No Time to Countdown: Migrating Backoff to the Frequency Domain

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

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

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

ABSTRACT

Conventional WiFi networks perform channel contention in time domain. This is known to be wasteful because the channel is forced to remain idle while all contending nodes are backing off for multiple time slots. This paper proposes to break away from convention and recreate the backing off operation in the frequency domain. Our basic idea leverages the observation that OFDM subcarriers can be treated as integer numbers. Thus, instead of picking a random backoff duration in time, a contending node can signal on a randomly chosen subcarrier. By employing a second antenna to listen to all the subcarriers, each node can determine whether its chosen integer (or subcarrier) is the smallest among all others. In fact, each node can even determine the rank of its chosen subcarrier, enabling the feasibility of scheduled transmissions after every round of contention. We develop these ideas into a `Back2F` protocol that migrates WiFi backoff to the frequency domain. Experiments on a prototype of 10 USRPs confirm feasibility, along with consistent throughput gains over 802.11. Trace based simulations affirm scalability to larger, real-world network topologies.

Categories and Subject Descriptors

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

General Terms

Design, Experimentation, Performance

Keywords

Wireless, Contention Resolution, Cross-Layer, Backoff

1. INTRODUCTION

Access control strategies are designed to arbitrate how multiple entities access a shared resource. Several distributed protocols embrace randomization to achieve arbitration. In WiFi networks, for example, each participating node picks a random number from a specified range and begins counting down. The node that finishes first, say N_1, wins channel contention and begins transmission. The other nodes *freeze* their countdown temporarily, and revive it only after N_1's transmission is complete. Since every node counts down at the same pace, this scheme produces an implicit ordering among nodes. Put differently, the node that picks the smallest random number transmits first, the one that picks the second-smallest number transmits second, and so on. The overall operation is often termed as "backoff".

While backoff arbitrates channel contention, it incurs a performance cost. Specifically, when multiple nodes are simultaneously backing off, the channel must remain idle, naturally leading to under-utilization. Moreover, network congestion prompts an exponential increase in the backoff range, introducing the possibility of greater channel wastage. Authors in [16] show more than 30% reduction in throughput due to backing off; [13] shows the severity at higher data rates. This paper attempts to address this problem by migrating the backoff operation to the frequency domain.

Our main idea is simple. When a node N_1 has a packet to transmit, it picks a random value, r_1, from a specified range $[0, F]$. Once the channel becomes idle, N_1 begins the backoff operation. However, instead of counting down from r_1 to 0, N_1 transmits a symbol on the r_1^{th} subcarrier[1]. We assume that each node has two antennas; thus, while one antenna transmits, the other antenna listens to determine which of the subcarriers are active. Assuming N_2 is also contending for the channel, and say has transmitted on the r_2^{th} subcarrier, N_1 observes activity on both subcarriers r_1 and r_2. Assuming $r_1 < r_2$, N_1 immediately infers that it has won channel contention, and begins transmission. N_2 learns that it has lost, and defers its own transmission until N_1 has finished. We call this approach `Back2F`, as an acronym for *migrating backoff to the frequency domain.*

The advantages of `Back2F` are two fold. First, one round of frequency domain backoff should ideally last for few OFDM symbols, substantially less than the average backoff in proto-

[1] Subcarriers are narrowband OFDM channels used by 802.11.

cols like 802.11. Second, Back2F creates a logical ordering among contending nodes, and each node learns its own rank in this order. This ranking among nodes creates the possibility of batched transmissions, eliminating the need for per-packet backoff. Since 802.11 currently backs off before every packet, Back2F helps improve the channel usage and network throughput.

Of course, extracting these gains entail a number of research challenges: (1) Active subcarriers need to be detected accurately in face of loose time synchronization among transmitters, energy leakage between narrow-band subcarriers, and channel fading. (2) Collision among nodes – which happens when multiple nodes choose the same subcarrier – needs to be mitigated successfully. (3) Finally, since nodes are located in different contention neighborhoods, the node rankings do not obey any global order. Back2F needs to cope with relative ranking among nodes, while maintaining spatial reuse and fairness comparable to 802.11. We address these challenges through activation of diverse subcarriers, multiple rounds of contention, and virtual countdown. We consolidate these ideas into a protocol, and implement a prototype on the USRP/ GNURadio platform. Experimental results show 95% accuracy in subcarrier detection, less than 2% probability of collision, and throughput gains of more than 35% at high bitrates. Trace driven simulations (with topologies and channel conditions drawn from our university network) confirms stability and scalability of Back2F to real-world scenarios.

Our contributions in this paper may be summarized as:

- *We identify an opportunity to migrate protocol operations from the time to the frequency domain.* Although we instantiate our ideas through a WiFi based MAC, they may be generalized to other arbitration strategies.
- *We design an OFDM based system where random backoff is realized by selectively transmitting on a subcarrier.* A logical order among senders is enforced in a decentralized manner, for improved channel usage.
- *We address the challenges behind such a scheme, and prototype it on the USRP/GNURadio platform*. Promising results, in terms of throughput, fairness, and scalability, give us confidence to build a larger system.

2. 802.11 AND OFDM

This section highlights the limitations of 802.11's backoff (in time domain), and presents a simple abstraction of OFDM (to better explain the shift to frequency domain).

802.11 Channel Access: WiFi prescribes each transmitter to backoff for a random number of slots, chosen from the range $[0, CW-1]$, where CW is the contention window. Each time slot corresponds to 9μs. The node counts down only if the channel is idle – if the node senses a busy channel, the countdown is frozen, and revived only after the channel is idle. Whichever node completes the countdown first begins transmission. When this transmission is complete, the other nodes wait for a DIFS duration, and continue with their remaining countdown. *Observe that 802.11 implicitly forms a queue among contending nodes, each node's position in the queue determined by the random number it chooses.* Prior analyses have shown that this scheme guarantees stability and fairness [6].

We make three observations that are not necessarily new. (1) Fundamentally, backing off is not a time domain operation. Its implementation is in the time domain, forcing the channel to be idle before each packet transmission. (2) The duration of each backoff slot is fixed, so the channel wastage grows with smaller packets and higher bitrates. A packet's airtime is shorter at higher rates, and hence, the fraction of channel-time occupied by idle slots is larger. (3) Finally, although channel utilization may improve with few nodes (backing off in parallel), just a few more nodes can cause collisions (802.11 experiences 18% collisions with 3 nodes and $CW{=}16$). A collision forces nodes to exponentially increase their backoff, pushing the system back to under-utilization. Fig. 1 shows the channel under-utilization due to 802.11's backoff, under varying bitrates and network densities. Authors in [16, 29] corroborate these findings with extensive analysis and measurements, emphasizing the need to improve wireless contention resolution.

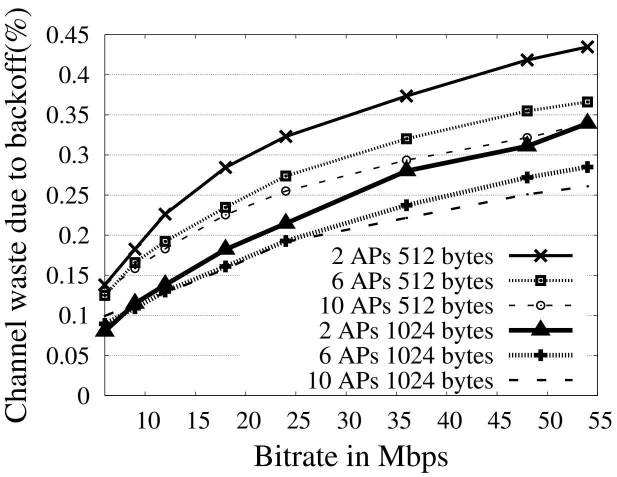

Figure 1: Overhead of 802.11 backoff. Larger fraction of channel wasted with smaller packets at higher bitrates.

Orthogonal Frequency Division Multiplexing: OFDM can be abstracted as a PHY scheme that divides the wireless spectrum into multiple narrow band channels, called subcarriers. The subcarriers carry modulated data streams in parallel, but at a lower rate per-subcarrier. The benefit of OFDM emerges from its ability to cope with channel adversities, including narrowband interference and frequency-selective fading due to multipath. The 802.11a/g implementation of OFDM has 52 subcarriers, of which 48 are used for data transmission, and 4 for equalization. A transmitter stripes bits across all subcarriers, however, it is possible to transmit/receive only on a subset of them.

As we will see later, a Back2F node picks a random number, say 11, and transmits a short signal *only* on the 11^{th} subcarrier. The node's second antenna detects a strong signal on the 11^{th} subcarrier, as well as on other subcarriers used by other contending nodes. Practical hardware constraints raise difficulties in discriminating between adjacent subcarriers. When the second antenna receives a strong signal on the 11^{th} subcarrier, "leakage" into adjacent subcarriers may mislead the receiver into detecting subcarriers 10 and 12 also as active. Higher point FFTs are useful to mitigate such effects – the spikes on subcarriers can be better isolated. Migrating to the frequency domain brings these problems, and Back2F needs to handle them.

3. ARCHITECTURE AND DESIGN

We present Back2F in 3 sub-parts. First, we describe the scheme under the assumption of a single collision domain (i.e., all nodes can hear each other). Second, we relax the assumption, and describe how the approach can be extended to networks with multiple collision domains. Finally, we explore optimizations to support batched transmissions, obviating the need to perform per-packet contention. The section concludes with discussion about practical challenges in realizing Back2F and techniques to overcome them.

3.1 Backoff within a Single Collision Domain

Figure 2 shows an example where contenders AP1 and AP2 choose random numbers 11 and 29, respectively. However, instead of counting down these numbers in time, they transmit a short signal on their corresponding subcarriers. While the signal is being transmitted on the transmit antenna, a listening antenna on each AP receives the combined signals from all the APs, as well as its own signal, called the *self-signal*. The listening antenna then extracts all the active subcarriers, thereby learning the backoff values of the other contenders. With knowledge of everyone's backoffs, each AP can instantaneously determine whether it has won the contention. If its own backoff is smaller than *all* others, it proceeds with data transmission; otherwise, it defers to a later time. Of course, data transmission is performed using all the subcarriers, identical to regular 802.11 operation.

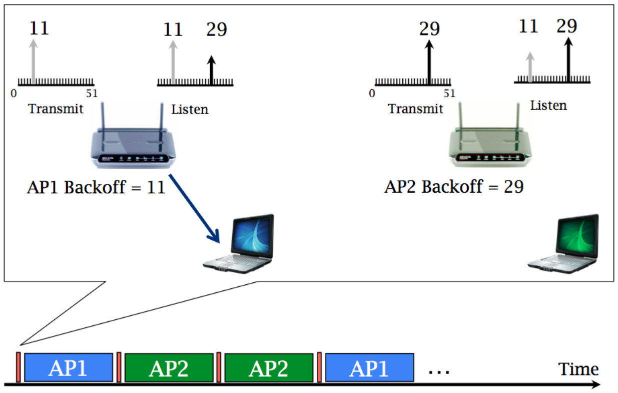

Figure 2: A close up view of the first backoff. AP1 picks/activates subcarrier 11 and AP2 chooses 29. They learn of other backoff values through subcarriers. AP1 with smaller backoff transmits whereas AP2 defers.

The signaling on subcarriers happen synchronously. Synchronization is achieved implicitly [28], i.e., both APs observe the same channel, and hence, when the channel becomes idle, both APs recognize it as a trigger to begin transmission. Of course, the synchronization may not be perfect due to differences in signal propagation delays. Subsection 3.5 discusses ways to cope with the problem.

In the above example, both AP1 and AP2 learn that the backoff values of all the contending nodes are 11 and 29. AP1 with smaller backoff of 11 proceeds to transmit, whereas AP2 with larger backoff defers. The deferred node deducts the smallest known backoff and contends again after the channel has become idle, i.e., after AP1 finishes, AP2 contends with a backoff value of $29 - 11 = 18$. Observe that the net effect is exactly like time-domain backoff in 802.11. All the contending nodes count down simultaneously till the smallest of them reaches zero; the node whose backoff reaches zero proceeds to transmit, while others contend later with their reduced backoff values. However, unlike 802.11, the time to pick a winner in Back2F is much shorter, in the order of few OFDM symbols. Algorithm 1 summarizes the essence of Back2F through a pseudocode.

Algorithm 1 : Basic operation of Back2F(pkt)

```
1: myback ← random(0, maxback)
2: wait for the channel to be idle for DIFS duration
3: transmit on subcarrier myback
4: in parallel, listen for active subcarriers (allbacks)
5: myback ← myback − min{allbacks}
6: if myback ≠ 0 then goto line 2
7: transmit pkt
```

What if two contending nodes choose the same backoff value – how does Back2F cope with collisions?

Collisions are certainly possible when two nodes pick the same random subcarrier. Back2F copes with collisions by introducing a second round of subcarrier based contention. A node that believes is a winner in the first round, retransmits on another randomly chosen subcarrier immediately after. Figure 3 illustrates the process.

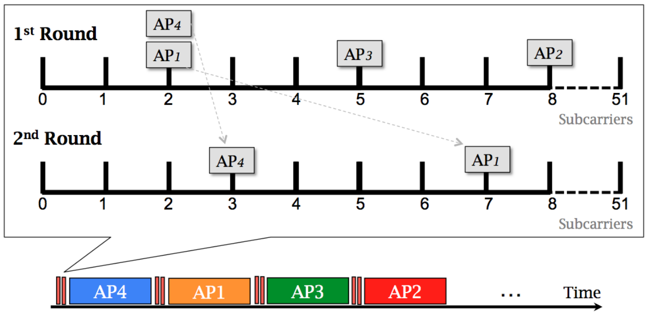

Figure 3: Illustration of Back2F with two contention rounds. AP1 and AP4 choose the same smallest backoff, and enter the second round of contention. AP4 wins the second round and accesses the channel.

AP1 and AP4 choose the same backoff value that happens to be smallest among all the other backoffs. Then, both APs advance into a second round of contention, and this time AP4 picks 3 while AP1 picks 7. AP4 being the winner of the second round proceeds to transmit while AP1 waits to participate in the next backoff. If multiple nodes chose the minimum number in the first round, the probability of them coinciding again is small. Back2F can reduce the collision probability to an arbitrarily small value, at the expense of more rounds. The evaluation section demonstrates that two rounds of contention suffice for up to 50 contenders.

Once the winning node completes transmission, all the losing nodes (AP1, AP2, and AP3) contend for the next opportunity to access the channel. However, instead of choosing a new backoff, they revise their prior backoff (as mentioned earlier). Figure 4 illustrates this as a follow up to Figure 3. The smallest backoff of 2 from the first round is deducted from each node's backoff. Thus, the resulting backoffs of AP1, AP2, and AP3 are 0, 3, and 6, respectively, which they now use for contention[2].

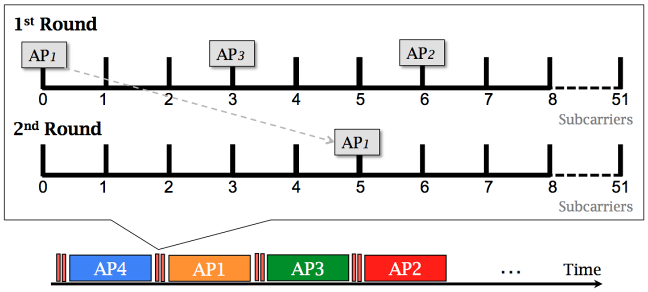

Figure 4: AP1, AP2, and AP3 lost the contention in Figure 3, hence, reduce their backoffs by 2 (i.e., AP4's first round backoff value). This reduction emulates (virtual) elapsing of 2 time slots due to AP4's countdown. After AP4 finishes, AP1, AP2, and AP3 contend with these revised backoff values. AP1 wins this time and transmits.

Ideally, Back2F should recognize that AP1 is the sole winner in Figure 4, and obviate the need for a second round of transmission. Unfortunately, there is no way to reliably tell between a sole winner and collisions. This is because a Back2F node can detect which subcarriers are active, but cannot tell how many nodes transmitted on a given subcarrier. Hence, we propose to always perform two rounds of contention, though it is suboptimal. As a result, only AP1 advances to the second round in Figure 4, obviously wins the contention, and proceeds to transmit its packet.

Algorithm 2 presents the pseudo code for two-round Back2F. Lines $1-6$ correspond to the first round and $7-10$ reflect the second round. If a node loses in the second round, it behaves just like it would after losing in the first round. It goes back to line 2 to contend later with a revised backoff as in line 5. Again, note that the second round is only meant to break ties, while the transmission order is solely based on the backoff values chosen in the first round.

3.2 Backoff over Multiple Collision Domains

For the ease of introduction, the above description of Back2F makes a simplifying assumption that all the contending nodes belong to a single collision domain. Since these nodes carrier sense each other, they become aware of all backoff values, resulting in a consistent view of the global ranking among nodes. In practice, however, a wireless network will obviously span over multiple collision domains as in Fig. 5. In this toy example, AP1 and AP2 belong to one collision domain, while AP2, AP3 and AP4 belong to a different one (i.e., AP1 does not carrier sense AP3 or AP4, and the vice versa). When a node such as AP2 belongs to two collision domains, it may be a winner in one but not in the other. Back2F copes with these cases as described next. Suppose the backoff values of AP1, AP2, AP3, and AP4 are 9, 7, 6, and 15, respectively. Then, according to AP1, node AP2 is the winner, whereas in AP2's view, the winner is AP3 (note that an AP actually does not know who the winner is; it only knows whether it is the winner or not). The consequence is that only AP3 proceeds to transmit, AP2 defers to AP3, and AP1 defers to AP2. This is unnecessary because AP1 could very well transmit in parallel to AP3. Back2F addresses this form of *head-of-line blocking* to uphold spatial reuse in the network. When AP1 observes that the channel is idle for DIFS duration, it infers that the winner is blocked by some other transmission. Hence, AP1 initiates a backoff with its revised value of $9 - 7 = 2$. Assuming AP1 is the only contender, it wins the channel and begins transmitting. Now, even though AP3 completes transmission, AP2 still does not transmit because it carrier senses AP1. AP4 now observes an idle channel, readjusts its backoff to $15 - 6 = 9$, and advances into communication. Once AP1 and AP4 are done, AP2 transmits its packet. Observe that the overall order of transmissions mimics 802.11; only the backoff procedures are quicker. This scheme also generalizes to larger topologies.

Algorithm 2 : Back2F with two rounds(pkt)

```
 1: myback ← random(0, maxback)
 2: wait for the channel to be idle for DIFS duration
 3: transmit on subcarrier myback
 4: in parallel, listen for active subcarriers (allbacks)
 5: myback ← myback − min{allbacks}
 6: if myback > 0 then goto line 2
 7: myback2 ← random(0, maxback)
 8: transmit on subcarrier myback2
 9: in parallel, listen for active subcarriers (allbacks2)
10: if myback2 ≠ min{allbacks2} then goto line 2
11: transmit pkt
```

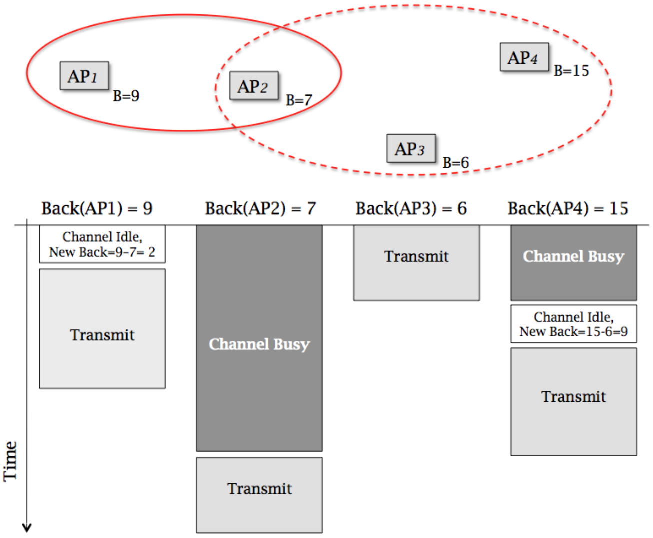

Figure 5: Back2F **with multiple collision domains: Due to differing views, AP1 thinks AP2 won the contention whereas AP2 lost to AP3. However, when the channel is idle for DIFS, AP1 performs backoff and transmits.**

[2] AP1 ignores the number 7 it chose in the second round of Figure 3. It is only used to break ties among first round winners.

3.3 Coping with Misdetection due to Fading

A natural question is *how does subcarrier misdetection (false positives and negatives) affect the performance of* `Back2F`*?* Consider a false positive where a node falsely detects a subcarrier f even though no one has transmitted on that subcarrier (while it is unlikely to find energy on a subcarrier in the absence of an actual transmission, we still consider this case for completeness). Observe that such a false positive does not affect `Back2F`, so long as f is not the smallest-valued subcarrier – the winner – among the contenders. When f is indeed the smallest, a false positive can mislead the (actual) winner to unnecessarily wait for a non-existent transmission. The wait, however, is bounded by DIFS – once the actual winner observes an idle channel for DIFS, it initiates backoff and subsequent operations. Thus, a false positive may, in some cases, cause the channel to be wasted for a DIFS duration.

Now consider the more likely case where a node with self-subcarrier i fails to detect a legitimate subcarrier f, i.e., a false negative. Again, a false negative has no effect when number i is less than f – in this case, i will still correctly detect that it has won the contention. Even if i is greater than f, the impact may not be severe if this occurs in the first round. This is because i may believe it is the winner of the first round, and advance to the second to contend again. However, if f is smaller than i in the second round, then the node with self-subcarrier i will wrongly assume itself to be the winner, causing a certain collision. This is a serious consequence of a false negative since it defeats the purpose of backing off.

We believe (and will evaluate later) that only under some restricted conditions discussed above, will false negatives affect performance. Nevertheless, we propose the following to alleviate the impact of false negatives. The main observation is that false negatives are of concern *only in the second round of contention*, and that the second round typically has a few nodes contending. This suggests that fewer (than 52) subcarriers may be adequate to select the winner among them. Thus, instead of transmitting on a single subcarrier, a node can transmit on multiple subcarriers on different parts of the frequency spectrum. For instance, a node that has picked a random number i, can transmit on subcarriers i and $((\frac{F}{2}+i)$ mod $F)$, where F is the total number of subcarriers. Given that these two subcarriers would be separated in frequency, they may observe dissimilar fading patterns, making at least one of them detectable (Figure 6). Further, subcarrier detection thresholds may also be tuned to lower false negatives at the expense of less-expensive false positives. We evaluate the overall impact of subcarrier misdetection in Sec. 4, and demonstrate that the performance is not overly sensitive.

3.4 Optimization: Batched Transmissions

Unlike 802.11, `Back2F` enables each node to learn its rank *a priori* in the sequence of pending transmissions. In fact, each node also knows the exact backoff values chosen by other nodes (*although the mapping between node ID and backoff value is not known*). `Back2F` aims to exploit this knowledge to batch transmissions (i.e., a train of back-to-back packets between successive backoffs). A node ranked n can transmit immediately after the $(n-1)^{th}$ ranked node finishes transmission. The protocol structure is as follows.

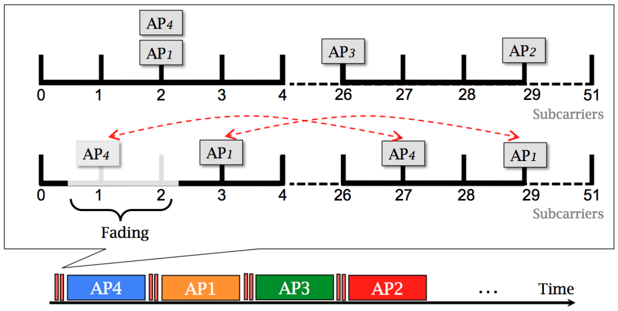

Figure 6: Signaling on subcarriers i and $26+i$ in the second round to alleviate the impact of false negatives.

Batch size and transmission order.
As a first step, `Back2F` promotes the *top-K* ranked APs to the second round, as opposed to the winner(s) alone. The choice of K brings out a tradeoff between collision probability and batch size. Higher values of K will result in longer batch size (better throughput), but at the risk of collisions. We set $K = 3$ in our implementation, empirically observed to provide the best tradeoff. Given the choice of K, each node knows whether it is *top-K* ranked after the first round of contention. Those that are, proceed to the second round, while others do not. As an example, suppose 4 APs in Figure 7 contend in the first round, and the protocol intends to promote the *top-2* ranked APs to the second round. AP1 and AP4, both ranked 1, naturally advance; AP3 also advances because it is ranked 2. These three nodes are then ordered as AP4, AP3, AP1 in their second round of contention, and they transmit back-to-back in that sequence, without a per-packet backoff.

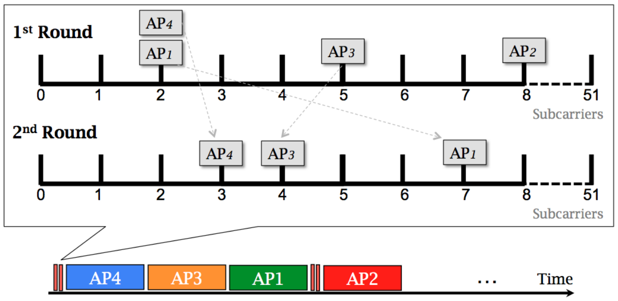

Figure 7: Backoff in the frequency domain followed by scheduled transmissions. All APs contend in the first round, but only AP1 and AP2 enter the second round. Based on backoff values in the second round, the schedule is AP1 followed by AP2. Only after the scheduled transmissions complete, AP3 and AP4 contend again.

Of course, `Back2F` cannot guarantee that exactly K nodes will proceed to the second round – "*top-K* ranked nodes" may imply more than K nodes. However, by observing the active subcarriers in the first round, each node can *probabilistically* estimate how many nodes have picked the same subcarrier. The promotion to the second round may be guided by this estimation. Thus, if the protocol desires 5 nodes in the second round, and probabilistic estimates suggest 1.25 nodes on average per subcarrier, nodes ranked 4 or less can proceed to the second round. Each contending node can make this decision independently.

The next question is: *with batched transmissions, how does a node know when it is its turn to transmit?* The general idea is to require node i to wait for $R_i - 1$ packet transmissions, where R_i is the rank of the i^{th} node. To be specific, Back2F separates the batched transmissions with a PIFS duration, where PIFS is less than DIFS. This ensures that other nodes cannot interject their transmissions while a batch is in progress[3]. Thus, a node ranked R waits for $(R-1)$ instances of "*channel busy followed by PIFS*" after the backoff operation. After that, the node initiates its own transmission.

Adjusting backoff values.
Given that Back2F intends to preserve 802.11's transmission ordering, the question is how are the backoff values of the loser nodes adjusted with batched transmissions. For this, we can pretend that all the nodes that enter the second round have already counted down to zero. In that case, the highest backoff value promoted to the second round needs to be deducted from the backoff values of all loser nodes. In the example in Figure 7, since AP3 enters the second round with backoff$=5$, we should pretend that 5 time slots have elapsed, and deduct it from AP2's backoff. Thus, when AP2 contends later, it uses a backoff of 3, and transmits on the third subcarrier. Observe that at every successive contention phase, a loser node will advance in its rank, and is eventually guaranteed to access the channel. This is well in alignment with 802.11's ability to avoid starvation. Of course, with batched transmissions, the precise ordering of transmissions with Back2F may deviate from 802.11. For instance, with Back2F, the ordering is AP4, AP3, AP1, AP2, whereas the 802.11 ordering would be AP1/AP4, AP3, AP2. Nevertheless, the differences are localized to one batch, and do not carry over to subsequent batches. Therefore, even with batched transmissions, Back2F roughly emulates 802.11 and provides similar fairness; throughput improves consistently.

Algorithm 3 presents the pseudo code for Back2F with batched transmissions. Based on the first round backoff values, a node determines whether its backoff is less than or equal to the K-th minimum (lines $5-6$). If not, it goes back to line 2 to contend later. Each node that enters the second round, picks another subcarrier number, learns of all active subcarriers, and determines its own rank (lines $7-10$). It then keeps decrementing its rank whenever it observes instances of "*channel busy followed by PIFS*". When its rank becomes 1, it proceeds to transmit the packet.

Batched transmissions over multiple collision domains.
Batching is applicable to multiple collision domains as well. Suppose the backoff values listed in Figure 5 correspond to the second round of contention. Then, AP1, AP2, AP3, and AP4 determine their ranks to be 2, 2, 1, and 3, respectively. Note that, from AP1's perspective, its own rank is 2 (after AP2), while AP2 also computes its own rank as 2 (after AP3). Consequently, AP3 proceeds to transmit whereas AP1 monitors the channel for 1 instance of "*channel busy followed by PIFS*". Now, AP1 finds the channel to be idle for DIFS – this is unexpected as it anticipates a higher ranked node to transmit on the channel. At this point, AP1 breaks away from the batch order, and initiates another contention. Note that AP2 does not contend this time because it is blocked by AP3's transmission. Thus, Back2F incurs a penalty of DIFS whenever the batch order is not followed, reducing the gains from batched transmissions. Of course, the reduction is not severe, as indicated by results in Section 4.

Algorithm 3 : Back2F with batched transmissions(pkt)

```
 1: myback ← random(0, maxback)
 2: wait for the channel to be idle for DIFS duration
 3: transmit on subcarrier myback
 4: in parallel, listen for active subcarriers (allbacks)
 5: myback ← max(0, myback − $K^{th}$min{allbacks})
 6: if myback $> 0$ then goto line $2$
 7: myback2 ← random(0, maxback)
 8: transmit on subcarrier myback2
 9: in parallel, listen for active subcarriers (allbacks2)
10: myrank ← rank(myback2, allbacks2)
11: while myrank $> 1$ do
12:     wait for "channel-busy-followed-by-PIFS"
13:     myrank ← myrank $-1$
14: transmit pkt
```

3.5 Points of Discussion

(1) The self-signal from the transmitting antenna to its own listening antenna is strong – how does this affect the detection of other subcarriers at the listening antenna?

Figure 8 shows the feasibility of discerning multiple adjacent subcarriers. The subcarrier with the highest spike is the self-subcarrier, while contending nodes were made to transmit on adjacent and nearby subcarriers. Figure 8(a) shows the (64 pt. FFT) spectrum when 10 consecutive subcarriers (out of 52) are active in a bandwidth of $8MHz$. Since transmissions leak into adjacent subcarriers, the ability to discriminate is weak. However, a higher point FFT reduces this leakage. Figure 8(b) and 8(c) show the frequency spectrum when 128 and 256 pt. FFTs are used. Even in presence of a high self-subcarrier, transmissions on adjacent and nearby subcarriers are reliably discerned.

Figure 9 zooms into the effect and shows the leakage into adjacent subcarriers, with varying FFT sizes. Observe that at 256 pt. FFT, the subcarrier adjacent to the self-subcarrier experiences less than $10dB$ leakage, and substantially lower at subcarrier separations of 2 or more. Since this leakage can be characterized for a given hardware, the adjacent subcarrier detection can be tackled with proper thresholding. In other words, the adjacent subcarrier would be considered active only if the SNR on this subcarrier is greater than $10 + \tau$, where τ is an estimate derived from sensing the channel in the preceding DIFS duration.

(2) Transmissions on subcarriers are not tightly time synchronized – how does this impact Back2F?

The lack of synchronization in initiating subcarrier transmission emerges from: (1) two nodes may not observe the channel to become idle at the exact same time due to difference in propagation delays (t_{pd}), and (2) once all nodes begin subcarrier signaling, the signals arrive at any receiver with some stagger, caused by t_{pd} again. This total stagger introduces difficulty in "catching" all the active subcarriers at the same time – all the signals need to overlap at a receiver's antenna for at

[3] Of course, this may break interoperability with PCF mode, in which case we need an xIFS duration between PIFS and DIFS.

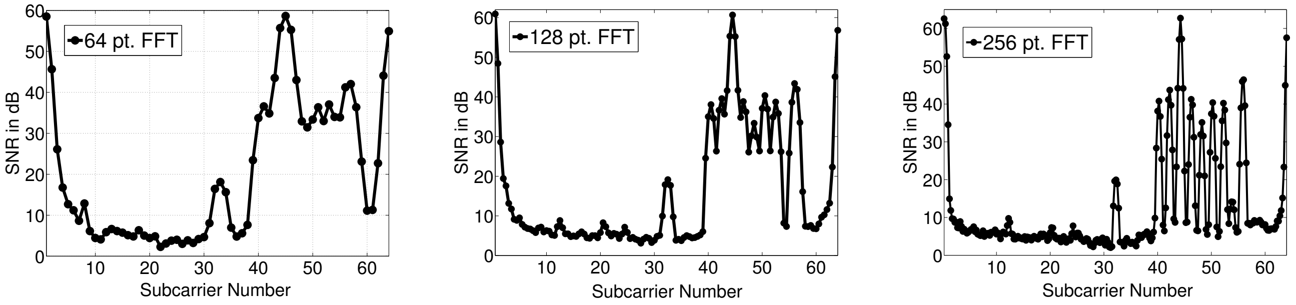

Figure 8: Active subcarrier detection when 10 nodes transmitting with 64pt IFFT using 8MHz bandwidth. Receiver using (a) 64pt FFT; (b) 128pt FFT; (c) 256 pt FFT. All active subcarriers are clearly discernible with higher pt. FFT

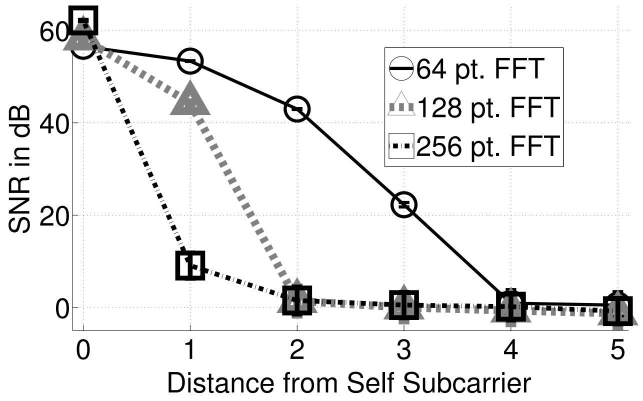

Figure 9: Effect of self signal on adjacent subcarriers. Effect is low (<10dB) for every consecutive, 2nd and 4th subcarrier for 256, 128, 64 pt. FFT respectively.

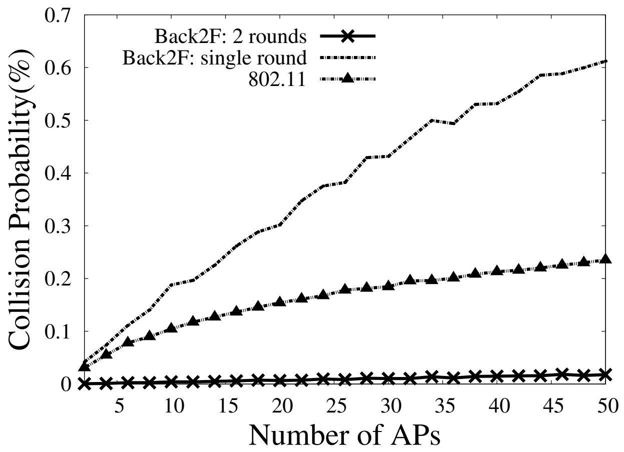

Figure 10: Collision probability of Back2F with two rounds remains below 2% in high density networks with 52 subcarriers. 802.11 experiences more collisions.

least one FFT window. Back2F copes with this problem by requiring the subcarrier signaling to occur for slightly longer duration than one OFDM symbol. This longer duration includes the maximum difference between propagation delays ($= 2t_{pd}$), time for the FFT computation, a hardware circuit delay, and guard factors. With $t_{pd} = 1\mu s$ in WLANs [10], $64pt$ FFT taking $3.2\mu s$ at 20MHz, and circuit delay of $3\mu s$, two rounds of Back2F contention incurs $16.4\mu s$. This is appreciably smaller than time domain backoff, varying uniformly between $9\mu s$ to $135\mu s$.

(3) Are 52 subcarriers and 2 rounds of contention adequate to cope with collision in high density scenarios?

Figure 10 shows the collision probability with increasing number of contenders. Observe that when using 52 subcarriers, the collision probability increases quickly with a single round of contention, and in fact, is worse than 802.11. However, promoting the winners of the first round to the second, drastically reduces the collision probability. Evidently, even when the number of contenders is more than 50 (a rare scenario in WiFi networks), the collision probability still remains less than 2%. 802.11, on the other hand, collides far more frequently, and adapts through exponentially increasing its contention window. Naturally, waiting for exponentially longer durations, incurs a performance penalty

4. IMPLEMENTATION AND EVALUATION

This section is organized to answer two main questions: (1) the reliability of subcarrier detection in an actual prototype network, and (2) Back2F's performance gain in realistic network topologies. We begin the discussion with a description of our prototype testbed.

4.1 USRP/GNURadio Prototype

We prototype Back2F on a small testbed of 10 USRPs – each transmitter formed by placing two USRPs adjacent to each other, with a separation of $\approx$ 20 inches[4]. The strong self-signal is at $60dB$. The transmitter transmits on a 8MHz band, while the listening antenna samples the same 8MHz channel, at the same center frequency. The transmitter is equipped to transmit on any of the 52 subcarriers, that are converted into a time domain signal using $64pt$ IFFT. The listening antenna can execute 64, 128, 256 point FFTs[5]. We show that the detection accuracy improves with FFT size. Note that this

[4] This distance can be reduced significantly without increasing the self-signal [9] – we have not adopted this optimization.

[5] The listening antenna may not be viewed as an additional wireless interface. It only performs FFT and energy detection per subcarrier.

requirement is specific to USRPs since the current 802.11a/g OFDM designs on USRPs need higher FFT sizes due to imprecision [26, 31]. We believe commercial hardware offers greater precision, obviating the need for higher FFT sizes at the listener.

The listening antenna detects subcarriers using a joint thresholding and peak-detection scheme. This is necessary because with practical hardware (especially USRPs), the subcarriers emerge as peaks rather than ideal impulses (Figure 8). Whenever a peak is above a threshold, Back2F declares it as an active subcarrier. Since backoff is always preceded by a DIFS interval in which the channel is idle, this threshold is adaptively chosen by sampling the noise and interference floor over this interval. This helps in keeping the false positives/negatives low.

Subcarrier Detection: The feasibility of detecting a subcarrier, in presence of a strong self-signal, is the problem of interest. In the test, transmitters were randomly placed and made to transmit signals on subcarriers at varying spectral separation from the self-subcarrier. Figure 11 shows the detection accuracy (1− FalseNegative) as a function of subcarrier distance from the self-subcarrier. As anticipated, the influence of the self-signal reduces with increasing distance. Also, with increasing FFT size at the listening antenna, even the nearby subcarriers can be detected more accurately. Using a 256 pt FFT, subcarriers above $14dB$ can be detected reliably. OFDM based carrier sense threshold in 802.11g/n permits transmission when the signal in the channel is $13dB$ or below in comparison to the noise floor [1, 8]; thus Back2F will almost be able to detect all links that are within the collision domain. Occasional false negatives may still happen, however, as we show later, Back2F is reasonably robust to such occurrences.

Impact of noise and interference: Back2F is most vulnerable to misdetection on the subcarrier adjacent to the self-subcarrier. Hence, we focus on the detection of this subcarrier to investigate the worst case performance of Back2F. Figure 12(a) shows the false negatives in detecting the adjacent subcarrier with $256pt$ FFT, for varying SNR of the signal on that subcarrier. It also shows the false positives, i.e., incorrectly detecting an inactive subcarrier. Clearly, false positives are rare, less than 2%. As the SNR of the signal on the adjacent subcarrier increases, the possibility of false negatives decreases, and at 14dB or more, it can be detected reliably.

Figure 12(b) also shows that background interference does not affect our results. Back2F does not falsely detect background interference as an active subcarrier because it adjusts its threshold (as explained above) depending on energy per-subcarrier estimated during the idle DIFS. We believe that at higher bandwidths (20MHz in 802.11), Back2F's detection accuracy may improve as a consequence of greater spectral separation between subcarriers. Due to limitations in the processing capability of USRPs, the experiments in this paper are performed at 8MHz bandwidth, but with all 52 subcarriers as in 802.11. Moreover, results presented later show that Back2F can tolerate some incidence of false negatives.

4.2 Trace based Performance Evaluation

The above USRP/GNURadio based prototype is suitable for demonstrating the feasibility of active subcarrier detection, but not the resulting gain from Back2F. Latency constraints with the USRP platform disallow realtime implementation of all the Back2F protocol operations. Therefore, we resort to trace based evaluation to assess performance in realistic scenarios.

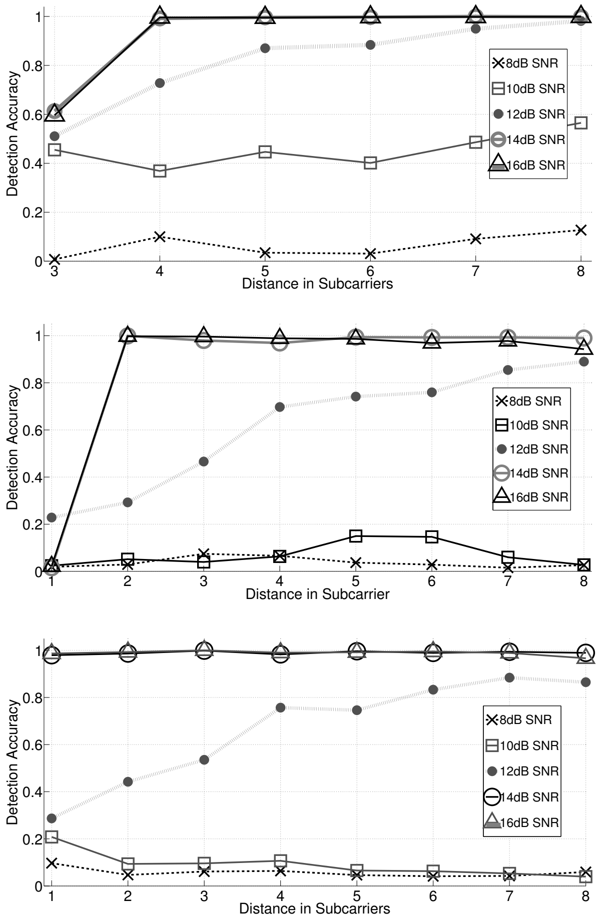

Figure 11: Detection accuracy of subcarriers at varying distance from the self subcarrier: Transmitter uses 64pt IFFT while listening antenna employs (a) 64, (b) 128, or (b) 256 pt FFT. With 256pt FFT, adjacent subcarriers with SNR 14dB or greater can be detected with 97% accuracy. Similar results for 128pt and 64pt FFT but at a higher separation. With 128pt and 64pt FFT, we can use every 2nd and 4th subcarrier respectively for Back2F.

Evaluation setting: To conduct high fidelity emulation of real world setting, we collected traces of channel characteristics and network traffic as follows. We placed APs in 20 locations and clients in 45 locations throughout our engineering building. To gather information per subcarrier, we use Intel 5300 chipset based wireless cards [2]. For each AP to client link, we recorded the RSSI, channel impulse response, transmission bitrate, and collision probability with respect to the strongest interfering AP. The transmission bitrate is experimentally selected as the highest bitrate that can support a delivery ratio of 90% or more. To estimate collision probability, we turn off carrier sensing at the APs and activate downlink transmissions in pairs. The collision probability is experimentally calculated with 10 runs of 500 packets of size 1500 bytes. We also gath-

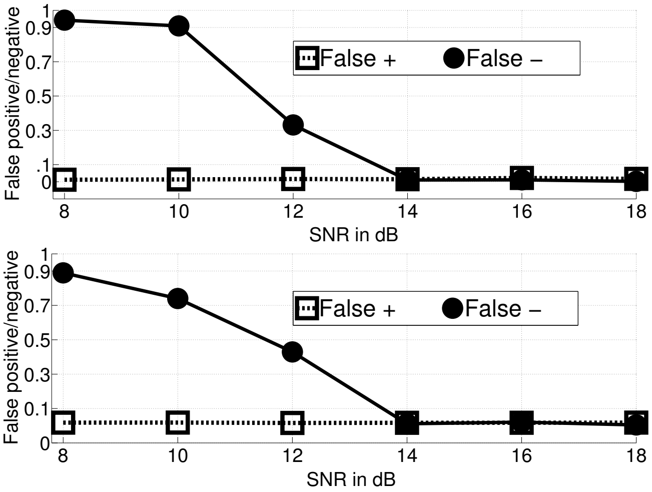

Figure 12: False positives/negatives in subcarrier detection with (a) 256pt FFT at receiver; (b) 256pt FFT in presence of 10dB interference. Low false positives in general and false negatives are low too at 14dB or more. Also, subcarrier detection is tolerant to interference.

ered RSSI and channel impulse response between every pair of APs. The Intel 5300 cards combine impulse responses to report the channel matrix for only 30 subcarriers. We estimate the channel matrix for all the 52 subcarriers via interpolation. As a representative of traffic mix in the real world, we collected traces from Skype (real time traffic), web browsing and HD streaming sessions. The average packet size in these cases were 511, 1063, and 1424 bytes respectively.

Based on the collected traces, we emulated topologies of various sizes. To model a topology with k transmitters, we uniformly choose k APs to cover the building. The remaining $65-k$ nodes are treated as clients and each of them is associated with the nearest AP (with the strongest RSSI) creating a wireless LAN like setting. We pick 100 instances of each topology size ranging from 6 APs to 18 APs. We use the real world traffic traces collected above to emulate download traffic from APs to clients. When an AP has a packet to transmit, we use the traces to determine which other APs can carrier sense this transmission, the collision probability with another hidden AP, its ability to detect active subcarriers, etc. These attempts are targeted to mimic real-world scenarios.

The relative performance of Back2F over 802.11 depends on the transmission bitrates — higher the bitrate, better the relative gain with Back2F. Hence, it is important to report the nature of links in the emulated topologies in our evaluation. Figure 13 shows the CDF of bitrates of links in each topological setting with varying number of APs. It indicates that our topologies included links with several different bitrates, not just the highest bitrate to favor Back2F.

Throughput gain: We compare Back2F's overall throughput against 802.11, under varying topologies and Internet traffic patterns. Figure 14(a) and (b) present the relative throughput gain due to Back2F without and with batched transmissions, respectively. The number of APs in the topologies vary from 6 to 18, while clients range from 59 to 47. Back2F consistently outperforms 802.11 across all scenarios. Batched transmissions further reduce backoff overhead and improve throughput by around 5%. The gains with batching is small because basic Back2F has already reduced the backoff duration substantially.

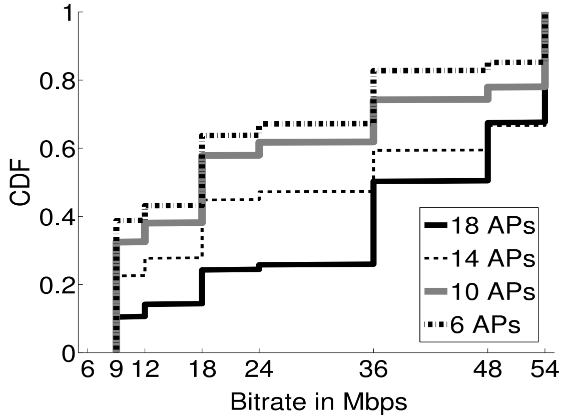

Figure 13: Bitrates of links in the emulated topologies.

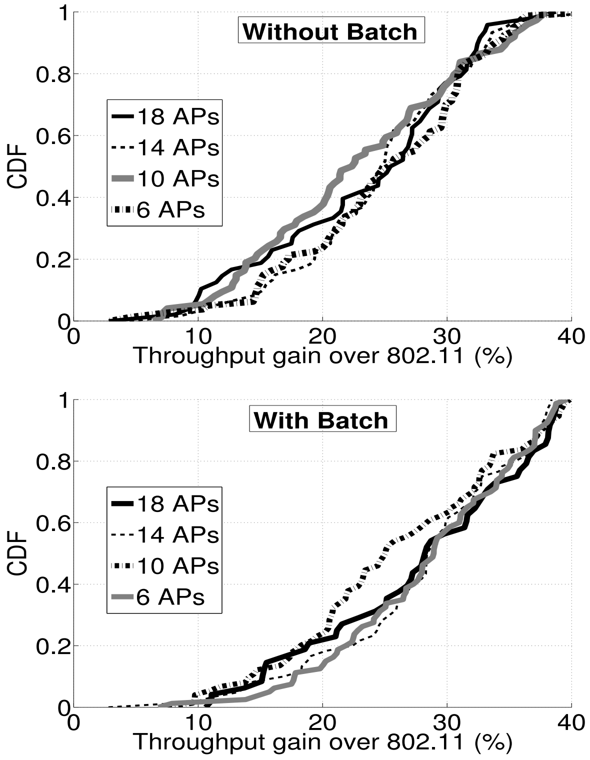

Figure 14: Throughput gain with Back2F over 802.11: (a) without and (b) with batching of transmissions by Back2F. Throughput gain with Back2F ranges from 5% to 38%. Batching contributes 5% of the gain.

Traffic type: Figure 15 reports the throughput gain with Back2F for Skype, Web browsing, and HD streaming traffic. Evidently, the benefits of Back2F are available across all these classes of traffic. Unsurprisingly, gains are better with Skype traffic due to smaller packet sizes. This is because backoff overheads are fixed, making it proportionally larger for short packets.

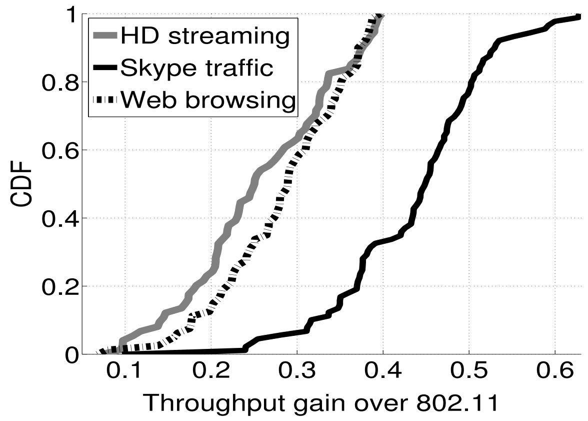

Figure 15: Different types of traffic: Gain with Back2F is more pronounced for Skype traffic with smaller packets.

Fairness: Since Back2F emulates the countdown of 802.11, it is expected to be similar in fairness to 802.11. To verify that, we compute Jain's fairness index on throughput obtained by each AP. Figure 16 shows the fairness index with Back2F and 802.11 for different topology settings. It gives the mean index and confidence interval over 100 instances for each topology size. Clearly, Back2F offers throughput gains, while sustaining fairness comparable to 802.11.

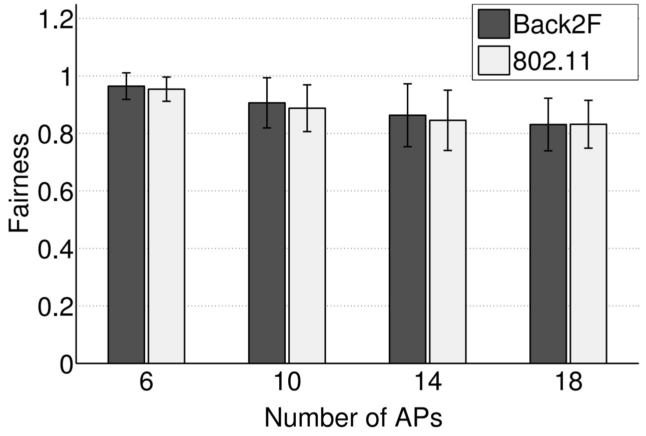

Figure 16: Fairness: Back2F emulates countdown of slots in 802.11 and provides similar fairness.

Impact of channel fading: A Back2F node may fail to detect an active subcarrier (false-negative). We study the impact of such misdetection on Back2F. To cope with misdetection, as explained in Section 3.5, in the second round, a node that picks random number i, transmits on subcarriers i and $((26+i) \mod 52)$. Figures 17 show the degradation from ideal throughout due to false-negatives (ideal throughput obtained with no false positives/negatives). Even with 20% of false-negatives, the resulting throughput degrades by only 5%. In essence, Back2F is a viable scheme that is not overly sensitive to subcarrier misdetection.

Dense networks: The above evaluation investigates the performance of Back2F with up to 18 APs placed in our engineering building. To investigate its scalability to denser networks, we simulated HD traffic in a single collision domain under varying densities and different bitrates. Figure 18 shows Back2F's throughput gain over 802.11, when up to 3 transmissions were allowed per batch. It also presents the performance of Back2F without batch at 54Mbps. Across all settings, Back2F provides gains are in the range of 15% to 30%, suggesting the possibility to scale to large networks.

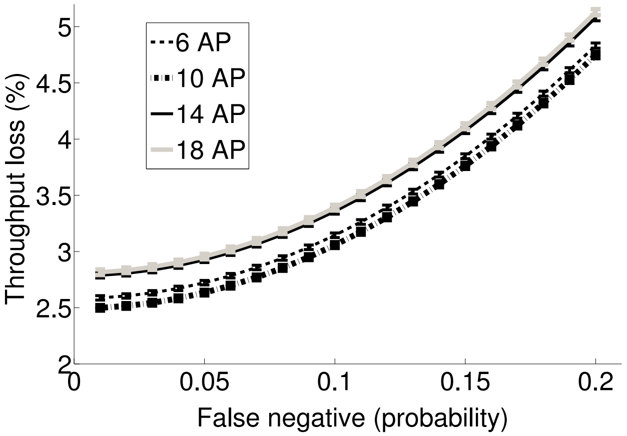

Figure 17: Impact of false negatives on Back2F throughput is shown in the form of deviation from ideal throughput. Due to redundant activation of subcarriers in the second round, the effect of false negatives is minimal.

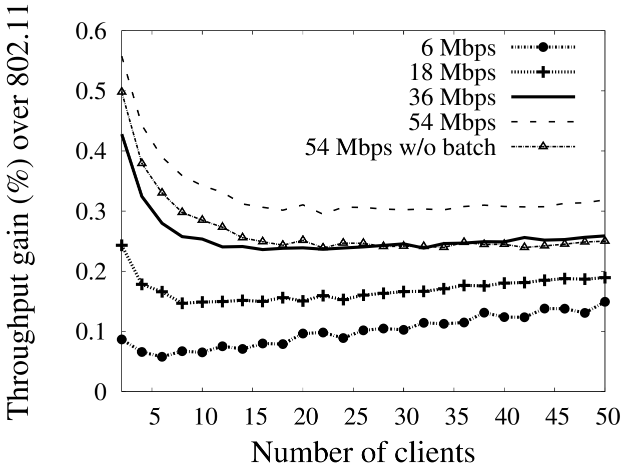

Figure 18: Performance of Back2F in single collision domain: Higher the rate better the gain. Batching (comparison shown only for 54 Mbps) offers around 6% gain.

5. LIMITATIONS AND ON-GOING WORK

Back2F breaks away from a long-standing method of contention resolution; to demonstrate success, it warrants continued research engagement. This paper may be viewed as a first step toward this goal. Several extensions and enhancements remain open for future work.

Robustness of subcarrier detection: The feasibility results in this paper are derived from lab experiments, without node/

environment mobility. Compared to time-domain backoff, Back2F may be more sensitive to channel fluctuations. As discussed earlier, subcarrier detection can be made more robust by striping signals over multiple subcarriers, in order to convey a backoff value. We need to investigate such techniques further and carefully evaluate subcarrier detection under harsh conditions.

Collisions due to hidden terminals: Contention resolution schemes are not designed to cope with hidden terminal problems. However, when using 802.11, the exponential increase in backoff may eventually separate the hidden terminals in time, permitting a successful transmission. Of course, once a success occurs, 802.11 resets its contention window, bringing back the hidden terminal problem. With Back2F, the hidden terminals would continue to collide if they continue to transmit, and in that sense, 802.11 might be slightly better. Nevertheless, we observe that Back2F collisions are solely due to hidden terminals; collisions caused by identical backoff values are far less likely in Back2F. Thus, Back2F can confidently diagnose the cause of collisions, and perhaps turn on RTS/CTS in the face of collisions. 802.11, on the other hand, would still need to discriminate between the cases of identical-backoff and hidden terminals.

Need for an additional antenna: Back2F has to transmit and listen simultaneously only during backoff. The listening antenna can very well act as an additional antenna during normal transmission/reception, such as in a MIMO system. In other words, Back2F is complementary to MIMO. In fact, the feasibility of higher data rates with MIMO emphasizes the need to eliminate idle slots, and thereby adopt Back2F-like schemes. Even without MIMO, given that there are other uses of an additional antenna [25], its inclusion in WiFi devices may very well be worthwhile.

Gain over packet aggregation: 802.11n uses packet aggregation to reduce the contention overhead. The natural question then is whether Back2F is still beneficial. Depending on the type of traffic (e.g., VoIP), aggregation may not be possible nor suitable [27, 29, 30]. Even with packet aggregation, Back2F provides gains at high bitrates. Besides, Back2F addresses a fundamental problem of resolving contention keeping the channel utilization high, regardless of the traffic pattern.

Interoperability with 802.11: We believe Back2F can interoperate with (legacy) 802.11 nodes but may cause unfairness to them. A potential approach to alleviate unfairness is to have Back2F wait for longer than DIFS before participating in a backoff. This gives legacy nodes opportunity to countdown and eventually transmit. We need to understand this interaction, and study the feasibility of incremental deployment of Back2F.

Analysis and Correctness: Back2F emulates the countdown of 802.11 and therefore we believe it is similar to 802.11 in correctness and fairness. We have simulated Back2F for more than 48 hours on various network topologies – we have not encountered deadlocks, starvations, or other correctness problems. However, we have not formally analyzed Back2F's correctness properties. We leave an analytical treatment of Back2F to future work.

6. RELATED WORK

The notion of backoff dates back to 1973, when pure/slotted ALOHA systems [4] were introduced (see [14] for a history on spectrum sharing). The core ideas from ALOHANet have found wide applicability in Ethernet, the Inmarsat satellite network, and most recently, in WiFi [3, 5]. With WiFi's popularity, exponential backoff became a heavily researched topic. Discussing this entire literature is difficult – we only discuss representative ideas, and discriminate them from Back2F.

Regulating Increase/Decrease: One thread of proposals have optimized the manner in which backoff adapts to collisions and network conditions. MACAW [5] proposes doubling of the backoff upon packet loss, but decrease of 1 upon success. PFCR proposed similar policies, but from the fairness perspective [19]. While these and other schemes [20] were appealing for their simplicity, practical measurements [15, 16] and analytical studies [6] show that the inherent inefficiencies remain, and become pronounced in unfavorable conditions.

Contention Estimation: In another research thread, researchers attempted to adapt the backoff scheme based on estimations of network traffic/contention [12]. Unfortunately, such estimations are not always reliable due to unpredictable variations in traffic patterns [7].

Scheduling (Centralized and Distributed): TCF [17] eliminates contention overhead by allocating the channel dynamically using a TDMA-style scheme. Noting the difficulties with synchronization and prediction in TDMA, ZMAC [22] proposed a hybrid MAC allowing CSMA for low contention environments and TDMA for high contention regimes. While creative, performance degradation in low contention regimes, as well as heavy coordination overhead, makes ZMAC impractical for dense networks. Several centralized solutions leverage a central controller to schedule transmissions in single-administrator environments, like offices, airports, etc. [18, 28]. Unfortunately, they do not scale to chaotic networks, such as in residential WLANs or for MiFi networks among personal devices.

PHY based Techniques: Recently, PHY capabilities are being leveraged to redesign higher layer protocols [10, 11, 21, 25, 29]. FICA [29] showed the possibility of signaling on the frequency domain to facilitate fine grained FDMA. While some ideas bear resemblance to Back2F, FICA requires involved RTS/CTS exchanges and a common "referee" node to perform the arbitration, similar to ideas in [23]. Also, the approach in [23] relies on tight time synchronization, that may experience practical challenges in a real system. The protocol in [24] performs frequency domain backoff, however, it does not address multiple collision domains, fairness, starvation, fading, etc. Evaluation is also limited. Back2F demonstrates the feasibility of a distributed backoff mechanism, free of RTS/CTS and referee nodes. The use of an additional antenna, a second round of contention, resilience to fading, packet batching, and a functional prototype, discriminate Back2F from existing work. We believe that Back2F is a departure from established ideas in time-domain contention resolution, begging the question: *what else can be migrated to the frequency domain?*

7. CONCLUSION

Randomization is an effective method of contention resolution in systems with shared resources. Several protocols implement contention resolution by requiring nodes to wait for random durations. During this wait, the channel must remain idle, forcing undesirable under-utilization of channel. This paper proposes a nearly-instantaneous contention resolution method by exploiting the opportunity to operate on the frequency domain (using OFDM subcarriers). A proof-of-concept on a small USRP testbed confirms feasibility and promising performance improvement. Developing a full-scale design is the natural next step.

8. ACKNOWLEDGEMENT

We sincerely thank 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] IEEE 802.11-2007 standard. *standards.ieee.org/getieee802/download/802.11-2007.pdf*.

[2] Intel802.11n channel measurement tool. *ils.intel-research.net/80211n-channel-measurement-tool*.

[3] 802.11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, 1997.

[4] Abramson, N. The Aloha System: another alternative for computer communications. In *Proceedings of the Fall joint computer conference* (1970), ACM.

[5] Bharghavan, V., Demers, A. J., Shenker, S., , and Zhang, L. MACAW: A media access protocol for wireless lan's. In *SIGCOMM* (1994).

[6] Bianchi, G. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on selected areas in communications 18*, 3 (2000), 535–547.

[7] Bianchi, G., and Tinnirello, I. Kalman filter estimation of the number of competing terminals in an IEEE 802.11 network. In *INFOCOM* (2003).

[8] Brodsky, M., and Morris, R. In defense of wireless carrier sense. In *ACM SIGCOMM* (2009).

[9] Choi, J., Jain, M., Srinivasan, K., Levis, P., and Katti, S. Achieving single channel, full duplex wireless communication. *ACM MOBICOM* (2010).

[10] Dutta, A., Saha, D., Grunwald, D., and Sicker, D. SMACK: a SMart ACKnowledgment scheme for broadcast messages in wireless networks. *ACM SIGCOMM* (2009).

[11] Gollakota, S., and Katabi, D. Zigzag decoding: Combating hidden terminals in wireless networks. *ACM SIGCOMM* (2008).

[12] Haas, Z., and Deng, J. On optimizing the backoff interval for random access schemes. *Communications, IEEE Transactions on 51*, 12 (2004), 2081–2090.

[13] He, Y., Yuan, R., Sun, J., and Gong, W. Semi-Random Backoff: Towards resource reservation for channel access in wireless LANs. *IEEE ICNP* (2009).

[14] Jackson, C. Dynamic sharing of radio spectrum: A brief history. In *IEEE DySPAN* (2005).

[15] Jardosh, A., Mittal, K., Ramachandran, K., Belding, E., and Almeroth, K. IQU: practical queue-based user association management for WLANs. In *MOBICOM* (2006).

[16] Jardosh, A., Ramachandran, K., Almeroth, K., and Belding-Royer, E. Understanding congestion in IEEE 802.11 b wireless networks. In *ACM IMC* (2005).

[17] Lim, C., and Choi, C. TDM-based coordination function (TCF) in WLAN for high throughput. In *GLOBECOM* (2005).

[18] Manweiler, J., Santhapuri, N., Sen, S., Roy Choudhury, R., Nelakuditi, S., and Munagala, K. Order matters: transmission reordering in wireless networks. In *Proceedings of the 15th annual international conference on Mobile computing and networking* (2009), ACM, pp. 61–72.

[19] Nandagopal, T., Kim, T., Gao, X., and Bharghavan, V. Achieving MAC layer fairness in wireless packet networks. In *ACM MOBICOM* (2000), pp. 87–98.

[20] Pang, Q., Liew, S., Lee, J., and Leung, V. Performance evaluation of an adaptive backoff scheme for WLAN. *Wireless Communications and Mobile Computing 4*, 8 (2004), 867–879.

[21] Rahul, H., Edalat, F., Katabi, D., and Sodini, C. Frequency-aware rate adaptation and MAC protocols. In *MOBICOM* (2009).

[22] Rhee, I., Warrier, A., Aia, M., and Min, J. ZMAC: A Hybrid MAC for Wireless Sensor Networks. *ACM SenSys* (2005).

[23] Roman, B., Stajano, F., Wassell, I., and Cottingham, D. Multi-carrier burst contention (MCBC): Scalable medium access control for wireless networks. In *IEEE WCNC* (2008).

[24] Sen, S., Choudhury, R., and Nelakuditi, S. Listen (on the Frequency Domain) Before You Talk. *ACM HOTNETS* (2010).

[25] Sen, S., Roy Choudhury, R., and Nelakuditi, S. CSMA/CN: Carrier Sense Multiple Access with Collision Notification. *ACM MOBICOM* (2010).

[26] Sen, S., Santhapuri, N., Choudhury, R., and Nelakuditi, S. AccuRate: Constellation Based Rate Estimation in Wireless Networks. In *NSDI* (2010).

[27] Shin, S., and Schulzrinne, H. Experimental measurement of the capacity for voip traffic in ieee 802.11 wlans. In *INFOCOM* (2007), IEEE.

[28] Shrivastava, V., Ahmed, N., Rayanchu, S., Banerjee, S., Keshav, S., Papagiannaki, K., and Mishra, A. Centaur: Realizing the full potential of centralized wlans through a hybrid data path. In *MOBICOM* (2009).

[29] Tan, K., Fang, J., Zhang, Y., Chen, S., Shi, L., Zhang, J., and Zhang, Y. Fine Grained Channel Access in Wireless LAN. *ACM SIGCOMM* (2010).

[30] Verkaik, P., Agarwal, Y., Gupta, R., and Snoeren, A. Softspeak: making VoIP play well in existing 802.11 deployments. In *NSDI* (2009).

[31] Vutukuru, M., Balakrishnan, H., and Jamieson, K. Cross-layer wireless bit rate adaptation. *ACM SIGCOMM* (2009).