, EqFEC, MaNell by 23.4%, 13.9%, and 12.2% on average, and 200%, 60%, and 49% at most.

**Goodput gain analysis**. We first analyze the goodput gain achieved by UnPKT, which is from the transmission failure avoidance over both the original and the augmented data rates. Figure 12 depicts the breakdown of the goodput gain. Transmission failure incurs the channel contention delay, packet retransmission delay, ACK feedback delay, etc. In addition, retransmissions usually adopt lower data rates. From Figure 12, we see that the transmission failure avoidance over the original data rate accounts for 58% of the goodput gain. As not all the data rates with high PRRs are augmented (Section IV-A), the goodput gain from the data rate augmentation is 42%. After the data rate is augmented, packets will be transmitted over the marginal links again. UnPKT can still prevent transmission failure and harness the extra goodput from the higher data rate.

To understand the goodput gains achieved by different approaches, we plot the average number of transmissions needed to delivery one packet from the sender to the receiver in Figure 13. From the result, we see that about 10% of packets need retransmissions in 802.11 and MaNell. However, the distribution in 802.11 suffers from a long tail, which leads to significant retransmission overhead to decrease the goodput. As the packet-level BER cannot precisely guide the amount of added FEC, the reduction in retransmissions is only slight. EqFEC cannot well prevent transmission failures, especially when the data rate is augmented. In UnPKT, the unequal packet protection bits can be properly supplemented. As a result, only about 1% of packets needs retransmissions.

In Figure 14(a), we further plot the percentages of the data rates used by different approaches in the evaluation. The *x*-axis represents all the single-stream data rates in 802.11n. As 802.11 and MaNell do not augment data rates and we adopt the same rate selection scheme for the four approaches, their rate selection choices are identical. Similarly, EqFEC and UnPKT have the same the rate selection. From the figure, we see that more rate selections are given to the five highest data rates in EqFEC and UnPKT, which lead to potential higher goodput. As UnPKT precisely protects error-prone bits and

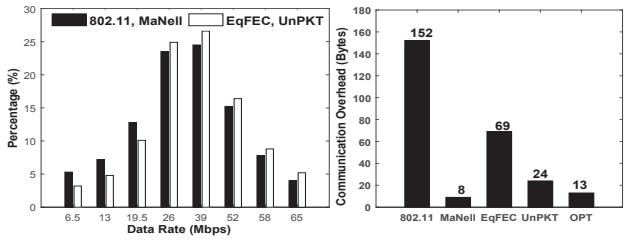


Fig. 14. (a) Fractions of data bit-rates used in different approaches; (b) Communication overhead of each approach.

prevents transmission failures, it can fully harness this goodput improvement opportunity.

**Communication overhead**. In Figure 14(b), we further analyze the communication overhead of each approach, which is measured by the extra error correction bits needed to deliver one 1000-byte packet. In 802.11, when a transmission fails, the sender retransmits the entire packet. As a result, the communication overhead is as high as 152 bytes in our evaluation. As the MAC-layer protection of EqFEC can prevent some transmission failures, the communication overhead of EqFEC is smaller than 802.11. However, EqFEC still suffers from non-negligible retransmissions, leading to high communication overhead. MaNell has the smallest communication overhead. This is because when a transmission fails, MaNell only retransmits the blocks containing error bits. However, MaNell does not reduce the number of retransmissions needed. Hence, its goodput is still limited. OPT adds the most appropriate protection to each packet and completely prevents the transmission failure. OPT thus also introduces communication overhead. Due to the accurate BER estimation, UnPKT has only slight communication overhead, which is close to OPT.

## VI. Conclusion

This paper proposes an unequal packet bit protection approach for 802.11 WLANs. The major design challenge lies in the difficulty to estimate the uneven packet bit BER distribution from the frequency selective channel cross the entire 802.11 PHY layer. In this paper, we observe that packet bit BERs strongly relate to the probability that dense errors occur in the codeword bits and the density of the codeword bit errors. With this observation, we propose UnPKT. UnPKT protects the packet bits using different MAC-layer FEC redundancies based on the bit-wise BER estimation to augment wide-band 802.11 transmissions. We extensively evaluate the performance of UnPKT using both Atheros 9580 NICs and the GNU Radio platform and obtain a significant experimental performance improvement over state-of-the-art approaches.

## Acknowledgement

The research leading to these results has received funding from Singapore MOE AcRF grant MOE2012-T2-1-070, MOE2013-T1-002-005, NTU NAP grant M4080738.020, City University of Hong Kong (Project No. 7200480/CS) and NSFC 61303233, the European Research Council under the EUs Seventh Framework Programme (FP/2007-2013)/ERC Grant Agreement NO. 279976.

## References

[1] *IEEE 802.11n standard*, 2009.
[2] M. Arslan, K. Pelechrinis, I. Broustis, S. Singh, S. Krishnamurthy, S. Addepalli, and K. Papagiannaki, "Acorn: An auto-configuration framework for 802.11n wlans," *IEEE/ACM Transactions on Networking*, vol. 21, no. 3, pp. 896–909, 2013.
[3] D. Gong and Y. Yang, "AP association in 802.11n WLANs with heterogeneous clients," in *IEEE INFOCOM*, 2012.
[4] J. G. Proakis, *Digital Communications*, 2007.
[5] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Predictable 802.11 packet delivery from wireless channel measurements," in *ACM SIGCOMM*, 2010.
[6] A. Bhartia, Y.-C. Chen, S. Rallapalli, and L. Qiu, "Harnessing frequency diversity in Wi-Fi networks," in *ACM MobiCom*, 2011.
[7] B. Han, L. Ji, S. Lee, B. Bhattacharjee, and R. R. Miller, "All bits are not equal-a study of IEEE 802.11 communication bit errors," in *IEEE INFOCOM*, 2009.
[8] Z. Li, Y. Xie, M. Li, and K. Jamieson, "Recitation: Rehearsing wireless packet reception in software," in *ACM MobiCom*, 2015.
[9] W.-L. Shen, Y.-C. Tung, K.-C. Lee, K. C.-J. Lin, S. Gollakota, D. Katabi, and M.-S. Chen, "Rate adaptation for 802.11 multiuser mimo networks," in *ACM MobiCom*, 2012.
[10] M. O. Khan, L. Qiu, A. Bhartia, and K. C.-J. Lin, "Smart retransmission and rate adaptation in wifi," in *IEEE ICNP*, 2015.
[11] H. Rahul, F. Edalat, D. Katabi, and C. G. Sodini, "Frequency-aware rate adaptation and MAC protocols," in *ACM MobiCom*, 2009.
[12] A.J.Viterbi and J.K.Omura, *Principles of digital communication and coding*, 1979.
[13] S. Sen, S. Gilani, S. Srinath, S. Schmitt, and S. Banerjee, "Design and implementation of an "approximate" communication system for wireless media applications," in *ACM SIGCOMM*, 2010.
[14] C.-M. Chen, C.-W. Lin, and Y.-C. Chen, "Unequal error protection for video streaming over wireless LANs using content-aware packet retry limit," in *IEEE ICME*, 2006.
[15] M. Khan, A. Moinuddin, E. Khan, and M. Ghanbari, "Optimized cross-layer unequal error protection for wireless video communication," in *IEEE TENCON*, 2013.
[16] S. Choi, Y. Choi, and I. Lee, "Ieee 802.11 mac-level fec scheme with retransmission combining," *IEEE Transactions on Wireless Communications*, vol. 5, no. 1, pp. 203–211, 2006.
[17] T.K.Moon, *Error Correction Coding*, 2005.
[18] 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 *USENIX NSDI*, 2010.
[19] K. C.-J. Lin, N. Kushman, and D. Katabi, "ZipTx: Harnessing partial packets in 802.11 networks," in *ACM MobiCom*, 2008.
[20] J. Xie, W. Hu, and Z. Zhang, "Revisiting partial packet recovery in 802.11 wireless LANs," in *ACM MobiSys*, 2011.
[21] B. Chen, Z. Zhou, Y. Zhao, and H. Yu, "Efficient error estimating coding: Feasibility and applications," in *ACM SIGCOMM*, 2010.
[22] M. Vutukuru, H. Balakrishnan, and K. Jamieson, "Cross-layer wireless bit rate adaptation," in *ACM SIGCOMM*, 2009.
[23] W. Dong, Y. Liu, Y. He, and T. Zhu, "Measurement and analysis on the packet delivery performance in a large scale sensor network," in *IEEE INFOCOM*, 2013.
[24] P.-J. Wan, B. Xu, O. Frieder, S. Ji, B. Wang, and X. Xu, "Capacity maximization in wireless MIMO networks with receiver-side interference suppression," in *ACM MobiHoc*, 2014.
[25] S. Nanda and K. REge, "Frame error rates for convolutional codes on fading channels and the concept of effective eb/n0," *IEEE Transactions on Vehicular Technology*, vol. 47, no. 4, pp. 1245–1250, 1998.
[26] A. Franchi and R. A. Harris, "On the error burst properties of viterbi decoding," in *IEEE ICC*, 1993.
[27] W. Dong, J. Yu, and X. Liu, "CARE: Corruption-aware retransmission with adaptive coding for the low-power wireless," in *IEEE ICNP*, 2015.
[28] N. Chen and Z. Yan, "Complexity analysis of reed-solomon decoding over gf(2m) without using syndromes," *EURASIP Journal on Wireless Communications and Networking*, vol. 2008, pp. 16:1–16:11, 2008.
[29] Y. Xie, Z. Li, and M. Li, "Precise power delay profiling with commodity wifi," in *ACM MobiCom*, 2015.
[30] "Atheros CSI tool," http://pdcc.ntu.edu.sg/wands/Atheros/.