CSMA/CN: Carrier Sense Multiple Access With Collision Notification

Souvik Sen, *Student Member, IEEE*, Romit Roy Choudhury, *Senior Member, IEEE*, and Srihari Nelakuditi, *Member, IEEE*

***Abstract*—A wireless transmitter learns of a packet loss and infers collision only after completing the entire transmission. If the transmitter could detect the collision early [such as with carrier sense multiple access with collision detection (CSMA/CD) in wired networks], it could immediately abort its transmission, freeing the channel for useful communication. There are two main hurdles to realize CSMA/CD in wireless networks. First, a wireless transmitter cannot simultaneously transmit and listen for a collision. Second, any channel activity around the transmitter may not be an indicator of collision at the receiver. This paper attempts to approximate CSMA/CD in wireless networks with a novel scheme called CSMA/CN (*collision notification*). Under CSMA/CN, the receiver uses PHY-layer information to detect a collision and immediately notifies the transmitter. The collision notification consists of a unique signature, sent on the same channel as the data. The transmitter employs a listener antenna and performs *signature correlation* to discern this notification. Once discerned, the transmitter immediately aborts the transmission. We show that the notification signature can be reliably detected at the listener antenna, even in the presence of a strong self-interference from the transmit antenna. A prototype testbed of 10 USRP/GNU Radios demonstrates the feasibility and effectiveness of CSMA/CN.**

***Index Terms*—Collision detection, wireless communication.**

I. Introduction

MAC PROTOCOLS in wired LANs are based on the principles of carrier sense multiple access with collision detection (CSMA/CD). With CSMA/CD, the transmitter simultaneously transmits and listens on the wired channel. On detecting a collision, the transmitter aborts its own transmission almost instantaneously. Performance improves because the remainder of the packet is not transmitted unnecessarily. Instead, the channel is released for other productive transmissions.

Wireless MAC protocols, however, must rely on CSMA/CA (collision avoidance). The transmitter must complete the entire packet transmission and then infer a collision from the absence of an ACK from the receiver. Channel utilization degrades because the failed packet will have to be retransmitted later. To reduce channel wastage, it would be desirable to emulate CSMA/CD-like behavior even in wireless networks.

Manuscript received November 07, 2010; revised May 31, 2011; accepted July 18, 2011. Date of publication December 16, 2011; date of current version April 12, 2012. This work was supported in part by the National Science Foundation (NSF) under Grants CNS-0448272, CNS-0747206, CNS-1017276, and CNS-1018369.

S. Sen and R. Roy Choudhury are with the Departments of Computer Science and Electrical and Computer Engineering, Duke University, Durham, NC 27708 USA (e-mail: souvik.sen@duke.edu; romit@ee.duke.edu).

S. Nelakuditi is with the Department of Computer Science and Engineering, University of South Carolina, Columbia, SC 29208 USA (e-mail: srihari@cse.sc.edu).

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.2011.2174461

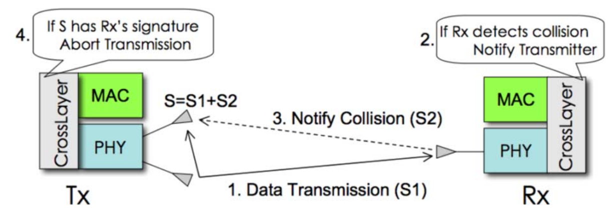


Fig. 1. Illustration of the basic operations of CSMA/CN protocol.

CSMA/CD is considered infeasible in wireless networks due to two main constraints. First, a wireless transmitter cannot transmit and listen on the same channel simultaneously. Even if it could, say with additional hardware, the signal strength of its own transmission (self-signal) would be too strong to detect a collision by the transmitter. Second, the wireless channel conditions are different at the transmitter and the receiver. Therefore, a collision detected by the transmitter may not indicate a collision at the receiver. Limited by these constraints, a wireless transmitter first completes the transmission and then waits for an ACK from the receiver. If the ACK does not come back within a timeout duration, the transmitter assumes collision and prepares for retransmission.

We propose to approximate CSMA/CD for wireless networks with a scheme called CSMA/CN (*collision notification*). The high-level idea is simple. While receiving a packet, the receiver uses physical-layer hints [1] to detect a collision and immediately notifies the transmitter. The transmitter utilizes two antennas: one for normal transmission, and another dedicated to listening for the notification. Upon detecting the notification, the transmitter aborts its transmission, freeing up the channel for other transmitters in the vicinity.

Even with an additional listener antenna at the transmitter, two challenges underlie the design of CSMA/CN: 1) How does the receiver detect a collision while the packet is being received? 2) How does the transmitter detect a collision notification during its own transmission? We propose to address these issues by exploiting *signal correlation*. This primitive allows for discerning a known bit pattern even in the presence of a strong interference. Empowered by the correlation primitive, we develop the complete CSMA/CN protocol.

The operation of CSMA/CN can be summarized as follows. The transmitter has two interfaces tuned to the *same channel*, one for transmission and another for listening. The receiver has a single interface (Fig. 1). Once the communication begins, the receiver exploits *preamble correlation* to detect the presence of an interfering frame. Realizing that the packet reception is likely to fail, the receiver checks the confidence of incoming bits via physical-layer hints from SoftPHY [1], [2]. When the receiver is reasonably confident of an error, it initiates a collision notification to the transmitter. The notification is a short signature unique to the receiver, also known to the transmitter. The transmitter's listening antenna continuously "searches" for this signature using correlation. We show that even in the presence of a strong signal from the transmit antenna, *signature correlation* at the listening antenna can reliably discern the collision notification. The transmitter aborts, releasing the channel for other nearby transmitters.

Several questions arise with respect to the design and applicability of CSMA/CN. We touch upon a few relevant ones here and discuss them in detail in Section VI.

1) Instead of aborting the transmission, *why not recover from packet errors* with a scheme such as Partial Packet Recovery (PPR) [2]? PPR guesses which parts of a packet are in error and requests the retransmission of only those parts. With few bit errors, PPR offers good gains because it avoids an entire packet retransmission. However, when a packet undergoes a collision, many bits are likely to be in error, and retransmitting all of them can be wasteful. Collision notification aborts the transmission of a colliding packet. The intuition is that aborting (or prevention) is better than recovery (or cure). The rest of the packet's transmission is resumed later after appropriate backoff adjustment.

2) Instead of correlation, *why not use interference cancellation* to remove self-interference, and then "decode" the collision notification? Observe that the collision notification will be significantly weaker than the self-signal. Decoding this weak signal [with signal-to-interference-plus-noise ratio (SINR) far less than 0 dB] will require near-accurate self-signal cancellation. Such precise cancellation is hard in practice, making decoding unreliable. Correlation, on the other hand, is more robust, especially if combined with some degree of (even imperfect) self-signal cancellation. Moreover, for decoding a packet, it needs a preamble increasing the overhead and turnaround time for the collision notification.

3) Instead of listening on the additional interface, *why not use the interface for parallel communication* on a different channel? Note that the "listener" may not be viewed as another interface, only an antenna with correlation (and self-signal suppression) logic. Decoding capabilities are not necessary, hence this logic can be part of the same interface. Moreover, collision notification is transmitted on the same frequency channel as data, thereby requiring no additional bandwidth.

In addition to addressing these performance-oriented questions, CSMA/CN could potentially contribute toward the architecture of future wireless systems. Harnessing this potential is a topic of our future work. In this paper, we confine our focus to the feasibility of a MAC scheme. Our main contributions can be summarized as follows.

- *Identify a middle ground between CSMA/CD and CA*. CSMA/CN is an early attempt to rethink medium access control protocols in wireless networks. This paper explores the first steps in this direction, demonstrating that further progress is feasible and worth pursuing.
- *Develop the Collision Notification architecture with practical constraints in mind*. We incorporate two methods of self-signal suppression: 1) modeling and subtracting the wireless self-signal; and 2) sending the self-signal over a physical wire, and then subtracting it with greater precision. We show the feasibility of detecting collisions at the receiver, as well as reliable identification of the collision notification at the transmitter.
- *Implement and evaluate CSMA/CN on a prototype of 10 USRP/GNU Radio nodes*. Experimental results show consistent throughput improvements over IEEE 802.11 and PPR. We identify several avenues of further research.

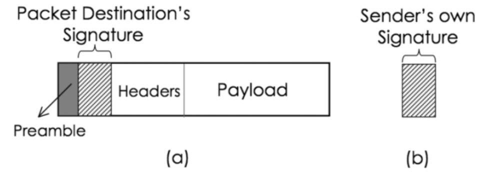


Fig. 2. Formats of CSMA/CN data frame and collision notification. Transmitter T inserts receiver R's signature after the data frames's preamble. Collision notification from R consists of only its own signature. (a) Data frame. (b) Collision notification.

Section II describes the architecture of CSMA/CN and focuses on the higher-level design of the system. The underlying challenges in implementing the key primitive—signal correlation—are detailed in Section III. Thereafter, we present the performance evaluation, limitations and opportunities, related work, and finally the conclusions.

II. Architecture and Design

We believe that any attempt to abort a colliding transmission in wireless networks will need to conform to the following functional requirements: 1) awireless transmitter T cannot reliably detect the collision on its own; the receiver R must get involved;) Receiver R will need to detect collision and convey it back before the packet is fully transmitted; 3) T needs at least an additional antenna for listening while transmitting. This section proposes CSMA/CN as a practical system that conforms to these requirements.

A. Transmission and Collision Detection

In CSMA/CN, the transmitter T uses one interface for transmitting and the other (listener) for listening. The receiver R uses its single interface for multiplexing between transmission and reception. Transmission is initiated as in IEEE 802.11, except one difference: For every packet, the PHY-layer preamble is concatenated with an additional bit sequence, a *signature*, uniquely computed from the intended receiver's identifier [Fig. 2(a)]. T ensures the channel is idle and transmits this packet using the transmit antenna. The listening antenna, by virtue of being very close to the transmitting antenna, receives this signal with a high signal strength—we call this the *self-signal*. The packet's intended receiver R also receives the transmitted signal and starts decoding the arriving bits. Simultaneously, R initiates collision detection.

Collision happens when a nearby transmitter $T1$ interferes with R's reception, causing packet corruption (Fig. 3). To detect such collisions, receiver R "searches" for a PHY-layer preamble in its incoming signal. Searching occurs through *correlation* of the preamble with the signal arriving at R's antenna. This happens in parallel and does not affect the normal packet decoding procedure. Once $T1$'s preamble impinges on R's antenna, the correlation exhibits a spike, raising an alert that the packet may be in "trouble." Of course, arrival of a new preamble may not necessarily cause a collision; reception of the packet may be

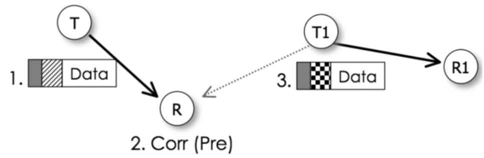


Fig. 3. While receiving from the transmitter T, the receiver R searches for a preamble of another frame via correlation [denoted Corr(Pre)].

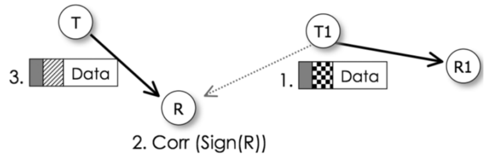


Fig. 4. While overhearing from the other transmitter $T1$, the receiver R searches for its own signature via correlation [denoted Corr(Sign(R))].

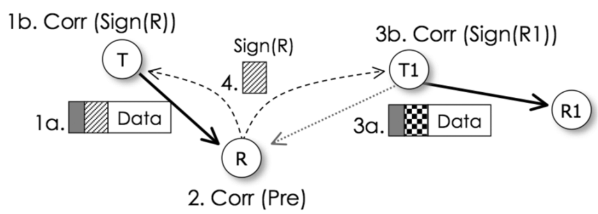


Fig. 5. During their respective transmissions, the transmitter T searches for its receiver R's signature, whereas $T1$ searches for $R1$'s signature. Hence, when R sends a collision notification, only T aborts its transmission.

successful sometimes even in presence of the interference. To verify the impact of interference, R consults SoftPHY [2] to obtain confidence values of the bits arriving from T. The confidence value is an indicator of how likely a bit is in error. Based on a window of confidence observations, R infers whether the packet is expected to get corrupted. If so, R halts reception and prepares to send a collision notification to transmitter T.

Now, if the interferer $T1$ starts first, and the transmission from T starts later, R may need to abort T (Fig. 4). However, R must first ensure that the later-arriving signal is actually meant for itself. Preamble correlation is not sufficient because T may use the same preamble for transmitting to some other receiver; R should not send an abort then. Because of this, R "searches" for its own signature in the signal. If T intends its transmission to R, it would embed R's signature in the packet. R will detect this through signature correlation.

To summarize, the receiver R searches for a preamble while receiving its frame of interest, but searches for its own signature while receiving an interfering frame.

B. Collision Notification and Abort

Upon detecting a collision, R stops receiving and prepares to transmit a collision notification (CN). The CN is composed of only R's own signature. This is the same bit sequence that T included in its packet to R [Fig. 2(a) and (b)]. The receiver transmits the CN packet like a regular 802.11 ACK—there is no carrier sensing, hence the CN is transmitted even though the transmitter is still transmitting. The listening antenna of the transmitter continuously correlates for the receiver's signature in the incoming signal (Fig. 5). This correlation is more challenging because the self-signal is much stronger than the notification. We show that even then the listener can discern the notification with consistent accuracy.

Upon detecting the collision notification, the listener immediately alerts the transmitting interface, which then suspends the transmission (other transmitters around, such as $T1$, do not suspend their transmissions because they are correlating with their respective receivers' signature, not R's). The correctly aborted transmitter backs off as prescribed in 802.11. Other backlogged nodes in the vicinity take up this opportunity to transmit; if no other node transmits, the same transmitter may resume the transmission of the aborted frame.

C. Packet Resumption and Acknowledgment

Under CSMA/CN, the transmitter does not retransmit the entire aborted packet. Instead, it resumes transmission from byte $B_{\rm re}$, where $B_{\rm re}$ indicates the maximum in-sequence byte received correctly by the receiver. $B_{\rm re}$ can be estimated because the receiver takes a constant time to detect the collision after its occurrence, responds with a fixed-size collision notification after SIFS interval, and the transmitter detects the notification signature in a constant time. Other remaining propagation delays are constant as well. Suppose the transmitter receives a notification while transmitting byte $B_{\rm now}$. Then, the estimate of $B_{\rm re} = B_{\rm now} - B_{\rm out}$ bytes, where $B_{\rm out}$ is determined based on the transmission bit rate. For example, in our design, collision detection takes time equivalent to 20 B. The time for notification signature of 20 B using BPSK over 20-MHz bandwidth is 8 μs. Thus, the turnaround time for notification including the SIFS interval of 10 μs would be 18 μs. This corresponds to 122 B at 54 Mb/s. Including the collision detection overhead of 20 B, a conservative estimate of $B_{\rm out}$ would be 150 B. Hence, if a sender is transmitting a 1500-B packet and aborts transmission at the 751th $(B_{\rm now})$ byte, it will resume from the 601st $(B_{\rm re})$ byte. Once the packet is transmitted, the CSMA/CN receiver responds with an ACK when it is received correctly. However, unlike 802.11 ACK frame, CSMA/CN ACK is simply a signature. If no ACK signature returns from the receiver, the transmitter times out and retransmits the entire packet.

We believe CSMA/CN is a simple approach to wireless medium access control. The following two pseudocodes capture the core flow of operations under CSMA/CN.

Algorithm 1: T.transmit(R, Data)

```
Begin sending frame <Preamble:Sign(R):Data>
Keep listening and correlating with Sign(R)
if Corr(Sign(R)) high then
    Suspend and resume transmission after backoff
if no Corr(Sign(ACK(R))) at the end of transmission then
    Retransmit after a random backoff
```

Algorithm 2: R.receive()

```
if frame of interest is already being received then
    if Corr(Pre) high and many bits suspect then
        Transmit <Sign(R)>
if interfering frame is being received then
    if Corr(Sign(R)) high then
        Transmit <Sign(R)>
if frame of interest reception successful then
    Transmit <Sign(ACK(R))$>$
```

D. Points of Discussion

There may be many concerns one might have related to the design of CSMA/CN. We list and clarify some of them below.

1) A pertinent issue is *whether the collision notification will interfere with nearby active transmissions*. This will certainly be true when the interferer's receiver $(R1)$ is close to the notification sender (R). Nevertheless, the small size of the notification permits various possibilities for efficient recovery. When it interferes, the short window of bit errors can be repaired by a scheme like PPR, as if it is a small burst of fading loss. PPR is an effective scheme to cope with fading and can handle errors due to notification as well. Alternatively, the packet may be augmented with just enough error correcting codes to recover from the notification-sized errors [3]. Finally, observe that 802.11 ACKs can also induce errors at a nearby receiver, much like CSMA/CN's collision notifications. They only differ in the kind of topologies they impact. We will later evaluate these impacts in Section IV. Also note that under CSMA/CN, a receiver first correlates with a preamble. Only upon strong correlation, it inspects bit confidence values for detecting a collision. A collision notification is simply a signature without any preamble. Therefore, a collision notification does not cause cascading collision notifications.

2) Another question is *what happens when two transmitters send to the same receiver?* The receiver locks on to the transmitter that starts first, and while decoding its bits, simultaneously searches for a second preamble. On detecting the second transmitter's preamble, and confirming collision, it sends a collision notification. Both transmitters listen for the common receiver's notification and abort their transmissions.

3) Since the detection of collision and notification both depend on signatures, *how many distinct signatures do we need for the CSMA/CN approach to work correctly?* The number of signatures in the network varies as a function of the number of nodes because the signature effectively identifies the recipient of the data frame (or transmitter of the notification). Hence, the required signature space is $O(n)$.

With a basic understanding of the CSMA/CN architecture, we now present the core *signal correlation* primitive that underpins CSMA/CN. We begin with a brief background on this topic, followed by the description of correlation and self-signal suppression techniques.

III. Correlation Process

CSMA/CN's two main challenges pertain to detecting a collision and discerning the notification. Both these operations amount to searching for a known pattern (preamble or signature) in an incoming signal. We propose to accomplish this by performing *cross correlation* (similar to that in [4]) between the known pattern and the arriving signal. It is expected that when the pattern is present in the arriving signal, their cross correlation would yield a high correlation value. Therefore, by tracking the correlation value for such a spike, a station can verify the presence or absence of a pattern in the received signal. We refer to this as *signal correlation*.

The application of signal correlation to detect a collision at a receiver is straightforward. While receiving the signal of interest, the receiver can simply correlate with the known preamble. While overhearing the interference, the receiver can correlate with its own signature to recognize when a new transmission is meant for itself. On the other hand, the detection of collision notification at the transmitter is more challenging. The strong self-signal at the listening antenna can mask the notification from a faraway receiver leading to weak correlation. To achieve correlation of even weak notifications, we propose to suppress the self-signal with the aid of interference cancellation techniques. A perfect cancellation is not necessary—rather an approximate suppression is sufficient to strengthen the correlation and discern the notification from a distant receiver. We describe these schemes next, beginning with a brief background on signal correlation. We also show later that careful assignment of signatures is not necessary to unambiguously identify the receivers.

A. Signal Correlation

A wireless transmitter maps the bits of a packet into complex symbols as part of digital modulation. Therefore, a transmitted signal can be treated as a sequence of complex symbols. Let $x[n]$ be the complex number representing the nth transmitted symbol. Let $y[n]$ represent the corresponding received symbol after it gets attenuated and phase-shifted by the wireless channel. We can approximate their relationship as $y[n] = Hx[n] + w[n]$, where H is also a complex number representing the channel coefficient between the transmitter and the receiver, and $w[n]$ is random noise.

Suppose we intend to search a known symbol pattern s of length L in the received signal y. We can then define their cross correlation at a shifted position p as

$$\mathcal{C}(s, y, p) = \Sigma_{k=1}^{L} s^*[k] y[k+p]$$

where $s^*[k]$ is the complex conjugate of $s[k]$. The correlation coefficient $\mathcal{C}(s, y, p)$ is low when s is not present in y. Even when s is present, it stays low until $y[p]$ aligns with the beginning of s, at which point there would be a sudden spike in the correlation. Thus, by tracking $\mathcal{C}(s, y, p)$, we can detect the presence of a known pattern as soon as it arrives.

One issue still needs to be addressed. Due to manufacturing limitations, the transmitter and the receiver are not centered on the same frequency, but have a small difference δf, i.e., $y[n] = Hx[n]e^{j2\pi n \delta f T} + w[n]$. Without correcting for it, the correlation may not be strong even when the known pattern is present in the signal. This offset, however, is relatively static and can be estimated based on the history. Therefore, we can compensate for the offset in the received signal before computing the correlation. Hence, we have

$$\mathcal{C}(s, y, p) = \Sigma_{k=1}^{L} s^*[k] y[k+p] e^{-j2\pi(k+p)\delta f T}$$

where T is the sampling period and $e^{-j2\pi(k+p)\delta f T}$ is the compensation factor for frequency offset δf.

Fig. 6 presents the outcome of signal correlation with a given pattern. It shows a spike in the correlation value every time the known pattern arrives. The effectiveness of correlation can be explained as follows. Suppose two transmitted signals $y_A[n]$ and $y_B[n]$ from A and B concurrently arrive at the receiver. The resulting received signal $y[n] = y_A[n] + y_B[n] + w[n]$. Then, the result of correlation would be

$$\mathcal{C}(s, y, p) = \Sigma_{k=1}^{L} s^*[k](y_A[k+p] + y_B[k+p] + w[k+p]).$$

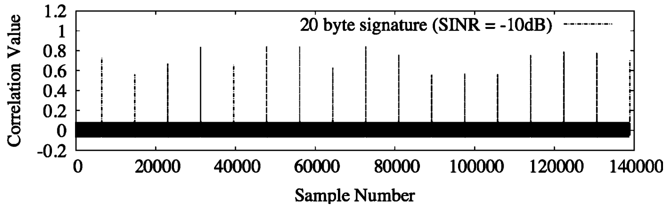


Fig. 6. Correlation spikes whenever a known pattern arrives amid a background transmission even if it is stronger (by 10 dB in this illustration).

Assume that the pattern we are looking for exists in the signal from B. Since the pattern is independent of the signal from A and the noise, the correlation with those terms would be close to zero. Also, canceling out the frequency offset by compensating for it, we have

$$\mathcal{C}(s,y,p) = \Sigma_{k=1}^{L} s^*[k] H_B x_B[k+p]$$

where x_B is the transmitted signal from B and H_B is the coefficient of the channel between B and the receiver. The correlation yields the highest value when x_B matches s. The resulting value of the spike is $\mathbb{R}(\mathcal{C}(s,y,\hat{p}))$, where $\mathcal{C}(s,y,\hat{p})$ is

$$\mathcal{C}(s,y,\hat{p}) = H_B \Sigma_{k=1}^{L} |s[k]|^2.$$

We normalize the correlation value to $\frac{\mathbb{R}(\mathcal{C}(s,y,\hat{p}))}{\Sigma_{k=1}^{L}|s[k]|}$ and apply a threshold on the resulting value to detect the presence of the known pattern. Thus, while receiving a packet from T, R can continuously search for a new interfering frame by making $s =$ "universally known preamble." If the interfering frame starts first, the receiver R can search for T's transmission by making $s =$ "R's own signature." Thus, signal correlation allows a receiver to promptly detect a collision.

Threshold Selection: The correlation value depends on signal strength and transceiver characteristics [5]. The higher the normalized correlation value, the easier it is to identify the pattern without false positives/negatives. Choosing a threshold is not hard considering that successful correlation yields a sudden spike as evident from Fig. 6. For CSMA/CN, low false positive is desirable, even at the expense of slightly higher false negatives. In our prototype implementation, the threshold is empirically estimated from an experiment to keep false positives below 5%. In practice, suitable thresholding is relatively easier at a receiver for detecting a collision than at a transmitter for detecting a collision notification in the presence of a self-signal. However, a transmitter can adapt its threshold using the results from previous correlations. Based on the past correlations corresponding to successful transmissions, a transmitter can choose a threshold to keep false positives low.

B. Self-Signal Suppression

A transmitter also uses signal correlation for detecting the collision notification at the listening interface by setting $s =$ receiver's signature. However, it is more challenging to detect the notification because the self-signal is much stronger than the notification at the listening antenna. Fig. 7 plots the normalized correlation value as a function of the difference in signal strengths between the self-signal and the notification. The default settings in our experiments are as follows. The transmit power is 12 dBm and the signal-to-noise ratio (SNR) of self-signal is 50 dB. The self-signal was kept at 50 dB to prevent the analog-to-digital converter (ADC) from saturation. The listening and transmitting antennas are separated by 2 ft. The size of the notification signature is 20 B (note that, like preamble, the signature is transmitted using BPSK). As evident from the results, the transmitter can reliably detect the notification when the receiver is nearby. If the receiver is not nearby (i.e., self-signal minus notification greater than 16 dB), correlation yields false detections of greater than 20%. This is not desirable because collisions are more likely when the receiver is farther from the transmitter.

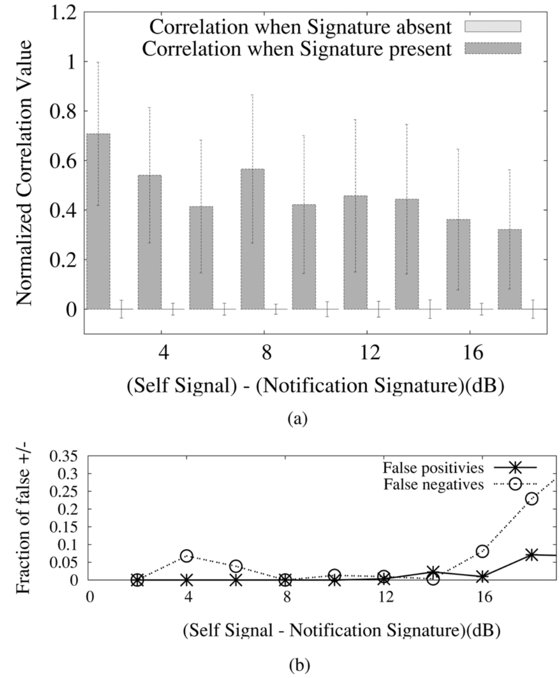


Fig. 7. Correlation performance with varying transmitter–receiver separation. When signature is absent, correlation values are quite small and thus not visible. Total false detection (positive and negative) could be higher than 20% if notification is weaker than self-signal by greater than 16 dB. (a) Correlation when signature is present/absent. (b) False positive/negative rates of discerning notification.

By the definition of correlation, the relative strengths of self-signal and collision notification should not matter in theory. Even when the notification is relatively weak, the correlation value should depend solely on the received energy of the notification. However, the theory assumes that the two signals (y_A and y_B) and channel coefficients (H_A and H_B) are independent. In practice, they are not completely independent. Consequently, the much stronger self-signal dominates in determining the outcome of correlation, making it harder to discern the notification from distant receivers.

To overcome this limitation, we propose to suppress the self-signal with the aid of interference cancellation techniques [4], [6], [7]. The major challenge is to model the various hardware and channel-specific effects experienced by the self-signal. However, we observe that the self-signal under CSMA/CN is more amenable for cancellation for the following reasons. First, the self-signal is a known signal for the transmitter. Second, the transmitting and listening interfaces will be close to each other, and so the wireless channel effects will be relatively small and time-invariant. Third, the listener hears the

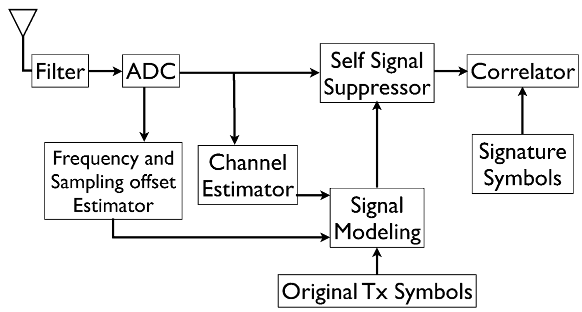


Fig. 8. Schematic of the self-signal suppression over wireless.

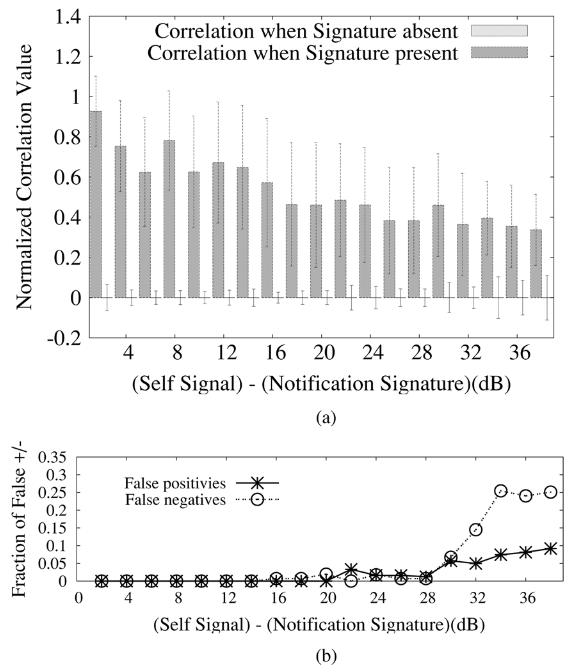


Fig. 9. Self-signal suppression over wireless with varying transmitter–receiver separation. The difference in signal strengths of self-signal and notification can be as high as 32 dB (total false detection <20%). (a) Correlation when signature is present/absent. (b) False positive/negative rates of discerning notification.

earlier part of the self-signal in the clear since the notification arrives only after the receiver has detected the collision and sent back the notification. This provides adequate opportunity to estimate and model the channel and hardware effects. Finally, since our aim is not to decode the bits but to improve the correlation, an approximate cancellation might suffice in discerning the notification from a distant receiver. That is why we refer to our approach as self-signal suppression.

Our self-signal suppression process is similar to interference cancellation in [4] and [7]. The signal received at the listening antenna is a combination of the transmitted self-signal and the collision notification signal from the client. The listener knows the symbols being sent and thus the transmitted signal $x[n]$. The received self-signal $y[n]$ is the result of several effects such as channel distortion, sampling offset, and frequency offset on the transmitted signal $x[n]$. These effects are modeled from the received self-signal in the clear prior to any potential collision notification. Once modeled, the derived signal is subtracted from the received signal. What remains is the collision notification signal along with the noise due to imperfect modeling of self-signal. The listening antenna can now correlate this residue with the receiver signature to detect the notification. Fig. 8 shows the overall architecture for wireless signal suppression. We elaborate on each of the steps involved in modeling the self-signal.

Channel Distortion: Due to the proximity of the two antennas, there would be relatively less path loss. Hence, we can maintain a crude estimate of the channel using the method mentioned in [4]. Specifically, when the preamble of the self-signal is received, we compute the complex channel impulse response from the received signal and the known preamble.

Sampling Offset: Since the transmitters and the receiver's clocks are not synchronized, the receiver may not sample at the ideal sampling points. A practical wireless receiver tracks the sampling offset (τ) and performs interpolation to estimate the "ideal" sampled points $(x[n])$. Using the same estimate of τ, we can interpolate to find the complex values corresponding to the "actual" sampled points $(x'[n])$ as follows:

$$x'(n+\tau) = \Sigma_{k=-L}^{L} x[n+k] \sin c(\pi(n+\tau-k)).$$

Frequency Offset: CSMA/CN uses the receiver's estimate of frequency offset δf. Frequency offset is relatively stable and does not change for long durations. To incorporate the effect of frequency offset, the nth transmitted sample is phase-shifted by a factor of $e^{2\pi n \delta f T}$, where T is the symbol duration.

Filter and Intersymbol Interference Effects: Due to multipath effects, a wireless symbol interferes with its adjacent symbols. Also, hardware filters deliberately blend adjacent symbols to reduce bandwidth leakage. We model these effects as a least mean square filter. We train this filter with the clear portions of the self-signal and the resulting signal after applying the above distortions on the transmitted signal $x[n]$.

CSMA/CN needs to cancel the self-signal as it arrives from the transmit to the listening antenna. Since the notification will come back after a delay, the listener exploits this opportunity to estimate the distortion from the clear part of the self-signal. Now, the listener also knows the actual set of transmitted symbols, and therefore can replay the estimated distortion onto them, i.e., the sampling offset, frequency offset, and channel effects in sequence. This artificially distorted signal is expected to be an approximation of the self-signal and is thus subtracted from the (wirelessly) received signal. The subtraction happens in blocks, resulting in a residue. The listener correlates for the notification in this residue, and if the correlation does not spike, the listener repeats the same procedure for the next signal block. Fig. 9 shows that, with this approach, notification can be reliably detected even when it is weaker than self-signal by 32 dB (as opposed to only 16 dB without self-signal suppression).

C. *Self-Signal Suppression Over the Wire*

Since the signal we are trying to cancel is a self-signal, it can be passed on to the listening interface (from the transmit antenna) over a physical wire. The advantage is that the signal received over the wire incurs the same filter distortion and frequency offset effects as that of the wireless signal. This precludes the need to model these effects separately. Fig. 10 illustrates this process. Observe that if the wirelessly received self-signal has to be recreated from the known bits, the various hardware distortions would have to be modeled precisely.

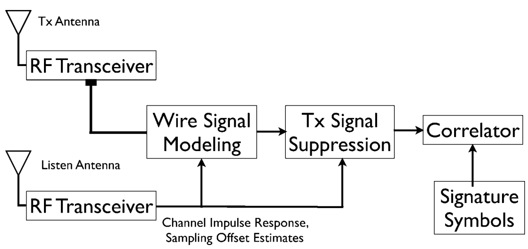


Fig. 10. Schematic of the self-signal suppression over the wire.

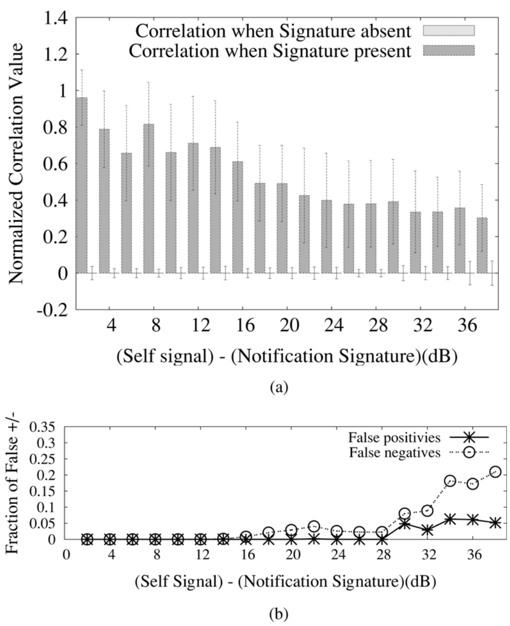


Fig. 11. Self-signal suppression over the wire with varying transmitter–receiver separation. The difference in signal strengths of self-signal and notification can be up to 34 dB (for less than 20% false detection). (a) Correlation when signature is present/absent. (b) False positive/negative rates of discerning notification.

We take the wired signal, compute its sampling offset, and then align it with the wireless signal based on their mutual difference in the offsets. At this point, the only distortion absent in the wired signal is the effect of channel and multipath. If we could "inject" these effects in the wired signal, it may be possible to create the wireless self-signal received by the listener antenna. For this, we capture these channel/multipath effects using a linear equalizer. Specifically, from the clear portion of the wireless self-signal, Y, we calculate a set of filter taps H, such that $Y - HX$ is minimized (here, X is the wired signal). The signal HX is then subtracted from the received signal Y, leaving a small residue. When the collision notification also arrives along with the self-signal, (i.e., Y is a sum of the self-signal and the notification), we expect the residue to contain the notification. The listener continuously correlates the signature with the residue and observes a spike at this time. Since wired cancellation is more reliable, the notification detection is more robust even under stronger self-signals. Fig. 11 shows that, when assisted by wired communication, the difference in signal strengths of self-signal and the notification can be up to 34 dB for reliable detection (compared to 32 dB, as shown in Fig. 9, with self-signal suppression over wireless).

D. Listener Antenna Orientation

The separation between antennas and their orientations influence the relative strengths of the self-signal and the notification. The self-signal is 50 dB stronger at a 2-ft separation. Since both the antennas have to be packaged on the same access point (AP), their maximum separation is limited. However, Fig. 12 demonstrates that antenna orientations can reduce the strength of the self-signal and hence can be exploited by CSMA/CN. Of course, it is relevant to ask whether *antenna orientations will also reduce the notification signal, affecting correlation*. Fig. 12(b) shows that when the original self-signal was 55 dB, the self-signal reduces by 12 dB while the notification strength reduces by 6 dB. In all our orientations, the transmit antenna is always placed upright to have minimum impact on the original transmission. Fig. 12(c) shows that by positioning the listening antenna correctly and performing self-signal suppression, it is possible to correlate notifications for clients that are 36 dB weaker than the self-signal.

We now discuss how, in the future, we can further suppress the self-signal to make the CSMA/CN scheme quite practical. With a reasonable antenna separation of 1 ft, modern APs will have a self-signal of 65 dB [8]. Our experimentation shows that we gain 15 dB from correlation and 20 dB from digital interference cancellation. Also, antenna orientation gives us a gain of 10 dB with 1-ft separation; see Fig. 12(b). Combining these, using only digital cancellation, we can offset a self-signal of 40 dB. We believe the remaining 25 dB can be handled using analog cancellation. There exist chipsets for analog cancellation that can cancel up to 30 dB of self-interference [9]–[11].

E. Signature Assignment

As discussed earlier, CSMA/CN needs $O(n)$ different signatures, where n is the number of nodes in the network. A natural question is *how "different" should the signatures be?* If n signatures must be very different, the signature size has to be larger. Fortunately, as discussed in Section III-A, there exists an inherent difference in the center frequency offsets for wireless radios (Fig. 13 confirms a wide range of diversity). When the transmitter is searching for the notification on its listening interface, it can account for its receiver's frequency offset. Even if some other node is transmitting a reasonably "similar" signature, the listening interface may not find a high correlation due to the differences in their frequency offsets and other hardware idiosyncrasies [5]. In other words, the frequency offset naturally creates some "dissimilarity" between the signatures, helping in keeping the signature short.

We studied *whether different signatures may induce similar correlations*. Fig. 5 shows a scenario where R's signature arrives when $T1$ is searching for $R1$'s signature. To ensure that $T1$ does not abort, CSMA/CN needs to ensure that the two signatures do not exhibit a high correlation (no false positives). Tables I and II show the fraction of false positive correlations between signatures with varying hamming distance and frequency offsets. Evidently, at practical frequency offsets, signatures that differ by as few as 48 b can be robustly distinguished (with less than 4% false positives). The number of signatures available, however, depends on its size. Depending on the number of nodes in the network, the receiver can dynamically select the size and pattern of such signatures.

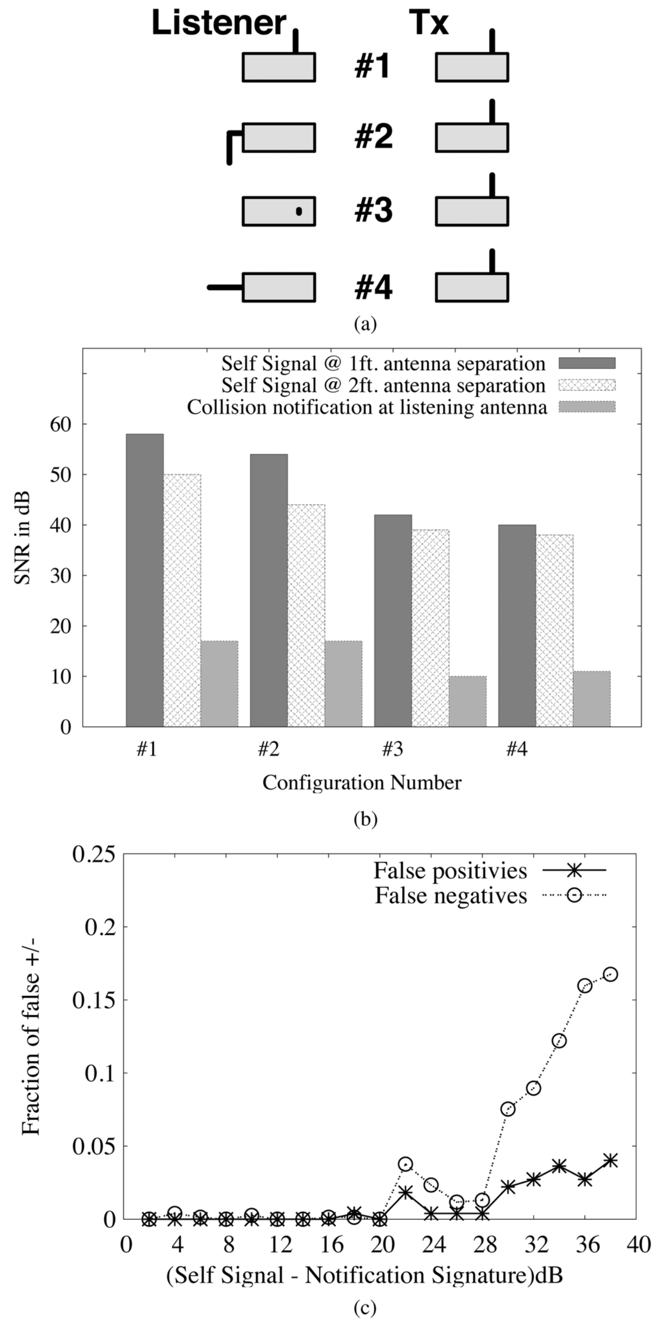


Fig. 12. Effect of listener antenna orientation on self-signal suppression. With proper orientation, the difference in signal strengths of self-signal and notification can be up to 36 dB (for less than 20% false detection). (a) Tested antenna orientations. (b) Signal strength between the transmit, listening antenna, as well as the receiver and listening antenna for the above orientations. (c) False positive/negative rates of discerning notification against varying transmitter–receiver separation with self-signal suppression for configuration 4. The original self-signal as in configuration 1 is 50 dB.

IV. Performance Evaluation

We have shown the robustness of notification detection at the transmitter in Section III. In this section, our experiments are designed to answer two questions on the performance of CSMA/CN: 1) What is the accuracy of detecting a collision at the receiver? 2) How much is CSMA/CN's throughput gain over 802.11 and PPR? We start with a brief description of the system implementation and then proceed to the results.

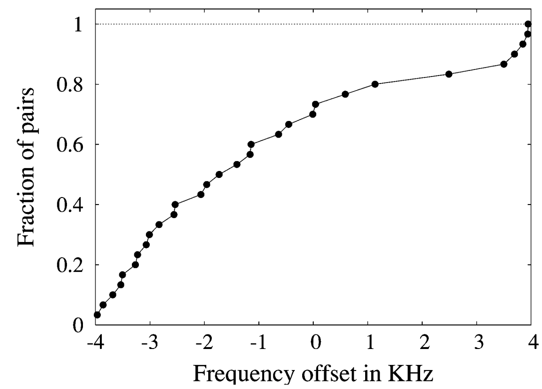


Fig. 13. Cumulative distribution function (cdf) of frequency offsets between all pairs of USRPs in our testbed.

TABLE I
False Positive Correlations Between a Pair of Signatures With Different Hamming Distances (Frequency Offset Difference Between the Signatures Is 0.4 kHz)

distance	16	32	48	64	80
falses	0.17	0.0992	0.0283	0.0221	0.016

TABLE II
False Positive Correlations Between Signatures With Different Frequency Offsets (Hamming Distance Between Them Is 48)

offset (KHz)	-2.029	-1.710	-3.262	0.636	3.939
falses	0.031	0.001	0.015	0.0212	0.037

A. Implementation

We have implemented CSMA/CN on a USRP testbed of 10 nodes. We used the GNU Radio framework with spread spectrum physical layer. Each USRP operates at 2.4 GHz with a sample rate of 2M samples/s. CSMA/CN uses BPSK and QPSK modulation schemes with convolution coding rate of $\frac{1}{2}$ and $\frac{3}{4}$, yielding four different bit rates. We incorporated the publicly available BCJR blocks of SoftPHY [1] along with signal correlation and collision detection logic in our codebase. The BCJR decoder outputs a log likelihood ratio (LLR) for each bit. SoftPHY bit error rate (BER) estimate was calculated from these LLR values as mentioned in [1].

B. Receiver-Side Collision Detection

We mentioned earlier that while receiving, the receiver employs preamble correlation to detect a new interference. However, not all interfering transmissions will cause collisions; the receiver needs to gain better confidence that a packet is truly failing. For this, the CSMA/CN receiver obtains physical-layer hints from SoftPHY [1], [2]. SoftPHY uses the output of BCJR [12] decoders to predict BER on a per symbol basis. We declare a bit is in error if the BER calculated by SoftPHY is more than a factor α than the BER of the clear frame. The factor α is chosen depending on the bit rate of the packet. The receiver then declares a collision if within a *window of 20 B* (from the point of preamble detection), SofPHY hints suggest that more than 30% bits have confidence less than α. Thus, if a preamble-correlation spike is followed by a train of low confidence symbols, the receiver stops reception and transmits the collision notification. Now, if the preamble itself is not detected, collisions may still occur. CSMA/CN therefore

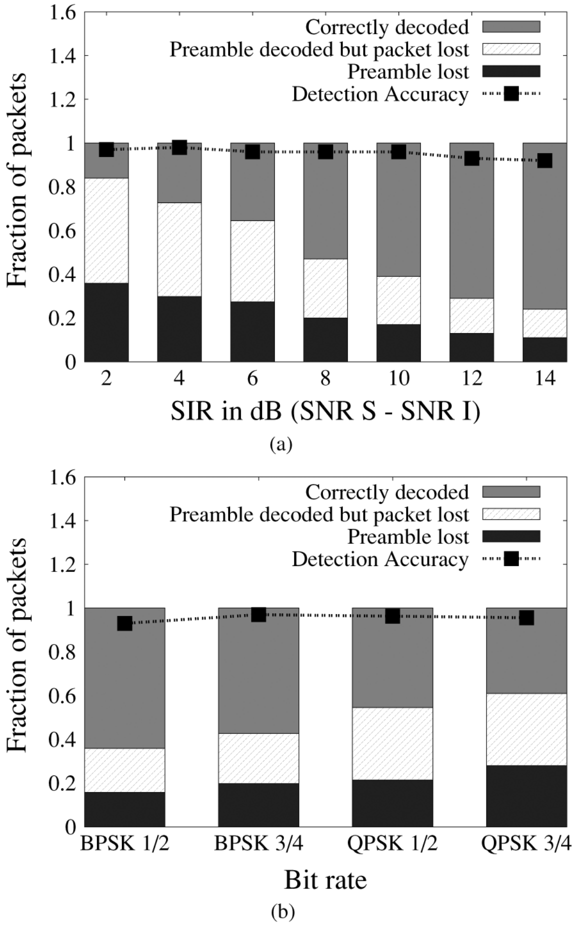


Fig. 14. CSMA/CN can detect a collision correctly (more than 92% of cases) even at high SIR and at all bit rates. The false positives are also negligible (1%). (a) Collision detection accuracy at different SIRs. (b) Collision detection accuracy at different bit rates.

continuously tracks the SoftPHY confidences, but uses a more conservative threshold to declare a collision.

To evaluate the accuracy of collision detection at the receiver, we set up a transmitter–receiver (T–R) pair and a moving interferer with backlogged traffic. All packets are 1500 B, and the bit rate is set to BPSK with $\frac{3}{4}$ coding. The T–R link delivers almost 100% of the packets without the interferer; any packet loss is mainly due to a collision. In the presence of interference, the packet is either: 1) decoded correctly; 2) received with errors (i.e., preamble decoded but packet lost); or 3) not received at all (preamble lost). Note that for a particular bit rate, links are attenuated until no higher rate is supported. Hence, our evaluation consists of both strong (high bit rate) and weak (low bit rate) links. Fig. 14(a) shows the breakup of these three cases with increasing signal-to-interference ratio (SIR) on the x-axis. Under such an interference condition, the dashed line in Fig. 14(a) shows CSMA/CN's collision detection accuracy. This accuracy is defined as the fraction of actual collisions detected by CSMA/CN. Evidently, when SIR is low, collision detection accuracy is close to 1. Even when SIR is high with 77% packets being successful (i.e., 23% collisions), CSMA/CN can still detect a collision correctly in 92% of the cases. Put differently, CSMA/CN fails to detect collisions with a low probability.

We also evaluated collision detection accuracy for different bit rates. Fig. 14(b) shows that CSMA/CN can detect most of the collisions at all bit rates. The accuracy per bit rate is derived from averaging over varying SIRs. We observed only a negligible number of false positives (1%) in the experiments. We believe this is tolerable for most practical purposes.

C. Throughput Evaluation

We now compare CSMA/CN's throughput against PPR [2] and a conventional scheme (we call it *802.11-like*). We discuss our experimental methodology followed by performance results.

Experimental Methodology: Software radios incur artificial delays in obtaining samples from the RF front end to the host program. Detecting a collision and transmitting back the notification will naturally include these delays. Thus, conducting a real-time evaluation of CSMA/CN is difficult. Hence, we resort to trace-based evaluation to study throughput gains with CSMA/CN. We set up random topologies with USRPs around our campus building. Each topology mimics three APs having 1–3 clients associated with them. Due to artificial communication delays between USRP and host computer, carrier sense incurs additional delays on USRPs [13]. This might cause unwarranted collisions giving unfair advantage to CSMA/CN. Thus, for a fair evaluation, we devise a methodology inspired by [4]. The basic idea is to extract traces from a testbed of laptops and then ask *what would happen if CSMA/CN-enabled USRPs were used instead of those laptops*.

To this end, we place a laptop at the position of each USRP. The laptops use power control to approximate the same topology as the USRPs would and perform regular carrier sensing. Also, for each AP–client link, the maximum possible bit rate is chosen at which the delivery ratios are consistently over 99%. Transmission bit rate is limited to 18 Mb/s to keep the modulation analogous to the corresponding USRP experiment using BPSK and QPSK. The experiments are performed at night in a static environment—this precludes interferers and allows for the chosen data rates to hold for longer timescales. Using this setup, we obtain the approximate interference map and use it later to generate collisions in our trace-based evaluation. The interference map is generated by taking pairs of APs, making them backlogged with traffic, and then making them transmit as with 802.11. The APs continue transmitting to specified clients until they drain out their entire backlogged traffic. We collect the traces at the clients and find the delivery ratio of the links in the presence of other interfering links (i.e., the other AP). This gives us the conditional probabilities of reception, i.e., with what probability will C1 successfully receive from AP1, if AP2 also transmits simultaneously. Equipped with this interference map, we repeat similar experiments on USRPs (but with carrier sensing turned off) to obtain collision detection probability for each pair of links.

We use the traces obtained from the previous experiment and emulate CSMA/CN. The APs are assumed to have 10 MB of data to be transmitted to each of its clients. Whenever there are concurrent transmissions, the emulator probabilistically determines whether a receiver experiences collision from the interference map of the network. In case of a collision, it transmits a collision notification, and the corresponding transmitter is aborted. We emulate backoff similar to 802.11, so the next random backoff for the colliding transmitter is chosen

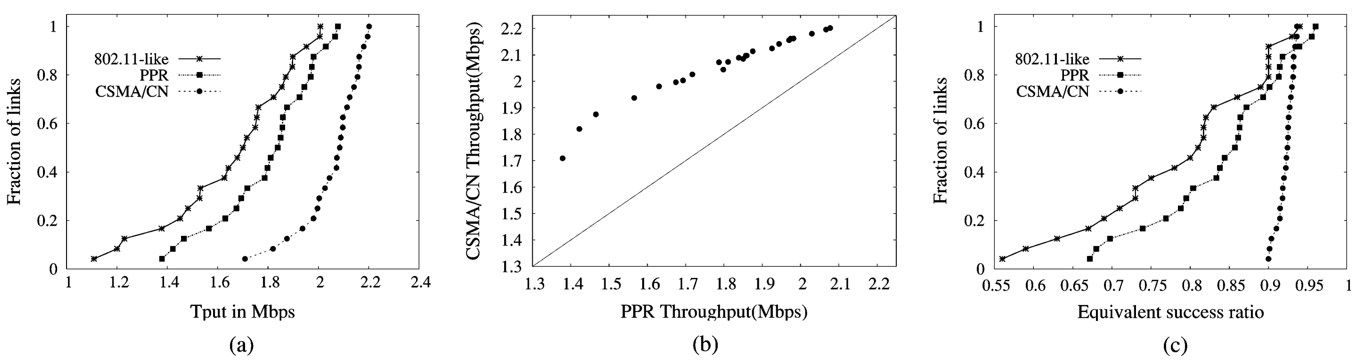


Fig. 15. Performance comparison of CSMA/CN scheme with 802.11-like and PPR schemes. The achieved throughput with CSMA/CN is significantly better than that with 802.11-like scheme. The relative throughput gain with CSMA/CN over PPR ranges from 10% to 30%. (a) Throughput per link. (b) Throughput of CSMA/CN versus PPR. (c) Fraction of successful bytes over the total number of transmitted bytes.

from a range that is double its previous value. Then, all the pending transmitters (including the just aborted transmitter) emulate carrier sensing and the backoff countdown. Thus, the transmitter with the lowest backoff will transmit next. Carrier sensing, backoff, DIFS, SIFS, and ACK time overheads are carefully accounted for between transmissions. This emulates (although with some approximation) what would have happened if CSMA/CN was running on the same network. We repeat similar emulation for 802.11 and PPR.

Performance Results: Fig. 15 compares the throughput of CSMA/CN scheme against PPR and 802.11-like schemes. Evidently, CSMA/CN offers improved throughput than the other two schemes. Specifically, from Fig. 15(a), we observe that around 80% of the links achieve more than 2 Mbps throughput with CSMA/CN. In contrast, 80% of the links obtain less than 2-Mb/s throughput under PPR and 802.11. Fig. 15(b) zooms into the comparative performance of CSMA/CN and PPR. Each dot on the graph corresponds to a link, and the x- and y-axis values correspond to throughputs achieved by PPR and CSMA/CN, respectively. Since CSMA/CN "prevents" a collision, instead of "recovering" from it like PPR, the throughput of the links improves. The relative improvement ranges from 10% to 30%.

We further analyze how PPR incurs relatively high retransmissions with respect to CSMA/CN. Fig. 15(c) shows the fraction of successful bytes over the total number of bytes transmitted for each scheme. Recall that CSMA/CN aborts transmission while PPR needs to retransmit the interfered chunk. Furthermore, when the interference starts first, PPR loses the entire packet. CSMA/CN, however, needs to retransmit only the bytes that were lost during the collision detection/notification operation. Thus, wasted transmissions are fewer in CSMA/CN, resulting in better overall throughput.

At higher transmission bit rates, collision notification will have a relatively higher overhead since it takes constant time. However, based on the example in Section II-C, at 54-Mb/s rate and 20-MHz bandwidth, this overhead amounts to less than 150 B. Therefore, when packets are of size 1500 B, aborting colliding transmissions is beneficial even at high bit rates. Moreover, there is an additional gain, particularly at high rates, from replacing the conventional ACK frame with the ACK signature. To understand CSMA/CN's gain at higher rates, we performed a custom simulation. The simulator does not model the detailed characteristics of the wireless channel and simulates only collisions of overlapping transmissions. We believe this is reasonable because our goal is to understand CSMA/CN's relative performance due to collisions induced by backoff and hidden terminals.

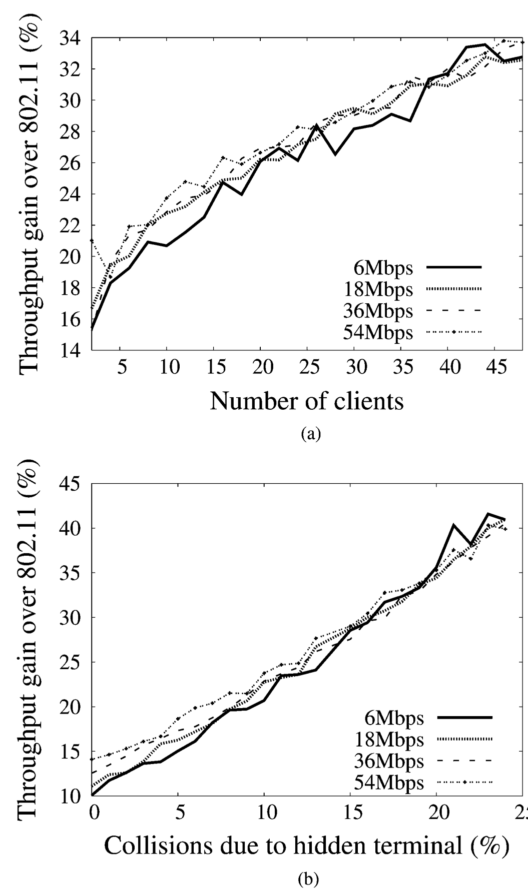


Fig. 16. Throughput gain with CSMA/CN over 802.11. (a) Varying number of nodes (collision probability 0.1). (b) Varying fraction of collisions (10-node topology).

Fig. 16(a) shows the performance improvement of CSMA/CN over 802.11 for increasing number of collisions due to hidden terminals. Noticeably, throughput gain is similar at various rates. In denser networks, backoff induced

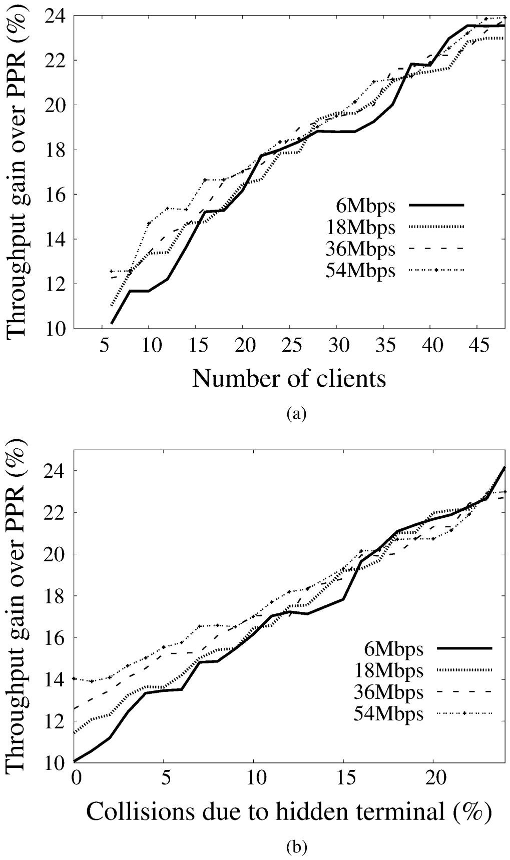


Fig. 17. Throughput gain with CSMA/CN over PPR. (a) Varying number of nodes (collision probability 0.1). (b) Varying fraction of collisions (10-node topology).

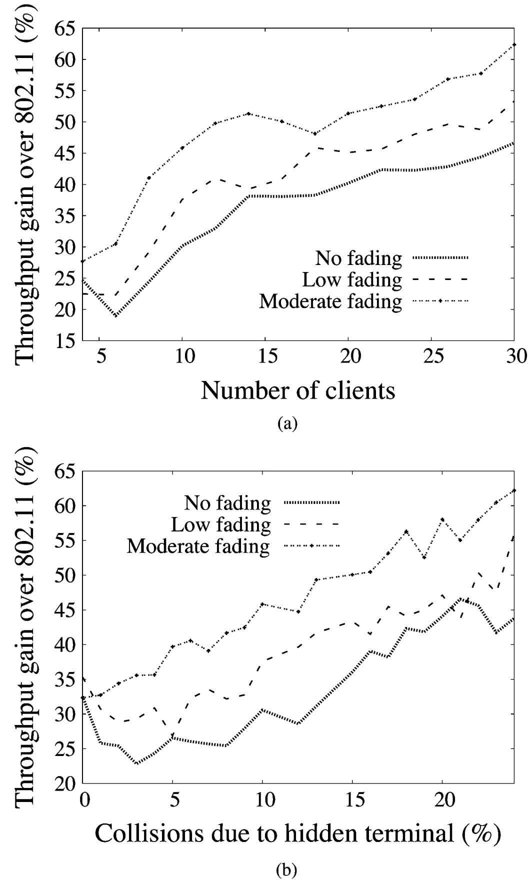


Fig. 18. Gain with CSMA/CN over 802.11 using ARF for rate adaptation. (a) Varying number of nodes (collision probability 0.1). (b) Varying fraction of collisions (10-node topology).

collisions (when multiple nodes choose the same backoff) will be higher [14]. Thus, the throughput gain over 802.11 increases with the increasing number of nodes, as in Fig. 16(b). We have also compared the performance of CSMA/CN with PPR in Fig. 17(a) and (b). It is evident that, though the throughout gains over PPR are less than that over 802.11, gains are significant and trends are similar.

CSMA/CN can potentially provide further throughput improvement when used in conjunction with a bit-rate adaptation scheme. Commonly deployed rate adaptation schemes like Auto Rate Fallback (ARF) reduce bit rate in response to packet loss. Ideally, bit rate should be decreased only when the loss is due to fading, but not in case of a collision. CSMA/CN can reliably detect collisions and hence can aid bit-rate adaptation. Fig. 18 shows throughput gains with CSMA/CN over 802.11 with rate adaptation using ARF. CSMA/CN improves performance further because it can correctly prescribe when the bit-rate adaptation scheme should reduce rate. Moreover, with higher channel fading, the relative gain with CSM/CN is larger. This is because CSMA/CN doubles backoff contention window only in case of collisions, but not losses due to fading. 802.11 will unnecessarily back off even when a packet is lost due to fading and consequently yields lower throughput.

With 802.11n rates, the overhead with CSMA/CN is relatively high compared to the air time of an individual frame. However, note that 802.11n employs frame aggregation, which combines multiple frames into a single transmission. These frames are acknowledged as a block only at the end of the aggregate frame transmission. Relative to that aggregate frame, the overhead of CSMA/CN is rather insignificant and potential gain is quite significant. Therefore, we believe CSMA/CN is beneficial even in 802.11n networks.

V. Issues and Discussions

We discuss some of the limitations and opportunities with CSMA/CN that remain unaddressed in this paper.

1) Can CSMA/CN be used in conjunction with multiple-input–multiple-output (MIMO)? This design and implementation of this paper assumes single-input–single-output (SISO) communication. Nevertheless, we hypothesize that with a better analog-to-digital (A/D) converter, the same listener logic can be shared by multiple antennas in a MIMO system. Each of the antennas can submit a distinct signature corresponding to its respective receiver. The listener logic can then execute the correlation (with each of the signatures) in parallel and abort the appropriate transmit antenna. If the cost of employing multiple correlation logic is a concern, the correlation can be performed serially, at the expense of a longer turn around time. We leave the implementation of such a system for future work.

2) Why not use tones to abort transmission (in the spirit of the DBTMA protocol [15])? Supporting $O(n)$ frequency tones will require additional channel resources. Excessively narrowband tones are prone to fading; tones also need to be separated by a guard band to cope with nonideal filters. Even though feasible, the aggregate bandwidth investment for a tone-based CSMA/CN may lead to channel wastage.

3) Can exposed terminals be addressed with signal correlation? The CMAP [16] proposal addresses the exposed terminal problem by estimating an interference map among neighboring links. If exposed terminals identify that their respective transmissions can be accomplished in parallel, they carry out the transmission. The interference map changes over time, and links expected to be parallel can mutually interfere. In such a case, the receiver of the failing packet can immediately abort its transmitter. More generally, CSMA/CN is a primitive that enables a variety of protocol possibilities—exposed terminals via CMAP like schemes is one of them.

4) Can CSMA/CN be applicable to broadcast settings? Wireless broadcast/multicast protocols traditionally suffer from the problem of excessive ACK overhead. CSMA/CN may resolve this problem if unique signatures can be assigned to each of the clients. So long as there are modest number of clients, the transmitter can continuously track how many clients are encountering collisions, and abort accordingly.

5) How do the neighbors of a transmitter utilize the channel soon after it aborts the transmission? In 802.11, exposed terminals will hear the PLCP header of an ongoing transmission and will set their NAV. This will prevent them from transmitting although the channel is cleared when the ongoing transmission is aborted. We argue that carrier sense is sufficient for CSMA/CN, and NAV is not necessary. Even without NAV, ACK signature (which is much shorter than the conventional ACK frame) can safely arrive at the sender. This is because an exposed terminal (around the sender) will carrier sense for DIFS duration before transmission. Since the ACK signature will appear within that interval, it will not interfere with ACK correlation at the sender. Thus, neighboring terminals of an aborted transmission need not wait for NAV and can occupy the channel as soon as it is clear.

6) Are there any additional incentives to deploy CSMA/CN? This paper is a first step toward adopting collision detection in wireless networks and is certainly amenable to various improvements. Yet, even this first step provides several potential secondary benefits. Past research shows transmission bit rate should not be reduced due to collisions and should only cater to fading [1], [17], [18]. CSMA/CN will aid rate adaptation with sound collision detection. Since correlation is more robust than decoding, the ACK loss in 802.11 can be mitigated by using signatures. 802.11 is by design conservative to prevent collisions. CSMA/CN has a low penalty due to collision, and thus it provides network administrators an opportunity to be more aggressive with carrier sense threshold, backoff, etc., potentially yielding higher network throughput.

VI. Related Work

Avoiding Collisions: There have been numerous MAC protocols proposed for wireless networks [19]. A common feature of most of these schemes is that they avoid collisions by utilizing control frames or out-of-band busy tones. These schemes tend to be either quite conservative by reserving a large space around the communicating nodes, or do not completely eliminate the collisions. Some studies have shown that enabling RTS-CTS, to avoid collisions, reduces the overall throughput in practice [20] and is hence disabled by default in many deployments [21]. Recently, several schemes use the knowledge of interference map to schedule transmissions intelligently [16], [22], [23]. Interference relationships vary with time and hence are difficult to monitor.

Recovering From Collisions: Apart from PPR mentioned earlier, ZipTx [3] and Maranello [24] make use of known pilot bits to detect errors and recover the partial packets. We do not insert any known bits for detecting collisions. A receiver could apply interference cancellation [6] to recover the frame of interest by decoding the interfering transmission first and then canceling it out. However, this approach works only when the relative strengths of the signals at the receiver satisfy certain thresholds. CSMA/CN also employs interference cancellation, but at the transmitter for the purpose of suppressing self-signal and strengthening signature correlation. ZigZag decoding [4] is a form of interference cancellation that recovers frames from repeated collisions. While this is a creative approach, it requires that the same set of frames be involved in multiple collisions. Similarly, ANC [25] requires the knowledge of one of the packets involved in a collision.

Detecting Collisions: Reference [17] enables a transmitter to distinguish between a fading and collision by having the receiver return the received bits. SoftRate [1] utilizes SoftPHY information to distinguish between collision and fading for rate adaptation. AccuRate [26] detects collisions by comparing constellation dispersions of preamble and postamble. In contrast, CSMA/CN detects and aborts collisions on the fly.

Aborting Collisions: A scheme that bears some similarity with CSMA/CN is [27]. The authors use an out-of-band control channel to transmit pulses for the purpose of indicating active transmissions. Transmitters sense the control channel to detect potential collisions, however such decisions at the transmitter are not an accurate indicator of collision at the receiver. CSMA/CN uses an *in-band* collision detection scheme at the receiver with explicit feedback to the transmitter to abort. Our previous work [28] is limited in its ability to detect a collision notification. This paper performs self-signal suppression through interference cancellation and antenna orientation making CSMA/CN suitable for distant transmitter–receiver pairs that are more vulnerable to collisions.

Suppressing Self-Signal: As mentioned above, there are several schemes proposed earlier for interference cancellation [6], [29], particularly for the case when the interfering bits are known [4], [10], [25], [30]. Self-signal is a special case of known interference where the receiver and the interferer antennas are both attached to the same node. While this scenario makes the interference quite strong and cancellation relatively hard, it also provides opportunities for suppressing self-signal with appropriate antenna placement. A recently proposed approach for full duplex communication [31] places two transmitting antennas such that their relative distances to the receiving antenna differ by half the carrier wavelength. This allows for a natural suppression of the self-signal at the receiver's antenna. Such an antenna placement would also benefit CSMA/CN. However, a pertinent question then is why not employ full duplex communication instead of CSMA/CN. Note that *correlation* is sufficient for detecting notification, whereas decoding is necessary for full duplex communication. Given that the self-signal is much stronger than the received signal, and that there could be several other signals in the environment, correlation is more robust than decoding. This robustness is critical, especially when the notification signal is weak, i.e., when the receiver is far away from the transmitter.

VII. Conclusion

CSMA/CN is an attempt to approximate CSMA/CD in wireless networks. We show that it is feasible to abort an unsuccessful transmission with the aid of a collision notification from the receiver. Techniques from signal correlation and SoftPHY-based hints are employed to this end. We believe that the proposed architecture is simple, the additional hardware requirements are tolerable, and the performance improvements are worthwhile. Perhaps more importantly, CSMA/CN is only one example of how signal correlation can be exploited in wireless systems. Exploring the possibilities across the protocol stack is an open area for future research.

References

[1] M. Vutukuru, H. Balakrishnan, and K. Jamieson, "Cross-layer wireless bit rate adaptation," in *Proc. ACM SIGCOMM*, 2009, pp. 3–14.

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

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

[4] S. Gollakota and D. Katabi, "Zigzag decoding: Combating hidden terminals in wireless networks," in *Proc. ACM SIGCOMM*, 2008, pp. 159–170.

[5] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *Proc. ACM MobiCom*, 2008, pp. 116–127.

[6] D. Halperin, T. Anderson, and D. Wetherall, "Taking the sting out of carrier sense: Interference cancellation for wireless LANs," in *Proc. ACM MobiCom*, 2008, pp. 339–350.

[7] K. Tan, H. Liu, J. Fang, W. Wang, J. Zhang, M. Chen, and G. M. Voelker, "SAM: Enabling practical spatial multiple access in wireless LAN," in *Proc. ACM MobiCom*, 2009, pp. 49–60.

[8] S. Kakumanu and R. Sivakumar, "Glia: A practical solution for effective high datarate wifi-arrays," in *Proc. ACM MobiCom*, 2009, pp. 229–240.

[9] "Quellan raises signal strength bars with newest noise canceller," Intersil, Milpitas, CA, 2009 [Online]. Available: http://www.quellan.com/pressroom/press_releases/pr010809-raises-signal-strength.html

[10] B. Radunovic, D. Gunawardena, P. Key, A. Proutiere, N. Singh, H. V. Balan, G. Dejean, and G. DeJean, "Rethinking indoor wireless: Low power, low frequency, full-duplex," in *Proc. IEEE WiMesh*, 2010, pp. 1–6.

[11] E. Gebara, E. M. Tentzeris, and J. Laskar, "Analysis and design of an interference canceller for collocated radios," *IEEE Trans. Microw. Theory Tech.*, vol. 53, no. 11, pp. 3498–3508, Nov. 2005.

[12] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. IT-20, no. 2, pp. 284–287, Mar. 1974.

[13] G. Nychis, T. Hottelier, Z. Yang, S. Seshan, and P. Steenkiste, "Enabling MAC protocol implementations on software-defined radios," in *Proc. USENIX NSDI*, 2009, pp. 91–105.

[14] K. Tan, J. Fang, Y. Zhang, S. Chen, L. Shi, J. Zhang, and Y. Zhang, "Fine-grained channel access in wireless LAN," in *Proc. ACM SIGCOMM*, 2010, pp. 147–158.

[15] J. Deng and Z. Haas, "Dual busy tone multiple access (DBTMA): A new medium access control for packet radio networks," in *Proc. IEEE ICUPC*, 1998, vol. 2, pp. 973–977.

[16] M. Vutukuru, K. Jamieson, and H. Balakrishnan, "Harnessing exposed terminals in wireless networks," in *Proc. 5th USENIX NSDI*, 2008, pp. 59–72.

[17] S. Rayanchu, A. Mishra, D. Agrawal, S. Saha, and S. Banerjee, "Diagnosing wireless packet losses in 802.11: Separating collision from weak signal," in *Proc. IEEE INFOCOM*, 2008, pp. 735–743.

[18] P. A. K. Acharya, A. Sharma, E. M. Belding, K. C. Almeroth, and K. Papagiannaki, "Congestion-aware rate adaptation in wireless networks: A measurement-driven approach," in *Proc. IEEE SECON*, 2008, pp. 1–9.

[19] S. Kumar, V. S. Raghavan, and J. Deng, "Medium access control protocols for ad hoc wireless networks: A survey," *Ad Hoc Netw.*, vol. 4, no. 3, pp. 326–358, May 2006.

[20] "802.11 WLAN performance," Atheros, Santa Clara, CA, 2003.

[21] *"Wireless LAN Adapter User Guide,"* Broadcom, Irvine, CA, 2005.

[22] J. Manweiler, N. Santhapuri, S. Sen, R. R. Choudhury, S. Nelakuditi, and K. Munagala, "Order matters: Transmission reordering in wireless networks," in *Proc. ACM MobiCom*, 2009, pp. 61–72.

[23] V. Shrivastava, N. Ahmed, S. Rayanchu, S. Banerjee, S. Keshav, K. Papagiannaki, and A. Mishra, "CENTAUR: Realizing the full potential of centralized WLANs through a hybrid data path," in *Proc. ACM MobiCom*, 2009, pp. 297–308.

[24] B. Han, A. Schulman, F. Gringoli, N. Spring, B. Bhattacharjee, L. Nava, L. Ji, S. Lee, and R. Miller, "Maranello: Practical partial packet recovery for 802.11," in *Proc. USENIX NSDI*, 2009, p. 14.

[25] S. Katti, S. Gollakota, and D. Katabi, "Embracing wireless interference: Analog network coding," in *Proc. ACM SIGCOMM*, 2007, pp. 397–408.

[26] S. Sen, N. Santhapuri, R. R. Choudhury, and S. Nelakuditi, "AccuRate: Constellation based rate estimation in wireless networks," in *Proc. USENIX NSDI*, 2010, p. 12.

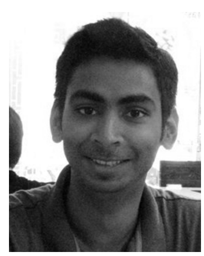
[27] J. Peng, L. Cheng, and B. Sikdar, "A wireless MAC protocol with collision detection," *IEEE Trans. Mobile Comput.*, vol. 6, no. 12, pp. 1357–1369, Dec. 2007.

[28] S. Sen, N. Santhapuri, R. R. Choudhury, and S. Nelakuditi, "Moving away from collision avoidance: Aborting collisions in wireless networks," in *Proc. ACM HotNets*, 2009.

[29] S. Verdu, *Multiuser Detection*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[30] C. Qin, N. Santhapuri, S. Sen, and S. Nelakuditi, "Known interference cancellation: Resolving collisions due to repeated transmissions," in *Proc. IEEE WiMesh*, 2010, pp. 1–6.

[31] J. Choi, M. Jain, K. Srinivasan, P. Levis, and S. Katti, "Achieving single channel, full duplex wireless communication," in *Proc. ACM MobiCom*, 2010, pp. 1–12.



Souvik Sen (S'11) received the B.E. degree in computer science from Bengal Engineering and Science University, Howrah, India, in 2007, and the M.S. degree in computer science from Duke University, Durham, NC, in 2011, and is currently pursuing the Ph.D. degree in computer science at Duke University.

His research interests include wireless networking and mobile computing.

Romit Roy Choudhury (M'03–SM'10) received the Ph.D. degree in computer science from the University of Illinois at Urbana–Champaign in 2006.

He is an Associate Professor of electrical and computer engineering and computer science with Duke University, Durham, NC, which he in 2006 after completing his Ph.D. studies. His research interests are in wireless protocol design mainly at the PHY/MAC layer and in mobile social computing at the application layer.

Dr. Roy Choudhury received the NSF CAREER Award in January 2008.

Srihari Nelakuditi (M'01) received the Ph.D. degree in computer science and engineering from the University of Minnesota, Minneapolis, in 2001.

He is currently an Associate Professor with the University of South Carolina, Columbia. Earlier, he worked as a Software Design Engineer with Texas Instruments, Bangalore, India. His research interests are in resilient routing, wireless networking, and mobile computing.

Dr. Nelakuditi was a recipient of the NSF CAREER Award in 2005.