CSMA/CN: Carrier Sense Multiple Access with Collision Notification

Souvik Sen
Duke University
Durham, NC, USA
ssen@cs.duke.edu

Romit Roy Choudhury
Duke University
Durham, NC, USA
romit@ee.duke.edu

Srihari Nelakuditi
University of South Carolina
Columbia, SC, USA
srihari@cse.sc.edu

ABSTRACT

A wireless transmitter learns of a packet loss, infers collision, only after completing the entire transmission. If the transmitter could detect the collision early (such as with 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 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 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/GNURadios demonstrates the feasibility and effectiveness of CSMA/CN.

Categories and Subject Descriptors

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

General Terms

Design, Experimentation, Performance

Keywords

Wireless, Collision Detection, Cross-Layer, CSMA/CD

1. 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.

CSMA/CD is considered infeasible in wireless networks for 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 both 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 (Figure 1). Once communication begins, the receiver exploits *preamble correlation* to detect the presence of an interference. Realizing that the packet is likely to fail, the receiver checks the confidence of incoming bits via physical layer hints from SoftPHY [2, 1]. 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 nearby transmitters.

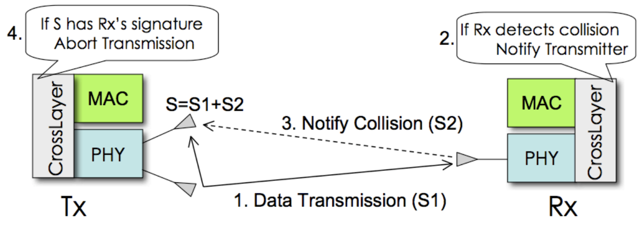

4. If S has Rx's signature Abort Transmission
2. If Rx detects collision Notify Transmitter
CrossLayer
MAC
PHY
S=S1+S2
3. Notify Collision (S2)
1. Data Transmission (S1)
Tx
Rx

Figure 1: Basic structure of CSMA/CN.

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 6.

(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 errors are likely, and retransmitting all of them can be wasteful. Collision notification aborts transmission of a colliding packet. The intuition is that aborting (or prevention) is better than recovery (or cure). The rest of the packet 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 SINR far less than 0dB) 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. Besides, 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 towards 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 are as follows.

- **Identify a middle ground between CSMA/CD and CA.** CSMA/CN is an early attempt to rethink medium access control 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: (i) modeling and subtracting the wireless self-signal, and (ii) 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/GnuRadio nodes.** Experimental results show consistent throughput improvements over 802.11 and PPR. We identify several avenues of further research.

The next section 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 the following section. Thereafter, we present the performance evaluation, limitations and opportunities, related work, and finally, the conclusions.

2. 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) A wireless transmitter T cannot reliably detect the collision on its own; the receiver R must get involved. (2) 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.

2.1 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 (Figure 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.

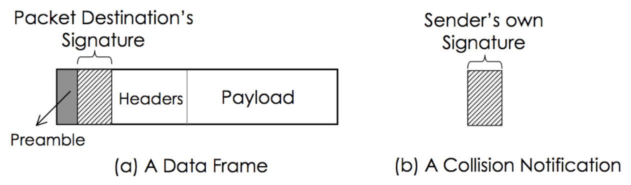


Figure 2: CSMA/CN frame formats. Transmitter T inserts receiver R's signature after the packet's preamble. Collision notification from R consists of only its own signature.

Collision happens when a nearby transmitter $T1$ interferes with R's reception, causing packet corruption (Figure 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; the reception may sometimes succeed 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 is able to infer whether the packet is expected to get corrupted. If so, R halts reception, and prepares to send a collision notification to transmitter T.

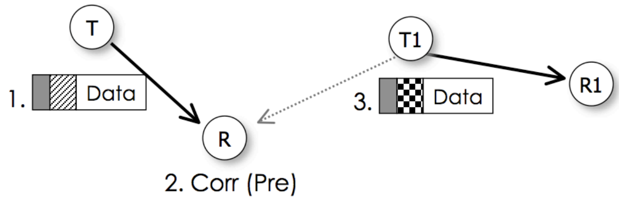


Figure 3: While receiving from T, the receiver R searches for a preamble via correlation (denoted Corr(Pre)).

Now, if the interferer $T1$ starts first, and the transmission from T starts later, R may need to abort T (Figure 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.

2.2 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 (Figures 2(a) and 2(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 (Figure 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.

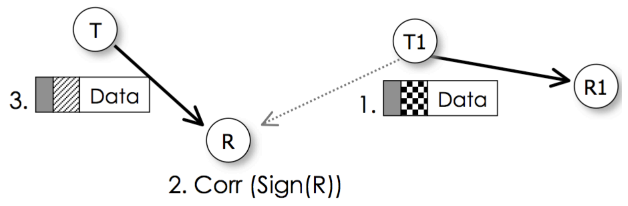


Figure 4: While overhearing from $T1$, R searches for its own signature via correlation (denoted Corr(Sign(R))).

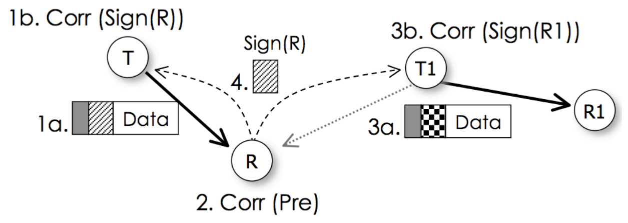


Figure 5: During their respective transmissions, T searches for R's signature whereas $T1$ searches for $R1$'s signature. Hence, when R sends a collision notification, only T aborts its transmission.

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.

2.3 Packet Resumption and Acknowledgment

Under CSMA/CN, the transmitter does not retransmit the entire aborted packet. Instead, it resumes transmission from byte B_{re}, where B_{re} indicates the maximum in-sequence byte received correctly by the receiver. The value of B_{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_{now}. Then the estimate of $B_{re} = B_{now} - B_{out}$ bytes, where B_{out} is determined based on the transmission bitrate. For example, in our design, collision detection takes time equivalent to 20 bytes. The time for notification signature of 20 bytes using BPSK over 20 MHz bandwidth is $8\mu s$. So the turnaround time for notification including the SIFS interval of 10μs would be 18μs. This corresponds to 122 bytes at 54 Mbps. Including the collision detection overhead of 20 bytes, a conservative estimate of B_{out} would be 150 bytes. Hence, if a sender is transmitting a 1500 byte packet and aborts transmission at 751th (B_{now}) byte, it will retransmit from 601th (B_{re}) byte. Once the packet is transmitted, the CSMA/CN receiver responds with an ACK when the packet 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 pseudo codes capture the core flow of operations under CSMA/CN.

Algorithm 1 : T.transmit(R, Data)

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

Algorithm 2 : R.receive()

```
1: if frame of interest is already being received then
2:     if Corr(Pre) high and many bits suspect then
3:         Transmit <Sign(R)>
4: if interfering frame is being received then
5:     if Corr(Sign(R)) high then
6:         Transmit <Sign(R)>
7: if frame of interest reception successful then
8:     Transmit <Sign(ACK(R))>
```

Points of Discussion

(1) A pertinent concern 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 its 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 4.

(2) Another question is *what happens when two transmitters send to the same receiver?* 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, depend on signatures, *how many distinct signatures do we need for 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.

3. 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.

3.1 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 n^{th} 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 the value of $\mathcal{C}(s,y,p)$, 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., $y[n] = Hx[n]e^{j2\pi n\delta fT} + w[n]$. 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

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

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

Figure 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

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

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

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

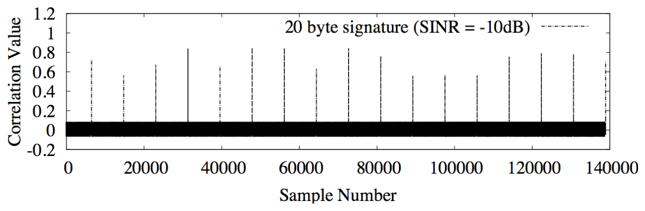


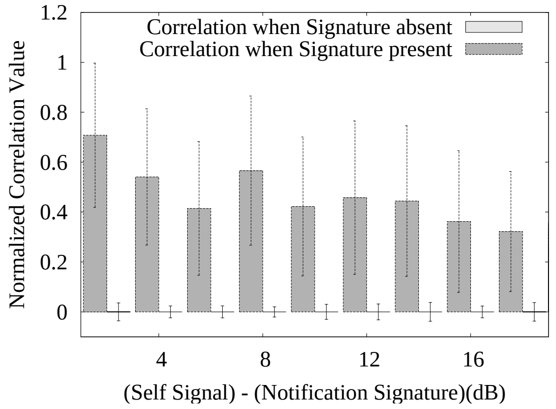
Figure 6: Correlation spikes whenever a known pattern arrives amid a (10dB stronger) background transmission.

We normalize the correlation value to $\frac{\mathbb{R}(\mathcal{C}(s,y,\hat{p}))}{\Sigma_{k=1}^{L}|s[k]|}$ and detect the spikes by thresholding it based on the signal strength of s. For CSMA/CN, low false positive is desirable, even at the expense of slightly higher false negatives. Evident from Figure 6, the higher the normalized correlation value, the easier it is to identify the pattern without false positives/negatives.

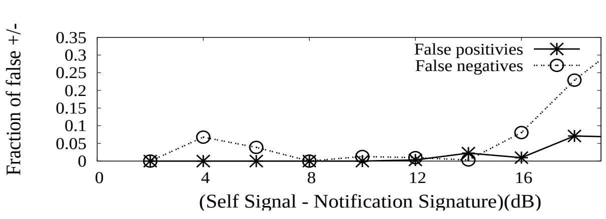
Thus, while receiving a packet from T, R can continuously search for new interference by making s = "universally known preamble". If the interference starts first, R can search for T's transmission by making s = "R's own signature". Thus, signal correlation allows a receiver to promptly detect a collision.

3.2 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. Figure 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 strength of self-signal is 50dB. The self-signal was kept at 50dB to prevent the ADC from saturation. The listening and transmitting antennas are separated by 2 ft. The size of the notification signature is 20 bytes (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 16dB), 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.



(a) Correlation when signature is present/absent



(b) False positive/negative rates of discerning notification.

Figure 7: Varying transmitter-receiver separation: Correlation values when signature is absent are quite small and thus not visible. Total false detection (positive and negative) would be higher than 20% if notification is weaker than self-signal by greater than 16dB.

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, 5, 6]. 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 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, 6]. The listener knows the bits being sent and thus the transmitted signal $x[n]$. The received signal $y[n]$ is the result of several effects such as channel distortion, sampling offset, and frequency offset on the transmitted signal $x[n]$. We elaborate on the estimation of these effects below:

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.

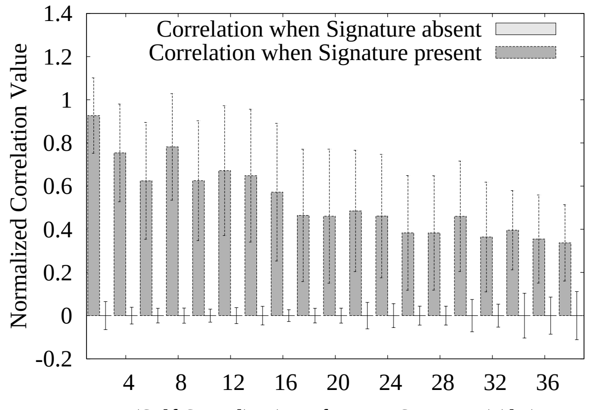
Sampling offset: Since the transmitters and the receiver's clocks are not synchronized, the receiver may not sample at the ideal sampling points. A practical wireless receiver tracks the sampling offset (τ) and performs interpolation to estimate the "ideal" sampled points ($x[n]$). Using the same estimate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

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

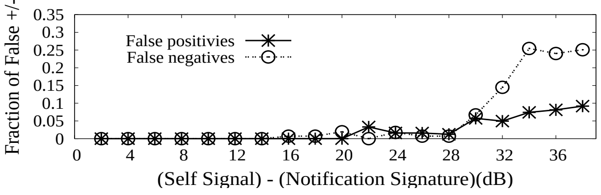
Frequency offset: CSMA/CN uses the receiver's estimate of frequency offset δf. Frequency offset is relatively stable and does not change for long durations. To incorporate the effect of frequency offset, the n^{th} transmitted sample is phase shifted by a factor of $e^{2\pi n\delta fT}$ where T is the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

Filter and Inter-Symbol 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 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. Figure 8 shows that, with this approach, notification can be reliably detected even when it is weaker than self-signal by 32dB (as opposed to only 16dB without self-signal suppression).



(a) Correlation when signature is present/absent

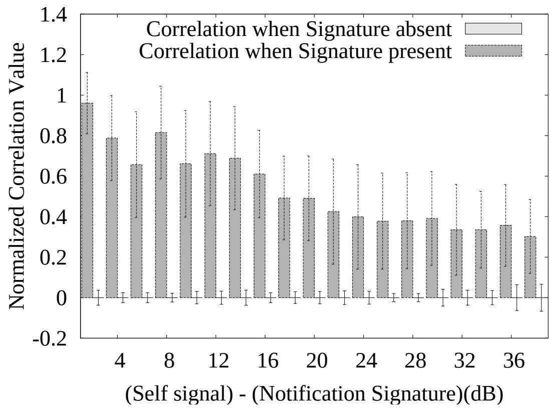


(b) False positive/negative rates of discerning notification

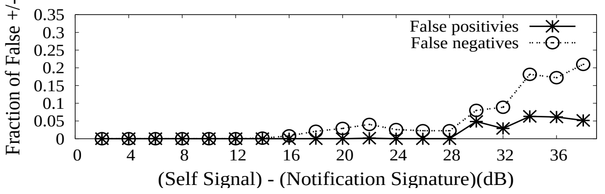
Figure 8: Self-signal suppression over wireless with varying transmitter-receiver separation: The difference in signal strengths of self-signal and notification can be as high as 32dB (total false detection <20%).

3.3 Self-Signal Suppression over the Wire

Since the signal we are trying to cancel is a self-signal, this signal can be passed on to the listening interface (from the transmit interface) over a physical wire. The advantage is that the signal received over the wire will have the same filter distortion and frequency offset effects, as that of the wireless signal. This precludes the need to model these effects separately. 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. 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. Figure 9 shows that, when assisted by wired communication, the difference in signal strengths of self-signal and the notification can be up to 34dB for reliable detection (compared to 32dB with self-signal suppression over wireless).



(a) Correlation when signature is present/absent

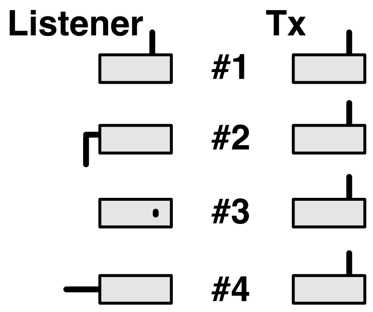


(b) False positive/negative rates of discerning notification

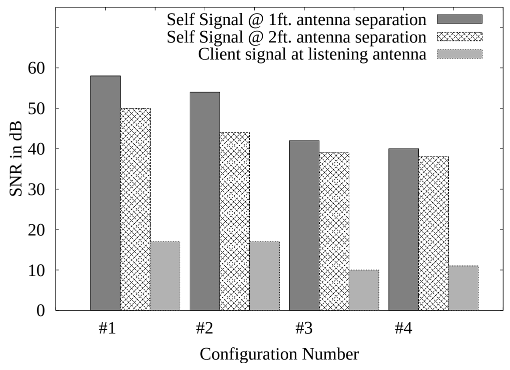
Figure 9: Self-signal suppression over the wire with varying transmitter-receiver separation: The difference in signal strengths of self-signal and notification can be up to 34dB (for less than 20% false detection).

3.4 Listener Antenna Orientation

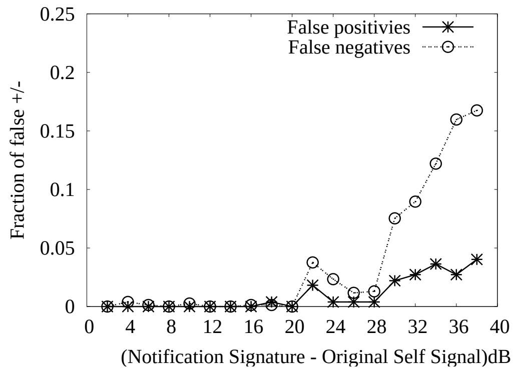
The separation between antennas and their orientations influence the relative strengths of the self-signal and the notification. The self-signal is 50 dB stronger at a 2ft separation. Since both the antennas have to be packaged on the same access point, their maximum separation is limited. However, Figures 10 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*. Figure 10(b) shows that when the original self-signal was 55dB, the self-signal reduces by 12dB while the notification strength reduces by 6dB. In all our orientations, the transmit antenna is always placed upright to have minimum impact on the original transmission. Figure 10(c) shows that by positioning the listening antenna correctly and performing self-signal suppression, it is possible to correlate notifications for clients that are 36dB weaker than the self-signal.



(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 50dB.

Figure 10: The 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 36dB (for less than 20% false detection).

Table 1: False positive correlations between signatures with different hamming distances (frequency offset difference between them is 0.4 KHz)

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

Table 2: 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

We now discuss how, in future, we can further suppress the self-signal to make the CSMA/CN scheme quite practical. With a reasonable antenna separation of 1 feet, modern APs will have a self-signal of 65dB [7]. Our experimentation shows that we gain 15dB from correlation, 20dB from digital interference cancellation. Also, antenna orientation gives us a gain of 10dB with 1 feet separation 10(b). Combining these, using only digital cancellation, we can offset a self-signal of 40dB. We believe the remaining 25dB can be handled using analog cancellation. There exist chipsets for analog cancellation that can cancel upto 30dB of self-interference [8, 9, 10]. Antenna placement, as a function of the signal wavelength can further benefit CSMA/CN [11].

3.5 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 3.1, there exists an inherent difference in the center frequency offsets for wireless radios (Figure 11 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 [12]. In other words, the frequency offset naturally creates some "dissimilarity" between the signatures, helping in keeping the signature short.

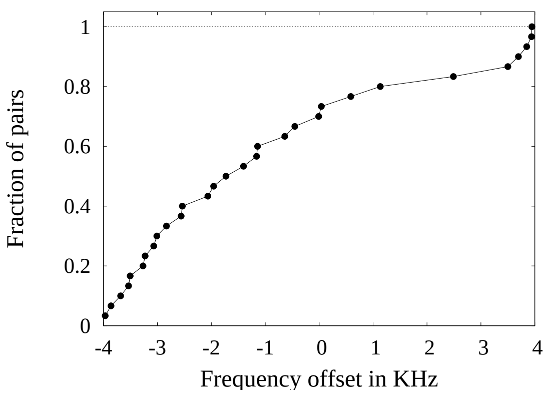


Figure 11: CDF of frequency offsets between all pairs of USRP nodes in our testbed.

We studied *whether different signatures may induce similar correlations.* Figure 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 1 and 2 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 bits 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.

4. PERFORMANCE EVALUATION

We have shown the robustness of notification detection at the transmitter in Section 3. 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.

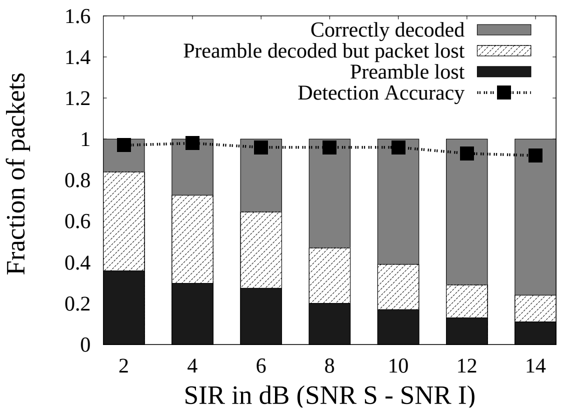
4.1 Implementation

We have implemented CSMA/CN on a USRP testbed of 10 nodes. We used the GNURadio framework with spread spectrum physical layer. Each USRP operates at 2.4 GHz with a sample rate of 2M samples/sec. CSMA/CN uses BPSK and QPSK modulation schemes with convolution coding rate of $\frac{1}{2}$ and $\frac{3}{4}$, yielding 4 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 BER estimate was calculated from these LLR values as mentioned in [1].

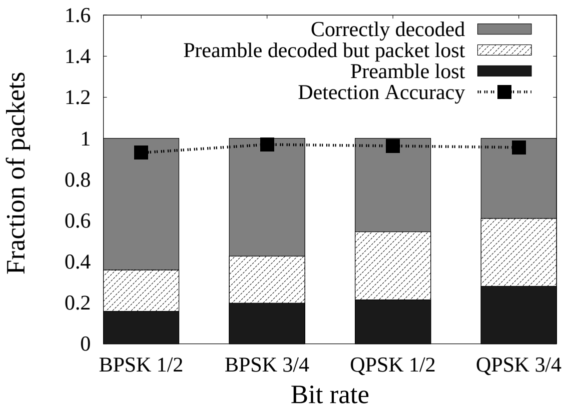
4.2 Receiver-side Collision Detection

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 [2, 1]. SoftPHY uses the output of BCJR [13] 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 bytes* (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 continuously tracks the SoftPHY confidences, but uses a more aggressive 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 bytes and the bitrate 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). Figure 12(a) shows the break-up of these 3 cases with increasing SIR (signal-to-interference-ratio) on the x-axis. Under such an interference condition, the dashed line in Figure 12(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.



(a) Collision detection accuracy at different SIRs



(b) Collision detection accuracy at different birates

Figure 12: CSMA/CN can detect a collision correctly (more than 92% of cases) even at high SIR and at all birates. The false positives are also negligible (1%).

We also evaluated collision detection accuracy for different bit-rates. Figure 12(b) shows that CSMA/CN can detect most of the collisions at all bit-rates. The accuracy per-bitrate 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.

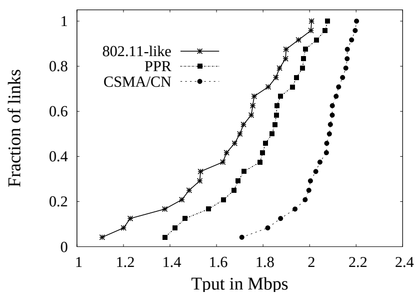
4.3 Throughput Evaluation

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

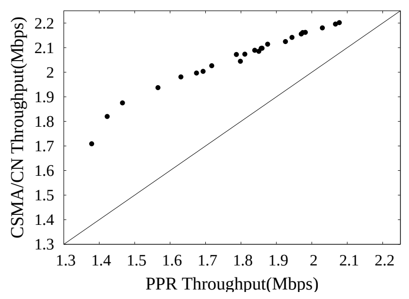
Experimental Methodology: Software radios incur artificial delays in obtaining samples from the RF front-end to the host program. Detecting a collision and transmitting back the notification will naturally include these delays, Thus, conducting a real time evaluation of CSMA/CN is difficult. Hence, we resort to trace-based evaluation to study throughput gains with CSMA/CN. We setup random topologies with USRPs around our campus building. Each topology mimics 3 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 [14]. 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 bitrate is chosen at which the delivery ratios are consistently over 99%. Transmission bitrate is limited to 18Mbps 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 time scales. Using this set up, 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.

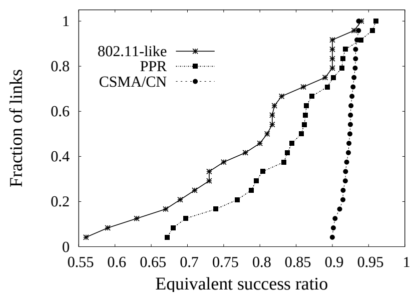
We use the traces obtained from the previous experiment and emulate CSMA/CN. The APs are assumed to have 10 MB of data to be transmitted to each of its clients. The emulator probabilistically determines collision from the interference map of the network. Depending on the detection accuracy of the link, it stops the failed link, transmits a notification and starts the next transmission earlier. 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 for comparison with CSMA/CN.



(a) Throughput per link



(b) Throughput of CSMA/CN vs PPR



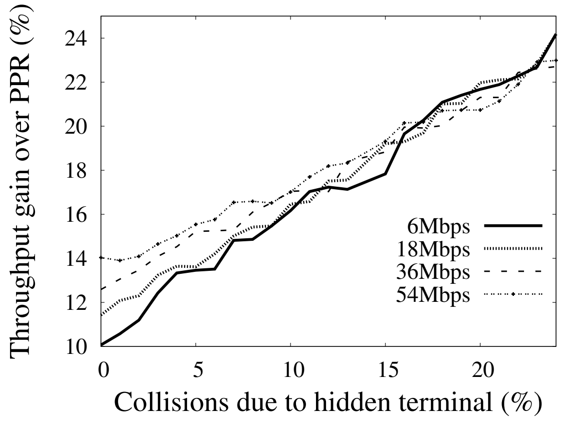
(c) The fraction of successful bytes over the total number of transmitted bytes

Figure 13: Performance comparison of CSMA/CN with 802.11-like and PPR schemes. CSMA/CN performs much better than 802.11-like scheme. The relative throughput gain with CSMA/CN over PPR ranges from 10% to 30%.

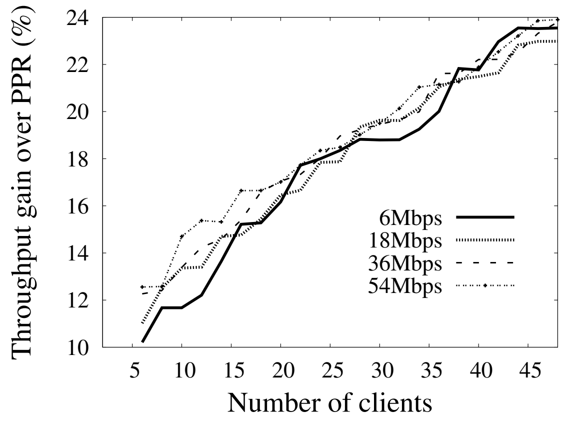
Performance Results: Figure 13 compares the throughput of CSMA/CN against PPR and 802.11. Evidently, CSMA/CN offers improved throughput than the other two schemes. Specifically, from Figure 13(a), we observe that around 80% of the links achieve more than 2 Mbps throughput with CSMA/CN. In contrast, 80% of the links obtain less than 2 Mbps throughput under PPR and 802.11. Figure 13(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 corresponds 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 improve. The relative improvement ranges from 10% to 30%.

We further analyze how PPR incurs relatively high retransmissions with respect to CSMA/CN. Figure 13(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. Further, when the interference starts first, PPR looses 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 rates, notification will have a relatively higher overhead, since it takes constant time. However, based on the example in Section 2.3, at 54 Mbps rate and 20 MHz bandwidth, this overhead amounts to less than 150 bytes. Therefore, when packets are of size 1500 bytes, aborting colliding transmissions is beneficial even at high rates. Moreover there is 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 custom simulation. The simulator does not model the detailed properties of the wireless channel and simulates only hidden terminal collisions. We believe this is reasonable because our goal is to understand CSMA/CN's performance due to collisions induced by backoff and hidden terminals. Fig 14(a) shows the performance improvement over PPR for increasing collisions due to hidden terminals. Noticeably, throughput gain is similar at various rates. In denser networks, backoff induced collisions (when multiple nodes choose the same backoff) will be higher [15]. Thus, with increasing number of nodes throughput gain over PPR increases (Figure 14(b)).



(a) Varying fraction of collisions (10 node topology)



(b) Varying number of nodes(collision probability 0.1)

Figure 14: Throughput gain with CSMA/CN over PPR.

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 quite significant. Therefore, we believe CSMA/CN is beneficial even in 802.11n networks.

5. 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 MIMO?** This design and implementation of this paper assumes single input single output (SISO) communication. Nevertheless, we hypothesize that with a better 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 [16])?** Supporting $O(n)$ frequency tones will require additional channel resources. Excessively narrow-band tones are prone to fading; tones also need to be separated by a guard band to cope with non-ideal 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 [17] 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 towards adopting collision detection in wireless networks, and certainly amenable to various improvements. Yet, even this first step provides several potential secondary benefits. Past research shows transmission bitrate should not be reduced due to collisions and should only cater to fading [18, 1, 19]. 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.

6. RELATED WORK

Avoiding Collisions: There have been numerous MAC protocols proposed for wireless networks [20]. 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 reduces the overall throughput [21] and hence disabled by default in many deployments [22]. Recently, several schemes use the knowledge of interference map to schedule transmissions intelligently [23, 17, 24]. Interference relationships vary with time and hence are difficult to monitor.

Recovering from Collisions: Apart from PPR mentioned earlier, ZipTx [3] and Maranello [25] makes 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 [5] 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 [26] requires the knowledge of one of the packets involved in a collision.

Detecting Collisions: [18] 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 [27] 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 [28]. 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 feed back to the transmitter to abort. Author's previous work [29] 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 which are more vulnerable to collisions.

7. CONCLUSIONS

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 tolerable, and the performance improvements, 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.

8. ACKNOWLEDGEMENT

We sincerely thank our shepherd Kun Tan, as well as the anonymous reviewers, for their valuable feedback on this paper. We are also grateful to NSF for partially funding this research through the following grants – CNS-0448272, CNS-0917020, CNS-0916995, and CNS-0747206.

9. REFERENCES

[1] M. Vutukuru, Hari Balakrishnan, and K. Jamieson, "Cross-Layer Wireless Bit Rate Adaptation," in *ACM SIGCOMM*, 2009.

[2] K. Jamieson and H. Balakrishnan, "PPR: Partial Packet Recovery for Wireless Networks," in *ACM SIGCOMM*, August 2007.

[3] K. Lin, N. Kushman, and D. Katabi, "ZipTx: Harnessing Partial Packets in 802.11 Networks," in *Proc. ACM Mobicom*, 2008.

[4] S. Gollakota and D. Katabi, "Zig-Zag Decoding: Combating Hidden Terminals in Wireless Networks," in *Proc. ACM Sigcomm*, 2008.

[5] D. Halperin, T. Anderson, and D. Wetherall, "Taking the sting out of carrier sense: Interference cancellation for wireless lans," in *ACM MOBICOM*, 2008.

[6] 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 *ACM MOBICOM*, 2009.

[7] S. Kakumanu and R. Sivakumar, "Glia: a practical solution for effective high datarate wifi-arrays," in *ACM MOBICOM*, 2009.

[8] "Quellan Noise Cancelers," *http://www:quellan:com/products/qhx220_ic.php*.

[9] 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 *WiMesh*, 2010.

[10] E. Gebara, E.M. Tentzeris, and J. Laskar, "Analysis and design of an interference canceller for collocated radios," *IEEE Trans. on Microwave Theory and Techniques*, 2005.

[11] J. Choi, M. Jain, K. Srinivasan, P. Levis, and S. Katti, "Achieving Single Channel, Full Duplex Wireless Communication," in *ACM MOBICOM*, 2010.

[12] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *ACM MOBICOM*, 2008.

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

[14] G. Nychis, T. Hottelier, Z. Yang, S. Seshan, and P. Steenkiste, "Enabling MAC Protocol Implementations on Software-defined Radios," in *USENIX NSDI*, 2009.

[15] K. Tan, J. Fang, Y. Zhang, S. Chen, L. Shi, J. Zhang, and Y. Zhang, "Fine-grained Channel Access in Wireless LAN," in *ACM SIGCOMM*, 2010.

[16] J. Deng and Z. Haas, "Dual busy tone multiple access (DBTMA): A new medium access control for packet radio networks," in *IEEE ICUPC*, 1998.

[17] M. Vutukuru, K. Jamieson, and Hari Balakrishnan, "Harnessing Exposed Terminals in Wireless Networks," in *5th USENIX NSDI*, 2008.

[18] S. Rayanchu, A. Mishra, D. Agrawal, S. Saha, and S. Banerjee, "Diagnosing Wireless Packet Losses in 802.11: Separating Collision from Weak Signal," in *IEEE INFOCOM*, 2008.

[19] 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 *IEEE SECON*, 2008.

[20] S. Kumar, V.S. Raghavan, and J. Deng, "Medium Access Control protocols for ad hoc wireless networks: A survey," *Elsevier Ad Hoc Networks*, vol. 4(3), May 2006.

[21] Atheros, "802.11 WLAN Performance," .

[22] Broadcom, "Wireless LAN Adapter User Guide.," .

[23] J. Manweiler, N. Santhapuri, S. Sen, R. Roy Choudhury, S. Nelakuditi, and K. Munagala, "Order matters: transmission reordering in wireless networks," in *ACM MOBICOM*, 2009.

[24] 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 *ACM MOBICOM*, 2009.

[25] 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," 2009.

[26] S. Katti, S. Gollakota, and D. Katabi, "Embracing Wireless Interference:Analog Network Coding," in *Proc. ACM Sigcomm*, 2007.

[27] S. Sen, N. Santhapuri, R.R. Choudhury, and S. Nelakuditi, "AccuRate: Constellation Based Rate Estimation in Wireless Networks," in *USENIX NSDI*, 2010.

[28] J. Peng, L. Cheng, and B. Sikdar, "A Wireless MAC Protocol with Collision Detection," *IEEE Transactions on Mobile Computing*, vol. 6(12), Dec 2007.

[29] S. Sen, N. Santhapuri, R.R. Choudhury, and S. Nelakuditi, "Moving Away from Collision Avoidance: Aborting Collisions in Wireless Networks," in *ACM HOTNETS*, 2009.