Are All Bits Equal?—Experimental Study of IEEE 802.11 Communication Bit Errors

Bo Han, Lusheng Ji, *Senior Member, IEEE*, Seungjoon Lee, Bobby Bhattacharjee, and Robert R. Miller

***Abstract*—Recently, practical subframe-level schemes, such as frame combining and partial packet recovery, have been proposed for combating wireless transmission errors. These approaches depend heavily on the bit error behavior of wireless data transmissions, which is overlooked in the literature. We study the characteristics of subframe bit errors and their location distribution by conducting extensive experiments on several IEEE 802.11 WLAN testbeds. Our measurement results identify three bit error patterns: slope-line, saw-line, and finger. Among these three patterns, we have verified that the slope-line and saw-line are present in different physical environments and across various hardware platforms. However, the finger pattern does not appear on some platforms. We discuss our current hypotheses for the reasons behind these bit error patterns and how identifying these patterns may help improve the robustness of WLAN transmissions. We believe that identifiable bit error patterns can potentially introduce new opportunities in channel coding, network coding, forward error correction (FEC), and frame combining.**

***Index Terms*—Bit error patterns, calibration, IEEE 802.11, measurement study, subframe bit errors.**

I. Introduction

In modern digital wireless communications, a transmitter maps data bits into states of information bearers, i.e., frequency, phase, and amplitude, of a sinusoidal electromagnetic wave called the carrier. Each block of data bits is modulated onto a segment of the carrier wave with persistent information bearer states. Such a segment of the carrier is called a *symbol*, which is often represented by a complex number.

As transmitter-emitted electromagnetic energy propagates through communication medium and reaches an intended receiver, the receiver demodulates and recovers the original data bits by detecting the states of the information bearers of each symbol. Signal quality is quantitatively described by the signal-to-interference-plus-noise ratio (SINR); environmental factors, such as the background noise power, interfering signal of other simultaneous transmissions, can distort symbols during transmission, propagation, and reception and can reduce SINR. Moreover, these factors also make the decoding of the original data bits more difficult.

Wireless communication signal attenuates much more rapidly over distance compared to wired communication because it uses an open space over the air as its medium. Instead of being contained within the physical boundaries of the wired medium, carrier energy of wireless communication is radiated along all directions. Due to the open nature of wireless communication medium, electromagnetic energy is often reflected, diffracted, and scattered by obstacles in the environment. Interference energy introduced by other nearby transmitters and noise energy present in both surrounding environment and circuitries of transmitters and receivers also affect the received signal quality. As a result, wireless communication is often characterized by its relatively low received signal quality and high variance in quality. Consequently, not only bit errors occur more often, but also the variance in bit error probability in wireless communication systems is high compared to wired communications.

Wireless channel errors have a significant impact on the performance of various protocols [1]–[3]. There are many techniques in wireless communication systems designed to overcome the bit error characteristics. A transmitter may use a modulation scheme with a sparse constellation to reduce the probability that a symbol is mistaken for another. This mistake may lead to data bits carried by this symbol being decoded incorrectly. Wireless communication systems use error detection and correction coding schemes that include redundant information in transmissions to help the receiver recover original data bits. Thus, modern wireless communication systems often support multiple modulation and channel coding schemes for balancing throughput and error correction capability under different channel conditions. Of course, all approaches have their limits and their achievable throughput is still bounded by laws such as the Shannon–Hartley capacity.

Compared to their wired counterparts, WLAN communications have unique transmission error characteristics. In this paper, we present experimental results obtained from a study focusing on WLAN transmission bit errors. We believe that getting a better understanding of such bit error behaviors can potentially introduce new possibilities for improving WLAN transmission robustness.

For systems such as IEEE 802.11 WLANs, each physical (PHY)-layer frame is a self-contained communication information unit with both control and data information. With a commonly known format, all PHY-layer communication parameters are embedded within each frame itself. For instance, there is no side band used for synchronization between a transmitter and a receiver. Such synchronization is achieved by the receiver

Manuscript received May 18, 2011; revised September 19, 2011; accepted December 05, 2011; approved by IEEE/ACM Transactions on Networking Editor K. Papagiannaki. Date of publication November 22, 2012; date of current version December 13, 2012. B. Han and B. Bhattacharjee were supported in part by the NSF under Award NeTS-NBD 0626636. B. Han was also supported in part by the NSF under Award CNS 1010789.

B. Han and B. Bhattacharjee are with the Department of Computer Science, University of Maryland, College Park, MD 20742 USA (e-mail: bohan@cs.umd.edu; bobby@cs.umd.edu).

L. Ji, S. Lee, and R. R. Miller are with AT&T Labs—Research, Florham Park, NJ 07932 USA (e-mail: lji@research.att.com; slee@research.att.com; rrm@research.att.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2012.2225842

receiving a special SYNC field, which is a fixed number of symbols of known contents, at the beginning of each frame. Modulation and coding specification is also embedded in the PHY-layer frame header. This design leads to per-frame error recovery. The acknowledgment-based retransmission mechanism is an integrated part of the IEEE 802.11 standard. Rate adaptation and other frame recovery schemes have also been proposed to improve packet loss resilience and increase the throughput of wireless networks [2], [4], [5].

Recent proposals [3], [6], [7] consider subframe information for error recovery. For example, with frame combining, multiple possibly erroneous receptions of a given frame are combined together to recover the original frame without further retransmissions. Partly motivated by this trend, we began to study the position of erroneous bits within a frame. We believe that repeatable and predictable patterns are helpful for designing subframe-level mechanisms, such as frame combining [2], [6], and may introduce new opportunities in channel coding, network coding [8], and forward error correction (FEC)-based error recovery protocols [7].

For WLAN transmissions, assuming both the transmitter and receiver are stationary, conventional wisdom dictates that bit errors should be independent and identically distributed [9]. This is largely due to the expectation that within frame-transmission duration the channel condition likely remains unchanged. Markov models with finite states are also popular [10], [11]. In addition, the Poisson-distributed bit error model has been used to measure the performance of wireless TCP protocols (e.g., the snoop protocol [12]). Köpke *et al.* [13] propose a chaotic map model that determines its parameters based on measurement data. There are also measurement studies of error characteristics for in-building wireless networks [14], wireless links in industrial environments [15], and urban mesh networks [16].

In order to better understand 802.11 data transmissions, we study the subframe bit error characteristics of 802.11 using a number of different testbeds. Our measurement results have identified that in addition to bit error distributions induced by channel conditions, other bit error probability patterns also exist. We start the experiments on an indoor testbed and observe three bit error patterns from the experimental results: “slope,” “sawtooth,” and “finger.” To ascertain whether the patterns are local to our initial testbed, we repeated our measurements on five different environments. Each show similar patterns. Furthermore, subsets of these patterns exist on different hardware combinations as well.

To the best of our knowledge, this is the first detailed systematic experimental study of subframe bit error characteristics. The contributions of our work are as follows.

- We have performed experiments on IEEE 802.11 WLAN testbeds to study subframe error characteristics and their location distribution.
- We have identified the superposition of three patterns for bit error probabilities with respect to bit position in a frame, namely the slope-line pattern, the saw-line pattern, and the finger pattern.
- We have verified that the first two patterns (i.e., slope-line and saw-line) exist in different physical environments and across different WLAN hardware platforms.

The rest of this paper is organized as follows. We first give a brief introduction of the IEEE 802.11 modulation and channel coding schemes in Section II. In Section III, we describe our testbed construction and experiment configurations. We report our measurement results in Section IV and discuss hypotheses for the reasons behind these bit error patterns in Section V. After reviewing related work in Section VI, we conclude with a discussion of future work in Section VII.

TABLE I
IEEE 802.11 PHY PARAMETERS

Rate (Mbps)	802.11 amendment	Modulation	Coding rate	Data bits / symbol
1	-/DSSS	DBPSK	1	1/11 chips
2	-/DSSS	DQPSK	1	2/11 chips
5.5	b/DSSS	CCK	1	4/8 chips
11	b/DSSS	CCK	1	8/8 chips
6	ag/OFDM	BPSK	1/2	24/OFDM Symbol
9	ag/OFDM	BPSK	3/4	36/OFDM Symbol
12	ag/OFDM	QPSK	1/2	48/OFDM Symbol
18	ag/OFDM	QPSK	3/4	72/OFDM Symbol
24	ag/OFDM	16-QAM	1/2	96/OFDM Symbol
36	ag/OFDM	16-QAM	3/4	144/OFDM Symbol
48	ag/OFDM	64-QAM	2/3	192/OFDM Symbol
54	ag/OFDM	64-QAM	3/4	216/OFDM Symbol

II. IEEE 802.11 Wireless LAN Background

The IEEE 802.11 standard covers both the medium access control (MAC) and PHY layers [17]. For our study, the most important parts of the PHY layer are modulation and channel coding schemes.

The original 802.11 standard defines a direct sequence spread spectrum (DSSS) system operating in the 2.4-GHz frequency band. A number of amendments have greatly expanded WLAN capability by specifying more modulation and coding schemes and more frequency bands. IEEE 802.11b uses DSSS and adds two more PHY-layer bit rates (5.5 and 11 Mb/s). Both IEEE 802.11a and 802.11g are orthogonal frequency-division multiplexing (OFDM) systems. We summarize the various PHY-layer parameters for different variations of the IEEE 802.11 standard in Table I.

In the following, we briefly describe the OFDM PHYs. More detailed information can be found in [17]. Each 802.11 frame begins with a PHY-layer header of a format that is known by all WLAN receivers. The PHY-layer header consists of a physical layer convergence procedure (PLCP) preamble and a PLCP header. The PLCP preamble contains a number of training symbols, which help receivers detect signal, configure gain control, align frequency, and synchronize timing. Time synchronization enables a receiver to determine the boundaries of each symbol. The PLCP header specifies the modulation and coding scheme and the length of a frame.

The data portion of each frame is the result of the PHY-layer encoding process, which is illustrated in Fig. 1. Data bits received from the MAC layer are first scrambled by XORing them with a scrambling sequence. The scrambler is used to randomize the data bits that may contain long sequence of binary 1’s or 0’s. The scrambled data bits are then encoded by a convolutional code with a rate of 1/2. Higher coding rates are achieved by discarding (puncturing) coded bits at certain positions. The scrambled and coded data bits are subsequently interleaved by a two-step permutation. The first permutation is used to map adjacent coded bits onto nonadjacent subcarriers.

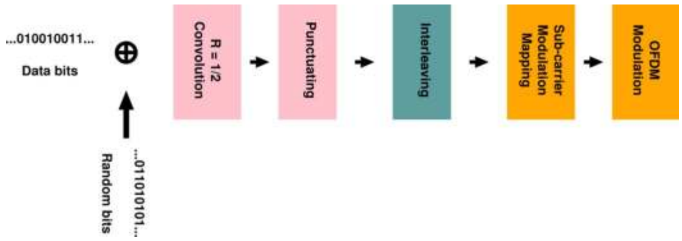

Fig. 1. IEEE 802.11 bit stream encoding process for OFDM modulation.

The second is used to avoid long runs of low reliability bits by mapping adjacent coded bits onto less and more significant bits of a constellation. Finally, the scrambled, encoded, and interleaved data bits are divided into groups with each group converted into a complex number according to the specified modulation scheme for each subcarrier of the OFDM system. Every 48 complex numbers are transformed into one clip of time-domain waveform, called an OFDM symbol, by an inverse fast Fourier transformation (IFFT).

III. Experimental Platform

We describe our experimental platform, including the hardware configuration, RSSI calibration, and experimental procedure.

A. *Hardware Configuration*

We use the same hardware platform for both transmitter and receiver nodes on the primary testbed. Each node is a Soekris Engineering net4826 embedded computer with two mini-PCI type-III sockets for options such as WLAN cards. We primarily use EMP-8602 and DCMA-82 mini PCI cards in our experiments. Both use Atheros AR5006 802.11a/b/g chipsets. We connect the WLAN card on each node to an omnidirectional antenna with 5 dBi (4.8 dBi after cable/connector loss) gain. We use the on-board USB port to dump the received frames to an external storage. Each node runs a Debian Linux distribution with kernel version 2.6.15, and its WLAN operation is supported by the MadWifi v0.9.3 device driver.

B. *RSSI Calibration*

Most WLAN chipsets report the received signal quality using a numerical value called the received signal strength indicator (RSSI) [18], [19]. RSSI is captured through an analog-to-digital converter on the intermediate frequency (IF) level, and we expect that the relationship between RSSI and dBm to be quasi-linear. There is, however, not a standard definition for RSSI, leaving device manufacturers to interpret and implement it differently. We verified that the RSSI reported by the MadWifi driver for Atheros chipsets is a linear-scale representation of the actual received signal power in dBm using an attenuator-based methodology. We calibrated the RSSI values of our WLAN cards with the setup shown in Fig. 2. In this setup, a step attenuator is placed between the receiver and the B port of a PE2031 RF signal splitter to produce different power levels of the received signal.

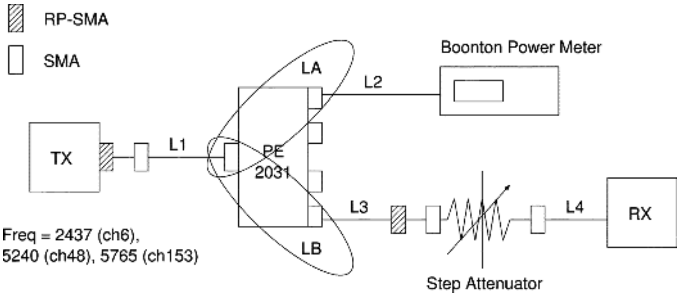

Fig. 2. Calibration setup.

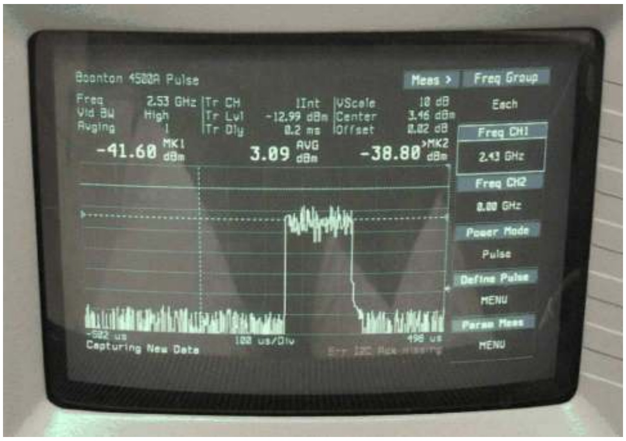

Fig. 3. Boonton 4400 power meter display.

With this setup, after the attenuation of all individual components is measured, the signal strength at the receiver S_{RX} can be calculated as

$$S_{\mathrm{RX}} = S_{\mathrm{PM}} + L2 + LA - LB - L3 - LS - L4$$

where Li is cable i's attenuation, LA and LB are the attenuations of splitter ports A and B, respectively, LS is the attenuation of the step attenuator, and S_{PM} is the power meter reading. During the calibration process, a WLAN transmitter periodically transmits data frames of the same length and contents on channel 6 (2.437 GHz). The transmissions are received by both the power meter and the WLAN receiver. Fig. 3 shows the screen of the Boonton 4400 RF Peak Power Meter[1] displaying a captured WLAN frame at 54 Mb/s bit rate. The received signal power at the WLAN receiver can then be calculated and compared with the RSSI value reported by the same WLAN card. The step attenuator is used to add series of different attenuations before the signal reaches the receiver, as a way of controlling different received signal power. Fig. 4 plots a typical calibration result that indicates that for our WLAN cards, RSSI has a *linear* relationship with the received signal power in dBm.

C. *Experimental Procedure*

During the experiments, we configure one node to be the transmitter and a number of nodes as the receivers. The EMP-8602 and DCMA-82 cards have two antenna ports, and we connect only one of them to the external antenna. We disable

[1] http://www.boonton.com

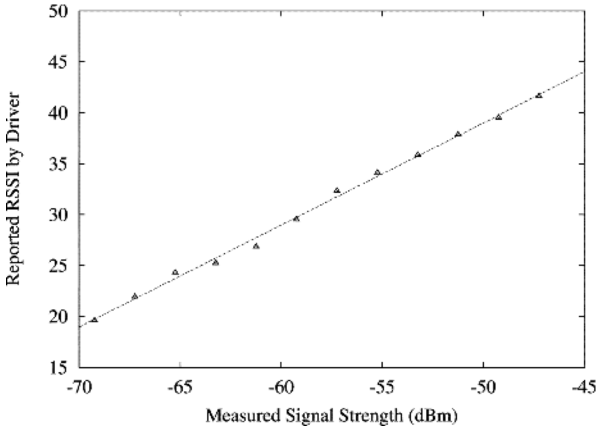

Fig. 4. RSSI to received signal power mapping. The slope of the fitting line is 1.002 with 95% confidence bounds (0.96, 1.044).

antenna diversity on both transmitter and receiver nodes to avoid signal quality variation caused by either end switching to a different antenna port. The transmitter continuously sends 1024-B-long UDP packets every 10 ms. Within each data packet, we reserve the first 4 data bytes as a sequence number to match received frames with originally transmitted frames. We put the receivers under "monitor" mode and configure them to pass all data frames received from the transmitter, regardless of their error status, to user space. The received frames are compared to the original frames to locate at what bit positions they differ.

It is worth noting that the MAC header and our data sequence number field are not immune to transmission errors, which may cause mismatching between a transmitted frame and a received frame, or discarding/accepting frames mistakenly. Such errors are identified in our experiments if possible or otherwise ignored. This type of error involves a relatively small number of bits, reducing the probability of observing such events.

We mostly use data packets with all data bytes set to 0x00. The PHY layer uses a scrambler to randomize the data, and we do not expect the contents of data packets to have significant impact on the experimental results. We also used data contents of all bytes set to 0xFF (all 1's), 0x55 (alternating 0's and 1's), random values, and real traces collected in an office environment. We present the experimental results using real traces in Section IV-H. We only study bit errors in UDP payload (not including the first 4-B sequence number). In each experiment, the transmitter sends out 100 000 identical packets unless stated otherwise.

Our primary testbed consists of six nodes deployed along a hallway of an office building, as illustrated in Fig. 5. Node 1 is configured as the transmitter, and the other five nodes are receivers. The transmitter and the first receiver is approximately 12 m apart, and the adjacent receivers are 6 m apart. This particular setup allows us to see how bit errors occur as the same transmission is received by receivers at increasing distance, (equivalently, decreasing signal quality), from the transmitter. Limited by physical space constraints, other testbeds consist of fewer receiver nodes. In these cases, we reduce transmit power or apply an attenuator to emulate attenuation produced by physical distance. All experiments on the primary testbed were performed during the daytime on weekdays with other nearby 802.11 networks operating on the same channel. We will explain the details of these secondary testbeds as we discuss their results.

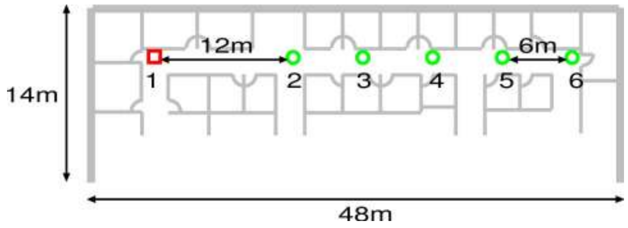

Fig. 5. Primary testbed topology.

We used fixed PHY-layer bit rates for all the experiments and present the results of 54 Mb/s for most of the experiments. As we will show later in Section IV, the peak-to-peak period of saw-line pattern is about the same as the number of bits per OFDM symbol. Using auto-rate could change the OFDM modulation schemes during the experiments, and thus obfuscate the saw-line pattern. Except for the primary testbed, we used only two wireless nodes, a transmitter and a receiver, for all other testbeds.

We point out two limitations of our experiments. First, we could only intercept the received bits at the top of the PHY layer (because in commercial WLAN products the processes in the PHY layer including channel encoding/decoding are concealed within hardware/firmware and not accessible from outside). Thus, we cannot measure all of the over-the-air bits, but only those that pass the channel-decoding procedure. The other is that not all experiments are conducted with the same transmission power. For the testbeds in small enclosed environments, node distances were constrained. We varied transmission power to emulate effects of physical distance.

IV. Experiments and Results

A. Overview

In this section, we first present the three bit error patterns, the slope-line, saw-line, and finger patterns, which we identified on the primary testbed. We then quantitatively model these patterns through curve fitting technology. Finally, we perform more experiments to exclude some possible reasons of these patterns, such as environmental effects and hardware platforms. We repeated the experiments in five other different physical environments—on the Emulab wireless testbed, in a shielded room, over the cable communications, in mobile and outdoor environments—to verify that these patterns are not caused by and unique to our primary testbed. We also repeated the experiments using different hardware platforms and device drivers, as listed in Table II. The experimental results show that the slope-line and saw-line patterns are also present on these hardware platforms. However, the finger pattern exists for only the receivers with Atheros AR5006/AR5212 chipsets.

We have tested not only IEEE 802.11b/g chipsets, but also 802.11n cards. For most of the experiments, we used the open-source device drivers in Linux for various cards. We used the proprietary Linux-based device driver for the Conexant 3894 mini PCI card with a PRISM chipset and the production-level

TABLE II
EXPERIMENT HARDWARE COMBINATIONS (INDICATED BY ∗)

Transmitter	Device Driver	Receiver EMP-8602 AR5006 MadWifi/ath5k	 DCMA-82 AR5006 MadWifi/ath5k	 Intel PRO 2100 ipw2100	 Broadcom BCM4318 b43	 Atheros AR9285 ath9k
EMP-8602 AR5006	MadWifi/ath5k	∗	∗		∗	∗
Intel PRO 2915	ipw2200	∗	∗		∗	∗
ZyXEL ZD1211	Windows driver		∗			
Conexant PRISM	Linux driver		∗	∗		
Agilent E4438C	n/a	∗				
Broadcom BCM4318	b43	∗	∗		∗	∗
TI WL1251	wl12xx	∗	∗		∗	∗

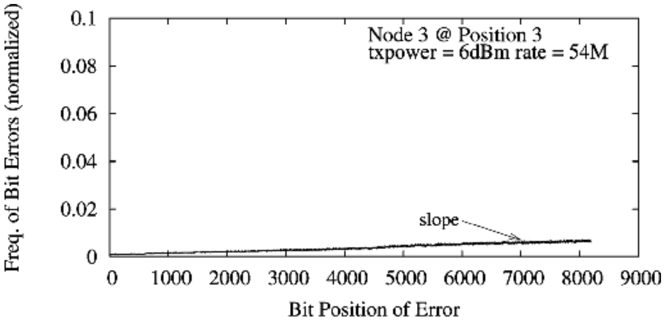

Fig. 6. Normalized bit error frequency, over the total number of received error packets, for node 3; bit rate set to 54 Mb/s.

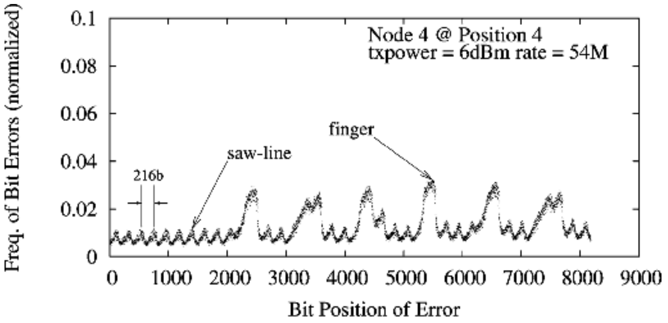

Fig. 7. Normalized bit error frequency for node 4 with bit rate 54 Mb/s. The average RSSIs of correct packets, truncated packets, and packets with bit errors are 36, 21, and 22, respectively.

Windows-based device driver for the ZyXEL AG-225H USB Adapter with a ZyDAS ZD1211 chipset.

B. Bit Error Distribution Patterns

As the received signal quality decreases, the difficulty for a receiver to receive a frame correctly increases. Loosely speaking, incorrectly received frames fall into one of three categories: frames received with bit errors, truncated frames, and completely lost frames. Frames with bit errors usually occur when the received signal quality is marginal. In this case, only some bits within a frame are decoded in error. Although 802.11a/g PHY layer utilizes a convolutional coding scheme for error corrections, once the number and distribution of erroneous bits exceed the coding correction capability, the resultant frame after the PHY-layer decoding will contain error bits. Such errors will likely be caught by the integrity check of MAC layer and cause the frame to be discarded.

During the reception of a frame, if the received signal quality drops so much that the receiver could no longer even detect the carrier, the PHY layer will prematurely exit from reception, which results in a truncated frame. In some cases, a transmitted frame may be completely lost. Various conditions can cause entire frames to be lost. For instance, the receiver may not detect the carrier at all, or it may not be able to lock its clock with the synchronization symbols included in the beginning of the frame, or it may not receive and/or decode the PLCP preamble and PLCP header of the frame.

We have identified a number of unexpected bit error probability patterns from the primary testbed measurements. Fig. 6 is a histogram of where the erroneous bits are located for receiver node 3 on the primary testbed. The x-axis is the bit position within the 1024-B data packets, and the y-axis is the error frequency for each bit position. The y-axis value is normalized over the total number of received error packets. In this experiment, we set the transmission power to 6 dBm and bit rate to 54 Mb/s. The average RSSIs for correct, truncated, and error packets received during this experiment are 37, 28, and 29, respectively. During the experiments, we send out 100 000 packets with all bytes set to 0x00. Among the 100 000 packets, the total number of received packets is 86 119, including 198 truncated packets and 5238 packets with bit errors. We plotted erroneous bits for only packets received with bit errors. Fig. 6 clearly shows that there exists a *linear* relationship, i.e., a slope-line pattern with $\sim 7.4 \times 10^{-7}$ slope, between the frequency of bit errors and their bit positions in a frame. A bit near the end of a frame is more likely to be received in error than a bit near the beginning of the frame. For example, a bit at position 8000 (0.00656) is about three times more likely to be received in error than a bit at position 1000 (0.00161).

We show the same bit error frequency versus bit position plot with the data collected on receiver node 4, which is farther away from the transmitter than node 3, during the same experiment in Fig. 7. This plot exhibits different bit error behavior. While the slope pattern is still present, Fig. 7 also displays two additional patterns: what we refer to as the *saw-line* pattern and the *finger* pattern. The saw-line pattern is the fine zigzag line that goes across the full length of the frame. What is interesting about this pattern is that the sawtooth peak-to-peak period is about the same as the number of bits each OFDM symbol carries at 54 Mb/s bit rate. The finger pattern refers to the larger peaks, which begins to appear after certain bit position (around the 2000th bit) and repeats at a fairly regular interval. The overall plot of bit error frequencies in Fig. 7 is actually the *superposition* of all three patterns.

We also observed similar patterns from the results obtained from nodes 5 and 6. Node 2 is the closest to the transmitter

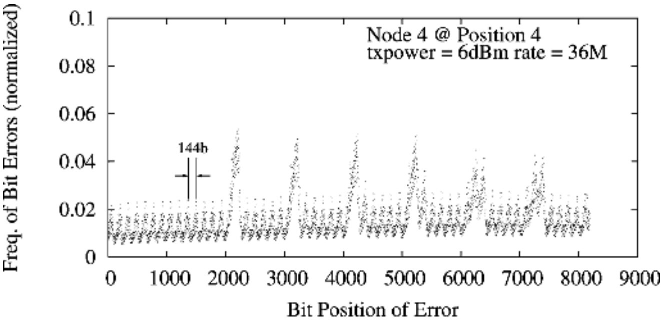

Fig. 8. Normalized bit error frequency for node 4 with bit rate 36 Mb/s. The average RSSIs of correct packets, truncated packets, and packets with bit errors are 34, 19, and 21, respectively.

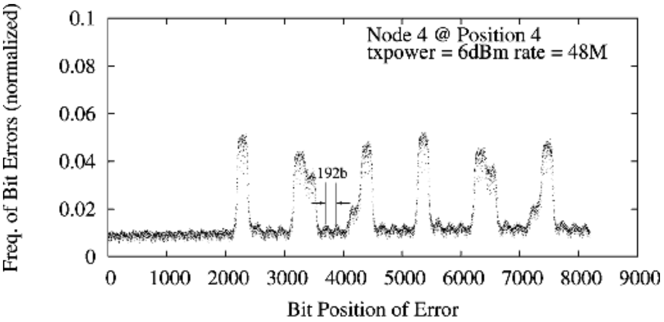

Fig. 9. Normalized bit error frequency for node 4 with bit rate 48 Mb/s. The average RSSIs of correct packets, truncated packets, and packets with bit errors are 35, 22, and 26, respectively.

among all receivers. It has the best received signal quality. We were not able to collect enough frames with erroneous bits to produce any meaningful bit error histogram plots for node 2.

We repeated the experiments with bit rates set to 36 and 48 Mb/s and with different data contents (all bytes set to 0xFF, 0x55, or random value). Due to space limitation, we only show the plots for 36 and 48 Mb/s with all bytes set to 0x00 in Figs. 8 and 9. While we can observe the same three patterns from all these plots, including those with 0xFF, 0x55, and random UDP payload, the peak-to-peak period of saw-line pattern changes for different OFDM bit rates (144 bits for 36 Mb/s and 192 bits for 48 Mb/s).

C. *Quantification of Patterns*

In this section, we further analyze the three patterns identified above by quantitatively modeling the patterns using curve fitting techniques.

As we mentioned above, the bit error patterns are apparently a superposition of slope-line, saw-line, and fingers. We first use a linear function $l(x) = u * x + v$ to fit the slope-line pattern. Because the fingers have high peaks that would affect the fitting result, we calculate the slope parameters using a modified plot by removing all the data points in the finger regions. We then model the saw-line for the first 2000 bits because the fingers only appear after a certain point and within the first 2000 bits there is no finger. Given the periodic nature of saw-line pattern, we use the most common periodic curve fitting function to model it

$$s(x) = a + b * cos(\omega * x) + c * sin(\omega * x) + l(x)$$

where $l(x)$ is the bit errors contributed by the slope line at position x.

TABLE III
SLOPES AND INTERCEPTS OF THE FITTING LINES, AND THE CALCULATED PERIODS OF THE FITTING SAW-LINES

Bit Rate	u	v	ω at 95% confidence	Period
54M	5.1×10^{-7}	7.3×10^{-3}	(0.02906, 0.02917)	215.8
48M	4.5×10^{-7}	8.8×10^{-3}	(0.0325, 0.033)	191.9
36M	6.8×10^{-7}	1.1×10^{-2}	(0.04354, 0.04372)	144.0

TABLE IV
FINGER WIDTH

Bit Rate	54M	48M	36M
Finger 1	648(3x)	775(4.036x)	436(3.028x)
Finger 2	858(3.972x)	768(4x)	436(3.028x)
Finger 3	848(4x)	768(4x)	432(3x)
Finger 4	648(3x)	768(4x)	432(3x)
Finger 5	649(3.005x)	768(4.x)	576(4x)
Finger 6	835(3.87x)	761(3.964x)	576(4x)

We summarize the fitting results for the patterns observed at node 4 for 54 Mb/s (Fig. 7), 48 Mb/s (Fig. 9), and 36 Mb/s (Fig. 8) in Table III. For the saw-line fitting, after we determine the value of ω, we can calculate the sawtooth period as $2 * \pi/\omega$, which is shown in the last column of Table III. The calculated sawtooth periods have verified our earlier observation that the saw-line period is exactly the symbol length for the corresponding bit rate (216 for 54 Mb/s, 192 for 48 Mb/s, and 144 for 36 Mb/s).

Once the bit errors contributed by the slope and saw-line patterns are determined, they can be removed, and all remaining bit errors are considered to be the result of finger pattern. We present the width of the six fingers found in the results for node 4 from all experiments in Table IV. The numbers in the parentheses are the ratio between the finger width and the corresponding symbol length. This table shows that the widths of the fingers are multiples of the corresponding number of data bits per OFDM symbol. We curve-fit the bit error patterns identified on other testbeds; we present results from these testbeds and their curve fits next.

D. *Different Physical Environments*

We have repeated our experiments in five other different environments (Emulab wireless testbed, a shielded room, over the cable communications, mobile and outdoor environments) to verify that the three identified patterns are not the result of the specific environment of our primary testbed.

1) Emulab Wireless Testbed: Although Emulab is often used to provide emulated network environments for experiments of wired networks, the Emulab wireless testbed uses over-the-air communication through IEEE 802.11 wireless interfaces between stationary PC nodes scattered around a typical office building. Each Emulab node has two Netgear WAG311 cards, which use Atheros AR5212 802.11a/b/g chipsets. Fig. 10 shows the result when node pcwf2 is selected as the transmitter and pcwf 13 is used as the receiver,[2] which verifies the three bit error patterns. We note that in this experiment, not only the

[2] The floorplan of the Emulab wireless testbed is available at https://www.emulab.net/floormap.php3.

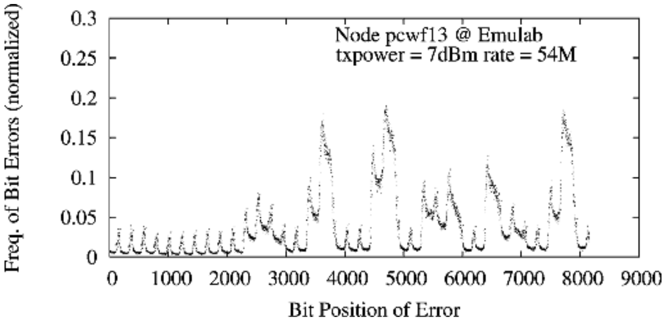

Fig. 10. Normalized bit error frequency for node pcwf13 of Emulab testbed. Node pcwf2 is selected as the transmitter. The slope of the fitting line is 2.553×10^{-6} with 95% confidence bounds (2.354×10^{-6}, 2.751×10^{-6}), and the sawtooth period is 215.917 with 95% confidence bounds (215.473, 216.438).

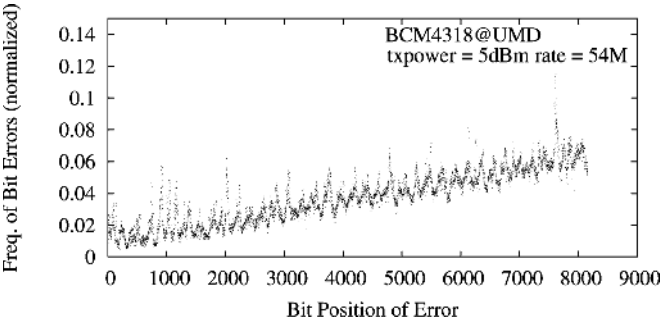

Fig. 11. Normalized bit error frequency for Broadcom BCM4318 to Broadcom BCM4318. The slope of the fitting line is 6.506×10^{-6} with 95% confidence bounds (6.444×10^{-6}, 6.572×10^{-6}), and the sawtooth period is 216.066 with 95% confidence bounds (215.917, 216.140).

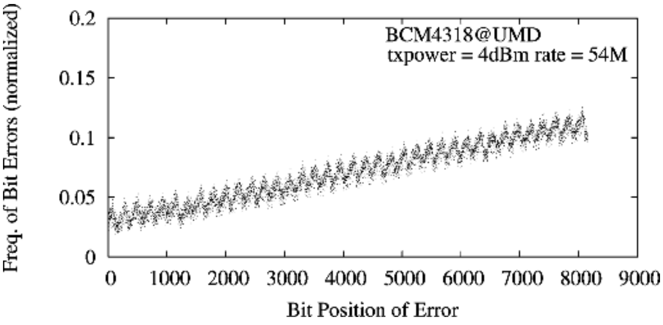

Fig. 12. Normalized bit error frequency for EMP Atheros AR5006 to Broadcom BCM4318. The slope of the fitting line is 1.022×10^{-5} with 95% confidence bounds (1.016×10^{-5}, 1.027×10^{-5}), and the sawtooth period is 215.843 with 95% confidence bounds (215.769, 215.917).

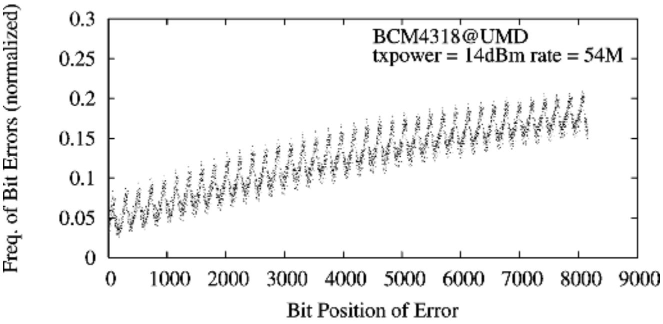

Fig. 13. Normalized bit error frequency for Intel PRO 2915 to Broadcom BCM4318. The slope of the fitting line is 1.638×10^{-5} with 95% confidence bounds (1.624×10^{-5}, 1.653×10^{-5}) and the sawtooth period is 215.769 with 95% confidence bounds (215.769, 215.843).

environment is different, the hardware platform is also different (Atheros AR5212 versus Atheros AR5006).

2) Others: We refer interested readers to Han *et al.* [20] for the results of experiments performed in a shielded room and over the cable. We present the experimental results of the last two challenged mobile and outdoor environments in Section IV-G.

E. Different Hardware Platforms

The experimental results presented so far were all obtained using WLAN cards made of Atheros AR5006/AR5212 chipsets. This raises another question: Do these patterns only occur on specific hardware platforms? In this section, we present experimental results obtained using hardware made by different manufactures with different chipsets.

A problem of using WLAN hardware with non-Atheros chipsets is that their device drivers normally support only a very limited configuration interface. We need to control the bit rate and transmit power for each transmitter and configure receivers to pass up frames with bit errors to user space for processing. These requirements, especially those on receivers, limited our choices to the combinations of transmitter and receiver hardware as listed in Table II.[3] The transmitters are shown in the leftmost column, and the receivers are shown in the top row. So far, we find only three (families of) chipsets that can be used as receivers: Atheros (including AR5006 802.11a/b/g and AR9285 802.11n), Broadcom BCM4306/4318/4320 802.11b/g, and Intel PRO 2100 802.11b.

1) Broadcom Receiver: Benefiting from OpenFWWF [21], an open-source firmware for Broadcom WiFi cards, we can also modify the firmware to make the BCM4318 chipsets pass the corrupted frames to user space. We show the experimental results when Broadcom BCM4318, EMP Atheros AR5006, and Intel PRO 2195 cards are used as transmitters in Figs. 11–13, respectively. Interestingly, when a Broadcom BCM4318 card is used as the receiver, we do not observe the finger pattern for these three transmitters. For the Broadcom BCM4318 transmitter, although the saw-line is not regular and some sawteeth have higher peaks, compared to the other two transmitters, we cannot consider these sawteeth as fingers because the widths of fingers are multiples (either 3 or 4, as in Table IV) of those of sawteeth. However, the slope- and saw-line patterns are still evident in these three figures.

2) Atheros Ar9285 Receiver: Leveraging the recently developed *iw* utility for wireless devices, we can easily add a monitor interface on Atheros AR9285 802.11n cards that can pass error packets to user space. We show the experimental results when EMP Atheros AR5006, Intel PRO 2195, and Broadcom BCM4318 cards are used as transmitters in Figs. 14–16, respectively. Similar to the results when a Broadcom BCM4318 card is used as the receiver, we do not observe the finger pattern for

[3] We have not experimented with all the possible combinations due to the limited access to some chipsets/devices.

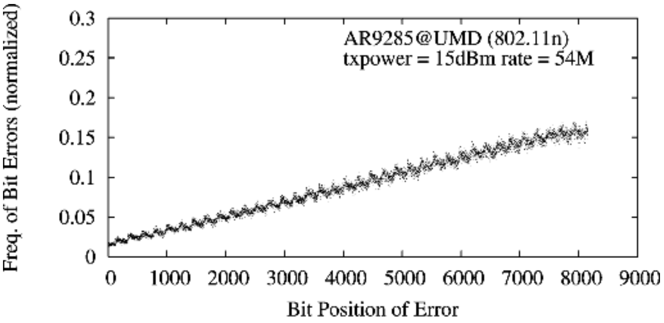

Fig. 14. Normalized bit error frequency for EMP Atheros AR5006 to Atheros AR9285. The slope of the fitting line is 1.970×10^{-5} with 95% confidence bounds (1.965×10^{-5}, 1.975×10^{-5}), and the sawtooth period is 215.769 with 95% confidence bounds (215.695, 215.917).

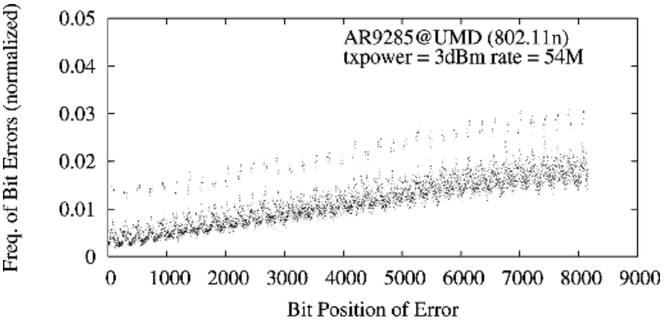

Fig. 15. Normalized bit error frequency for Intel PRO 2915 to Atheros AR9285. The slope of the fitting line is 1.935×10^{-6} with 95% confidence bounds (1.908×10^{-6}, 1.961×10^{-6}), and the sawtooth period is 216.289 with 95% confidence bounds (216.140, 216.363).

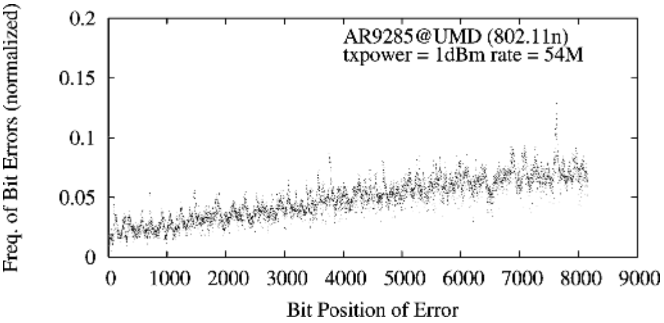

Fig. 16. Normalized bit error frequency for Broadcom BCM4318 to Atheros AR9285. The slope of the fitting line is 6.628×10^{-6} with 95% confidence bounds (6.555×10^{-6}, 6.701×10^{-6}), and the sawtooth period is 218.546 with 95% confidence bounds (216.438, 220.617).

these three transmitters with an Atheros AR9285 card as the receiver, although the slope-line and saw-line patterns still exist.

Remark: Similar finger patterns were also observed in prior work, such as from an 802.11b testbed using Harris/Intersil PRISM I chipsets in an industrial environment [15], an in-building 802.11a testbed with Atheros 5212 chipsets [2], and a testbed of a static access point (AP) and a mobile user [22]. However, all of these testbeds used the old version of 802.11 chipsets (e.g., PRISM I or Atheros 5212). We verified that the finger pattern does not appear when Broadcom BCM4318 and Atheros AR9285 802.11n chipsets (a newer product of Atheros) are used as the receivers.

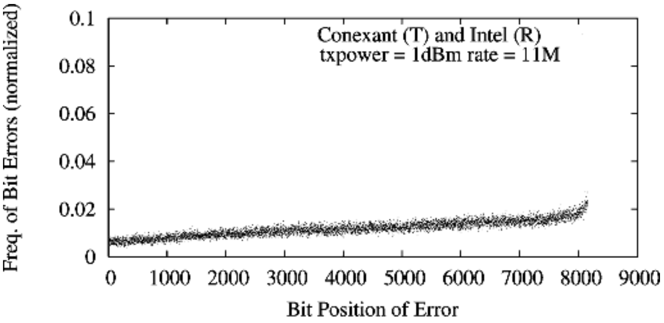

Fig. 17. Normalized bit error frequency for Conexant PRISM to Intel PRO 2100. The slope of the fitting line is 1.288×10^{-6} with 95% confidence bounds (1.275×10^{-6}, 1.301×10^{-6}).

3) Intel Receiver: Intel PRO 2100 chipsets support only 802.11b mode, which uses DSSS modulation. Thus, we present the experimental results using the Intel PRO 2100 receiver in Section IV-F (Fig. 17).

More experimental results of the Atheros AR5006 receiver are available in Han *et al.* [20].

F. Different Modulation

IEEE 802.11b uses DSSS CCK modulation, which is quite different from the OFDM modulation used by IEEE 802.11a/g. Fig. 17 shows the result when a Conexant PRISM card is used as the transmitter and an Intel PRO 2100 card is configured as the receiver. The bit rate is 11 Mb/s, the maximal rate for IEEE 802.11b that is the only mode supported by Intel PRO 2100. Under this configuration, we can still find the slope. We also repeated the experiments on the primary testbed with 802.11b settings (e.g., 11 Mb/s bit rate), and the results are available in Han *et al.* [20].

G. Challenged 802.11 Environments

With the increasing popularity of WiFi-enabled smartphones, IEEE 802.11 technology has been widely used for more challenged environments (compared to traditional indoor WLANs), including mobile and outdoor environments. We also performed experiments for these two challenged environments using smartphones. We used a Nokia N900 smartphone as the transmitter for these experiments. Its default OS, Maemo 5, is an open-source Linux distribution (2.6.28 kernel). The WiFi chipset is Texas Instruments WL1251, which supports 802.11b/g. The receiver was an Asus Eee PC netbook equipped with an Atheros AR9285 802.11n card.

During the mobile experiments, we set up the receiver (also the monitor to dump error packets) in a hallway of an office building, as shown in Fig. 18. We walked between two locations, A and B in Fig. 18, in the same hallway with the smartphone transmitter in hand. We performed the outdoor experiments in an empty parking lot at the University of Maryland, College Park, during a weekend. For both environments, we collected error packets transmitted at 54 Mb/s.

We show the result for mobile environment in Fig. 19 and outdoor environment in Fig. 20, respectively. As we can see from these two figures, the slope- and saw-line patterns are still present for these challenged environments. However, we do not

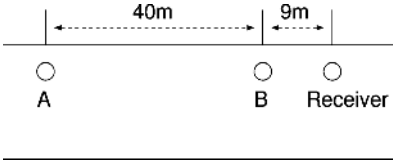

Fig. 18. Mobile testbed in a hallway. During the experiments, we walk between A and B with the smartphone transmitter in hand.

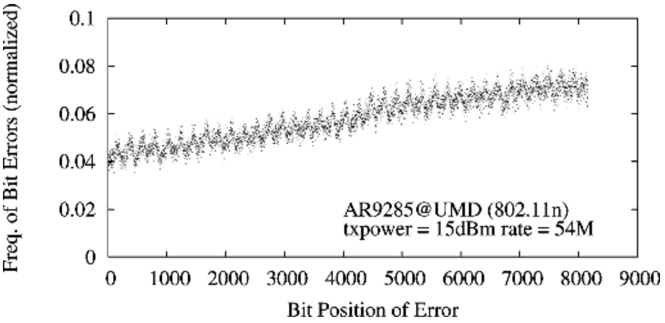

Fig. 19. Normalized bit error frequency for TI WL1251 to Atheros AR9285, mobile environment. The slope of the fitting line is 3.925×10^{-6} with 95% confidence bounds ($3.893 \times 10^{-6}, 3.957 \times 10^{-6}$), and the sawtooth period is 215.769 with 95% confidence bounds (215.695, 215.917).

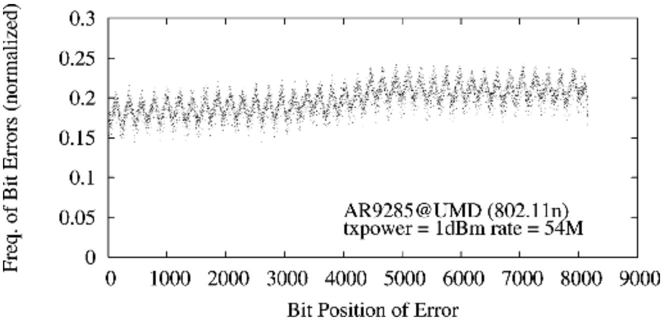

Fig. 20. Normalized bit error frequency for TI WL1251 to Atheros AR9285, outdoor environment. The slope of the fitting line is 4.676×10^{-6} with 95% confidence bounds ($4.551 \times 10^{-6}, 4.801 \times 10^{-6}$), and the sawtooth period is 215.917 with 95% confidence bounds (215.843, 215.991).

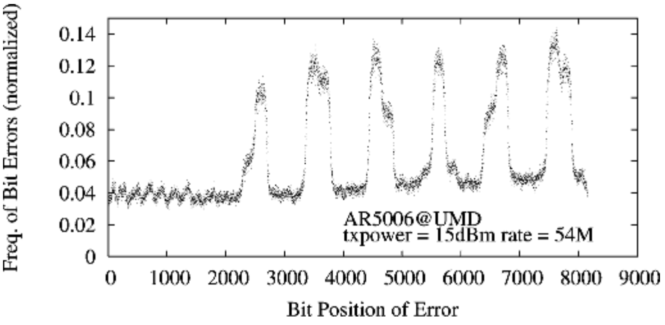

Fig. 21. Normalized bit error frequency for TI WL1251 to DCMA Atheros AR5006, mobile environment. The slope of the fitting line is 1.771×10^{-6} with 95% confidence bounds ($1.728 \times 10^{-6}, 1.815 \times 10^{-6}$), and the sawtooth period is 215.473 with 95% confidence bounds (214.443, 216.587).

identify a clear finger pattern, which further verifies the experimental results in Section IV-E. To figure out whether the finger pattern appears in these environments, we repeated the mobile experiments with a dedicated monitor using a DCMA Atheros AR5006 card. We plot the result in Fig. 21, which again shows a superposition of the three patterns. Note that the monitor was supported by the ath5k device driver for Atheros chipsets, which is a replacement of the MadWifi device driver that we have used for all the previous experiments, where Atheros AR5006 and AR5212 chipsets were involved.

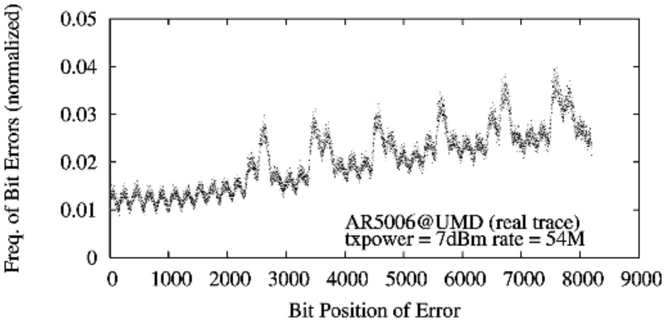

Fig. 22. Normalized bit error frequency for EMP Atheros AR5006 to DCMA Atheros AR5006 using real traces. The slope of the fitting line is 2.316×10^{-6} with 95% confidence bounds ($2.286 \times 10^{-6}, 2.346 \times 10^{-6}$), and the sawtooth period is 214.957 with 95% confidence bounds (214.370, 215.547).

H. Real Traces

We finally performed experiments to study bit error patterns using traces collected in an office environment, although we are confident that the identified bit error patterns are not caused by packet contents.

We collected real IEEE 802.11 traces over the air in an office building that contain only data packets with captured length at least 1200 B. Then, we fed the traces to a tool, called `Bits-Analyzer`, which works as follows. At the beginning of a single experiment, the transmitter retrieves a packet from the traces, sends it to the receiver through an Ethernet control channel, and then repeatedly transmits 1024 B of its payload over the wireless channel. When the receiver gets a corrupted packet, it compares the received data bits to those in the reference packet received through the control channel to determine which bits are corrupted. After the receiver gets enough number of error receptions of a data packet (10 in our experiments) on its monitor mode 802.11 interface, it notifies the transmitter to move on to the next packet in the traces.

We present the experimental results using the EMP Atheros AR5006 transmitter and the DCMA Atheros AR5006 receiver in Fig. 22. The real trace contains 10 000 data packets. As we can see from this figure, these three patterns are independent of packet payloads and still exist when an Atheros AR5006 chipset is used as a receiver.

I. Summary

During the measurement study on IEEE 802.11 WLAN testbeds, we have identified three distinct patterns for bit error probabilities with respect to bit positions: slope-line, saw-line, and finger. We have verified that the presence of the first two patterns is consistent in different environments and across different hardware platforms.

In our experience, the slope pattern is universal. It is present in all experimental results. This pattern shows that there is apparently a linear relationship between the chance of bit error occurrence and its bit position. Bits near the end of a frame are more likely to be received in error compared to bits in earlier portion of a frame.

The slope-line pattern may appear alone. However, as signal quality drops further, the other two patterns begin to show. For example, we can see only the slope pattern in Fig. 6 for receiver node 3, but all three patterns in Fig. 7 for receiver node 4. As node 4 is farther away from the transmitter, compared to node 3 (as shown in Fig. 5), the quality of the received signal at node 4 may be worse than that at node 3 for the same transmitted packet. The saw-line is also observable in almost all experimental results. For OFDM transmissions, the sawtooth peak-to-peak distance is exactly the number of bits carried by each OFDM symbol. For DSSS transmissions, the peak-to-peak distance appears to be a multiple of the number of bits carried by each symbol.

The finger pattern has been observed mainly in OFDM transmissions, but not for all hardware platforms. So far, we have not identified clear finger patterns for the Broadcom BCM4318 and Atheros AR9285 receivers. It may be either in the form of "peaks" or "valleys." The width of the fingers is a multiple of the number of bits carried by each symbol, usually three to four symbols.

V. Hypotheses and Discussions

It is difficult to pinpoint the exact causes of the identified patterns without access to detailed WLAN hardware design. We explore some possible reasons for the slope-line, saw-line, and finger patterns in this section. We note that these patterns are not likely to be caused by flaws/bugs in device drivers because we used nine different drivers on 10 different platforms in our experiments.

Two apparent reasons for the slope-line pattern are clock drift and changes of channel conditions. As mentioned before, synchronization between receiver and transmitter clocks is done only through receiving special symbols prepended at the very beginning of each frame. Although there are four pilot subcarriers in each OFDM symbol in order to make the coherent detection robust against phase noise and frequency offsets [17], and thus make a receiver be able to track clock drifts and channel errors, commodity hardware may do a poor job in implementing these pilot subcarriers, probably due to cost reasons. Thus, because of synchronization errors and clock drifting, as time goes on and bit reception progresses, the offset between the receiver's clock and transmitter's clock increases. As a result, boundary alignment of transmitted symbols and receiver samples deteriorates. This inevitably leads to increased bit error probability. Moreover, transmitters only sense the wireless channel prior to transmission. Therefore, some hidden terminals may start their own transmissions during a packet reception, which will generate external interferences. Although this is more likely to cause truncated frames, we cannot rule out this being a reason for later positions having higher bit error probability than earlier positions.

The saw-line pattern of OFDM transmissions is likely caused by the frequency selectivity characteristic of wireless channel, the transmitter, and the receiver [23]. Because of this frequency selectivity, certain OFDM subcarriers may experience higher error rates than others [24]. The interleaver of 802.11a/g is designed to map adjacent data bits to subcarriers that are far apart from each other. However, because the interleaving permutation is identical for all symbols, frequency selectivity induced bit error pattern will also be repeated for every symbol. This is the reason that the saw-line peak-to-peak distance is exactly the OFDM symbol length. By exploring the difference between the error rates of these subcarriers, we may be able to design more efficient retransmission protocols. For example, Li *et al.* [25] recently proposed Remap, a scheme that permutes the bit-to-subcarrier mapping after each retransmission and thus improves decoding efficiency and link throughput.

Another possible reason for the saw-line pattern is the residual sampling frequency offset (SFO), which is caused by small oscillator frequency differences between transmitters and receivers. One of the principal disadvantages of OFDM is its vulnerability to synchronization errors. For OFDM systems, the bit error rate is very sensitive to mismatches of both timing and frequency between oscillators of transmitters and receivers [26]. Although large sampling frequency offsets can be corrected during receiver acquisition [27], small residual offsets (e.g., errors in sampling offset estimates) result in a phase increase across frequencies that grows *linearly* across OFDM subcarriers [28].

The finger pattern is the most difficult to explain, although it exists mainly on Atheros AR5006 and AR5212 receivers. One possibility is that this pattern is caused by the interplay between the transmitter's power control loop and the receiver's gain control loop. The finger pattern may heavily depend on the OFDM receiver hardware design of some specific 802.11 chipsets (e.g., Atheros AR5006), as it does not appear for other types of chipsets (Atheros AR9285 and Broadcom BCM4318). Further experiments and investigations on the reasons for the finger pattern are part of our future work.

Previously the research community has been mainly focusing on characterizing channel fading, noise, and interference resulted bit errors. However, none of these reasons is likely to produce the patterns reported here. Most of our current hypotheses point to hardware-related reasons. We believe that hardware induced bit error patterns do exist and play an important role in causing bit errors in WLAN systems.

Despite the uncertainties in the root causes for these bit error patterns, we believe that identifying these patterns alone is beneficial for a number of subframe error recovery mechanisms [3]. For instance, knowing the slope-line bit error pattern, instead of transmitting the same frame for the second time, retransmitting a frame with data bits reordered in reversed order from the original frame may improve loss resilience for retransmission-with-memory techniques [4]. Moreover, in many cases the fingers are where most bit errors occur. For instance, for node 4 of our primary testbed, in some cases (e.g., 48 Mb/s transmission bit rate) 17.64% of packets received with bit errors have all their erroneous bits under the fingers. A variable coding scheme that can code bits in the finger regions with rates lower than other regions may potentially reduce the number of packets received with bit errors by a healthy margin. For example, multirate wireless packetization [22] is a scheme for which different parts of the same data packet are modulated at different physical-layer bit rates. It proposes to use the highest possible bit rate for bit positions with low error probabilities and reduce the bit rate for those with higher error probabilities.

VI. Related Work

In this section, we review related literature in two categories: modeling of bit errors and measurement study of packet and bit errors.

A. Modeling of Bit Errors Over Wireless Channel

Theoretical models have been proposed to describe communication bit errors over various wireless channels. The simulation-based performance evaluation of a given protocol heavily depends on the choice of these channel error models. As our focus in this paper is a measurement study of bit error behavior, we only provide a brief review of these models. Among these models, finite-state Markov chain-based models [29] are among the most popular. For example, Zorzi *et al.* investigate the behavior of block errors in data transmissions over fading channels [9]. They consider the details of specific coding/modulation scheme and track the fading process symbol by symbol. Their results show that a Markov approximation for block error process works very well for a broad range of parameters. Besides these Markovian models, Köpke *et al.* propose to use a chaotic map as a model for bit errors over wireless channels and describe how to determine the model parameters based on measurement data [13].

B. Measurement Study of Packet/Bit Errors

Other researchers report measurement results of bit and packet errors in various environments. Using AT&T WaveLAN wireless interfaces, Eckhardt and Steenkiste [14] characterize packet errors and evaluate the effects of interference and attenuation due to distance and obstacles on packet loss rate and bit error rate. Willig *et al.* [15] present results of bit error measurements obtained using an IEEE 802.11-compliant radio in an industrial environment. They show that the popular Gilbert/Elliot model and its slight modification are useful for simulating bit errors on a wireless link. The main focus of their work is to simulate wireless transmission errors more accurately. Aguayo *et al.* [16] analyze the causes of packet loss in a 38-node urban multihop 802.11b network. They find that link error rates stay relatively uniform for the majority of links. Reis *et al.* [18] propose practical measurement-based models for packet reception and interference in static wireless networks. They use the measured RSS values and packet delivery probability to characterize link quality. They also find that generally packet loss at one receiver does not mean loss elsewhere.

As mentioned above, similar finger patterns were also observed in previous work. Miu *et al.* [2] find the burst pattern from data packets transmitted at a single bit rate, 48 Mb/s. They show that the bit error distribution is often clustered within 300–400 bits, and the clusters are around 800–1200 bit positions apart. Their hypothesis is that this burst nature is due to the OFDM scheme employed in 802.11a/g. Willig *et al.* [15] find the finger pattern at lower bit rates in an industrial environment. They observe that the finger-to-finger distance is 64 bits for BPSK and 128 bits for QPSK. They believe this pattern is caused by the artifacts of bit synchronization algorithm on the wireless receivers. Mishra *et al.* [22] also find a periodic trend of high bit error rate about 1000 bit positions apart on a small testbed of a static AP and a mobile user. Their hypothesis is that this trend is due to the mapping of data bits to the different OFDM subcarriers.

Bit error characteristics have also been studied for other wireless networks. For example, Ilyas and Radha [30] study the error process of 802.15.4 wireless networks. They analyze the bit error rate at both individual bit and packet-by-packet levels by examining traffic traces between a transmitter and a receiver. They also investigate the distribution of bit errors within a packet to understand the channel memory of an 802.15.4 link.

VII. Conclusion

In this paper, we have presented experimental results from a study for understanding the subframe bit error characteristics of IEEE 802.11 transmissions. We have identified three distinct patterns for subframe bit error frequency versus bit position: slope-line, saw-line, and fingers. We verified that the first two patterns exist in different physical environments and across different hardware platforms. We have offered hypotheses for what may cause these bit error patterns.

Lacking detailed knowledge of how hardware vendors design and implement their WLAN chips, it is difficult to pinpoint the exact causes of the bit error patterns that we have discovered from our experiments. However, we believe that identifying repeatable and predictable bit error patterns is important in itself because the patterns may provide valuable insights for modeling subframe bit errors.

In the future, in addition to further experimentation and investigation with other platforms (e.g., GNU Radio and WARP from Rice University, Houston, TX), we plan to take into account these bit error patterns when designing more efficient dynamic intrapacket encoding schemes and packet retransmission mechanisms. Exploring the bit error behavior in wireless sensor networks, such as IEEE 802.15.4 wireless networks, will be another interesting area of future work.

Acknowledgment

The authors thank the anonymous reviewers for their insightful comments. They thank H. Worstell, S. Ghassemzadeh, M. Mao, F. Gringoli, T. Li, T. C. Clancy, D. Halperin, and A. Schulman for their support and valuable inputs. They thank E. Chai, Z. Zhang, R. Jana, C. Wang, and D. Bai for discussions. Part of this work was done when B. Han was a summer intern with AT&T Labs—Research.

References

[1] M. Zorzi and R. R. Rao, “Perspectives on the impact of error statistics on protocols for wireless networks,” *IEEE Pers. Commun.*, vol. 6, no. 5, pp. 32–40, Oct. 1999.

[2] A. Miu, H. Balakrishnan, and C. E. Koksal, “Improving loss resilience with multi-radio diversity in wireless networks,” in *Proc. MobiCom*, Aug.–Sep. 2005, pp. 16–30.

[3] K. Jamieson and H. Balakrishnan, “PPR: Partial packet recovery for wireless networks,” in *Proc. SIGCOMM*, Aug. 2007, pp. 409–420.

[4] P. S. Sindhu, “Retransmission error control with memory,” *IEEE Trans. Commun.*, vol. 25, no. 5, pp. 473–479, May 1977.

[5] S. Choi, Y. Choi, and I. Lee, “IEEE 802.11 MAC-level FEC scheme with retransmission combining,” *IEEE Trans. Wireless Commun.*, vol. 5, no. 1, pp. 203–211, Jan. 2006.

[6] G. Woo, P. Kheradpour, D. Shen, and D. Katabi, “Beyond the bits: Cooperative packet recovery using physical layer information,” in *Proc. MobiCom*, Sep. 2007, pp. 147–158.

[7] K. C.-J. Lin, N. Kushman, and D. Katabi, "ZipTx: Harnessing partial packets in 802.11 networks," in *Proc. MobiCom*, Sep. 2008, pp. 351–362.

[8] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: Practical wireless network coding," in *Proc. SIGCOMM*, Sep. 2006, pp. 243–254.

[9] M. Zorzi, R. R. Rao, and L. B. Milstein, "Error statistics in data transmission over fading channels," *IEEE Trans. Commun.*, vol. 46, no. 11, pp. 1468–1477, Nov. 1998.

[10] E. N. Gilbert, "Capacity of a burst-noise channel," *Bell Syst. Tech. J.*, vol. 39, pp. 1253–1265, Sep. 1960.

[11] E. O. Elliot, "Estimates of error rates for codes on burst-noise channels," *Bell Syst. Tech. J.*, vol. 42, pp. 1977–1997, Sep. 1963.

[12] H. Balakrishnan, S. Seshan, E. Amir, and R. H. Katz, "Improving TCP/IP performance over wireless networks," in *Proc. MobiCom*, Nov. 1995, pp. 2–11.

[13] A. Köpke, A. Willig, and H. Karl, "Chaotic maps as parsimonious bit error models of wireless channels," in *Proc. IEEE INFOCOM*, Mar. –Apr. 2003, pp. 513–523.

[14] D. Eckhardt and P. Steenkiste, "Measurement and analysis of the error characteristics of an in-building wireless network," in *Proc. SIGCOMM*, Aug. 1996, pp. 243–254.

[15] A. Willig, M. Kubisch, C. Hoene, and A. Wolisz, "Measurements of a wireless link in an industrial environment using an IEEE 802.11-compliant physical layer," *IEEE Trans. Ind. Electron.*, vol. 49, no. 6, pp. 1265–1282, Dec. 2002.

[16] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, "Link-level measurements from an 802.11b mesh network," in *Proc. SIGCOMM*, Aug.–Sep. 2004, pp. 121–132.

[17] *IEEE Standard for Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, ANSI/ IEEE Std 802.11 1999 Edition (R2003), Jun. 2003.

[18] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, "Measurement-based models of delivery and interference in static wireless networks," in *Proc. SIGCOMM*, Sep. 2006, pp. 51–62.

[19] L. Qiu, Y. Zhang, F. Wang, M.-K. Han, and R. Mahajan, "A general model of wireless interference," in *Proc. MobiCom*, Sep. 2007, pp. 171–182.

[20] 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," *Proc. IEEE INFOCOM*, pp. 1602–1610, Apr. 2009.

[21] F. Gringoli and L. Nava, "Open firmware for WiFi networks," 2009 [Online]. Available: http://www.ing.unibs.it/openfwwf/

[22] A. Mishra, S. Rayanchu, D. Agrawal, and S. Banerjee, "Supporting continuous mobility through multi-rate wireless packetization," in *Proc. HotMobile*, Feb. 2008, pp. 33–37.

[23] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[24] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Predictable 802.11 packet delivery from wireless channel measurements," in *Proc. SIGCOMM*, Aug.–Sep. 2010, pp. 159–170.

[25] L. E. Li, K. Tan, H. Viswanathan, Y. Xu, and Y. R. Yang, "Retransmission $\neq$ repeat: Simple retransmission permutation can resolve overlapping channel collisions," in *Proc. MobiCom*, 2010, pp. 281–292.

[26] P. H. Moose, "A technique for orthogonal frequency division multiplexing frequency offset correction," *IEEE Trans. Commun.*, vol. 42, no. 10, pp. 2908–2914, Oct. 1994.

[27] M. Speth, S. A. Fechtel, G. Fock, and H. Meyr, "Optimum receiver design for wireless broad-band systems using OFDM—Part I," *IEEE Trans. Commun.*, vol. 47, no. 11, pp. 1668–1677, Nov. 1999.

[28] C. Oberli, "ML-based tracking algorithms for MIMO-OFDM," *IEEE Trans. Wireless Commun.*, vol. 6, no. 7, pp. 2630–2639, Jul. 2007.

[29] C. C. Tan and N. C. Beaulieu, "On first-order Markov modeling for the Rayleigh fading channel," *IEEE Trans. Commun.*, vol. 48, no. 12, pp. 2032–2040, Dec. 2000.

[30] M. U. Ilyas and H. Radha, "Measurement based analysis and modeling of the error process in IEEE 802.15.4 LR-WPANs," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 1274–1282.

Bo Han received the Bachelor's degree in computer science and technology from Tsinghua University, Beijing, China, in 2000, and the M.Phil. degree in computer science from City University of Hong Kong, Hong Kong, in 2006, and is currently pursuing the Ph.D. degree in computer science at the University of Maryland, College Park.

He worked as a Research Intern during the summer with AT&T Labs—Research, Florham Park, NJ, during 2007 to 2009, Deutsche Telekom Laboratories, Berlin, Germany, in 2010, and HP Labs, Palo Alto, CA, in 2011. His research interests include wireless networking, mobile computing, distributed and randomized algorithms, and Internet routing.

Lusheng Ji (M'01–SM'06) received the Ph.D. degree in computer science from the University of Maryland, College Park, in 2001.

He developed multicast protocols for ad hoc networks at the University of Maryland. He was a Research Scientist with the Fujitsu Laboratories of America, College Park, MD. He is currently a Principal Member of Technical Staff with AT&T Labs—Research, Florham Park, NJ. His research interests include ad hoc networks, routing protocols, m-commerce, and wireless security.

Seungjoon Lee received the Bachelor's and Master's degrees from Seoul National University, Seoul, Korea, in 1996 and 2000, respectively, and the Ph.D. degree from the University of Maryland, College Park, in 2006, all in computer science.

Currently, he is a Senior Member of Technical Staff with AT&T Labs—Research, Florham Park, NJ. His research interests include wireless networks, mobile computing, peer-to-peer systems, and multicasting.

Bobby Bhattacharjee received the Bachelor's degrees in computer science and mathematics from Georgia College, Milledgeville, in 1994, and the Ph.D. degree in computer science from the Georgia Institute of Technology, Atlanta, in 1999.

He is currently a Professor with the Department of Computer Science, University of Maryland, College Park. His research interests include the design and implementation of decentralized systems, network security protocols, and wireless systems.

Robert R. Miller received the B.S. degree from University of Pittsburgh, Pittsburgh, PA, in 1969, and the M.S. degree from Stevens Institute of Technology, Hoboken, NJ, in 1977, both in electrical engineering.

He currently heads the Communication Technology Research Department, AT&T Shannon Laboratory, Florham Park, NJ. His career spans over 40 years commencing at Bell Laboratories, where he was involved with design of the first cellular telephones and systems in the 1970's. Since then, he has led R&D groups in design of contactless smart cards, digital telephones, high-performance conferencing systems, and multimedia wireless systems and devices, including recent work on new networked telehealth, energy management, and other IP-based M2M systems and services. He has also been active in standards development including the IEEE 802.11 QoS MAC, the first in a series of enhancements to prepare wireless LANs for broadband multimedia applications. He holds numerous patents associated with RF, signal processing, and acoustic and wireless systems. His recent research focus has been on capacity enhancement, QoS, radio resource management, cognitive radio, and automatic system configuration advances for next-generation wireless networks.

Mr. Miller is a Bell Laboratories and AT&T Labs Fellow. He received the AT&T Science and Technology Award Medal in 2004 and the AT&T Intellectual Property Achievement Award in 2008.