# ADOPT: Practical Add-On MIMO Receiver for Concurrent Transmissions

Sungro Yoon, †Bang Chul Jung, Kyunghan Lee and Injong Rhee
North Carolina State University, †Gyeongsang National University
{syoon4, klee8, rhee}@ncsu.edu, †bcjung@gnu.ac.kr

**Abstract**—Recent MIMO-based techniques allow concurrent transmissions of packets. With signals received from multiple antennas, a receiver can extract concurrently transmitted packets solving linear equations. While the receiver requires essential physical layer parameters for such MIMO decoding, existing signal processing techniques are not capable of obtaining those parameters from interfered preambles. Thus recent proposals enforce communicating senders and receivers go through non-overlapping training period per each packet transmission. For example, senders precisely schedule transmissions such that preambles of the concurrent packets do not overlap. But this coordination process is not trivial and significantly reduces the efficiency of concurrent transmissions. We present ADOPT, a practical add-on MIMO receiver for decoding concurrent transmissions in the absence of coordination. Using the unique combination of novel signal processing technique, called *FDCM*, with existing linear adaptive filters, ADOPT successfully performs packet decodings in uncoordinated transmissions. Using FDCM, ADOPT can obtain the essential physical layer parameters - carrier frequency offset and the start of frames, from overlapped or interfered preambles. The information is then fed into the linear adaptive filter, that in turn extracts each packet successively. As a result, ADOPT builds as a complete black box decoder and increases the link efficiency. We implement ADOPT with 8 USRP2s and test in two of the most common CSMA networks, ZigBee and WiFi. Combined with an aggressive channel access, our extensive experimental evaluations show that the throughput gain becomes up to 231% in WiFi and 319% in ZigBee.

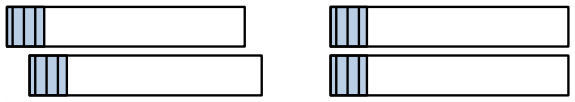
**Index Terms**—MIMO, Multipacket Reception, Receive Beamforming

## 1 Introduction

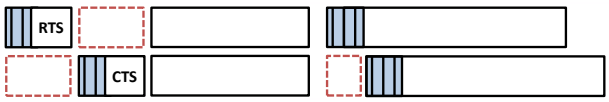
The proliferation of wireless devices caused the public wireless bands, such as ISM (industrial, scientific and medical), heavily saturated. In such a saturated wireless band, when two or more packets are transmitted simultaneously, which frequently happens because of high contention or hidden terminals, the packets become corrupt and cannot be decoded at receivers. Measurement studies [1], [2] over corporate networks show that about 10% of wireless links can suffer more than 70% throughput loss due to collisions from concurrent packet transmissions.

Recently there have been significant advances in the MIMO-based multi-user communication (MUC) [3], [4], [5]. Even though packets are concurrently transmitted and thus interfere with each other, a receiver can decode the packets by solving linear equations with signal inputs from multiple antennas. Naturally, The link capacity becomes larger with more number of antennas.

But those techniques have limitations. Essential parameters required for MIMO decoding, such as *starts of frames*, *carrier frequency offset* (CFO) and *channel state information* (CSI), have been only collected via the non-overlapping, exclusive exchange of known symbol sequences, e.g., packet preambles. The reason is that existing signal processing techniques are mostly designed only for single transmission system and do not consider the cases where the packet preambles are interfered by concurrent transmissions. If a packet preamble is interfered, as Figure 1 (a) shows, a receiver could not perform MIMO decoding. Thus in previous proposals, as Figure 1 (b) illustrates, MUC senders used to exchange separate non-overlapping training packets or perform carrier sense in order to avoid the overlaps in the packet preambles. Such modification in the channel access reduces the applicability of the proposals in the real-world scenarios. Moreover the efforts to coordinate preambles incur inefficiency since the



(a) Overlapped preambles in uncoordinated transmissions




(b) Non-overlapped preambles in coordinated transmissions

Fig. 1. Shaded regions represent packet preambles. (a) When preambles are partially or fully overlapped, current MUC techniques cannot obtain physical layer parameters. (b) MUC techniques arrange transmissions in such ways that the preambles do not overlap. The region in dotted line is where channel is under-utilized.

preambles are usually transmitted at the lowest link rate and so the relative time portion is quite high [5]. Consequently the channel is under-utilized well below the capacity region of the channel, defined by the degree of freedom (DoF).

Our goal is to design and build a MIMO receiver that enables decoding of uncoordinated packets transmitted concurrently. To this end, the main challenges lie in how a receiver should (1) extract the essential physical layer parameters and then (2) effectively decode packets, while the packet preambles are overlapped and thus interfered. The difficulty is that the physical layer parameters are tightly coupled. For instances, conventional decoding techniques do not allow a receiver detect CFO using interfered preambles. With the CFO unknown, it is not possible to detect the start of frames and thus to decode the packets. Therefore, our challenges involve simultaneous acquisition of multiple unknown physical layer parameters such as start of frames, CFO and CSI (channel state information) together from interfered preambles.

As an answer, we present a practical MIMO receiver for concurrent transmissions. We call our approach ADOPT, Add-on Decoder for Overlapping Preamble Transmissions. ADOPT detects the CFO and the start of frames simultaneously using a novel signal processing technique called *frequency domain correlation and matching* (FDCM), which substantially reduces complexity pertaining to detection of CFO and start of frames by comparing received signals and the known preamble in the frequency domain. Then ADOPT decodes each packet utilizing a linear adaptive filter. The technique, which we call *interference suppression and cancellation*, regards overlapping packets as unknown interferences and extracts the packets one by one.

The unique features of ADOPT can be summarized as follow. First, ADOPT increases channel utilization. The channel time is saved since now the transmissions of packet preambles can be fully overlapped. Second, ADOPT can operate independently from channel access schemes, not requiring supports from upper layer protocols. Thus ADOPT is not exclusive with existing MUC protocols and can work with any of them. In other words, ADOPT builds a complete black box decoder. Just replacing the receiver can improve the network throughput.

We implement ADOPT using 8 USRP2s, well known software defined radio platforms, and evaluate performance in WiFi and ZigBee networks. Our extensive experimental results show that ADOPT can be easily combined with the existing standards. When ADOPT is used with a MAC protocol allowing aggressive packet transmissions, the throughput increases up to 319% and 231% compared to native 802.11b and 802.15.4 protocols, respectively.

## 2 Motivation and Background

In order to perform MIMO decoding, a receiver needs to know channel state information (CSI), starts of packets and carrier frequency offset (CFO). However most existing signal processing techniques, commonly occupied in MIMO-based MUC techniques, are originally designed for a single transmission system (e.g., single user MIMO). The challenges can be summarized as follow.

- Acquisition of CSI requires both CFO and start of frames.
- Acquisition of either the CFO or the start of frame requires information of each other.
- When the packet preamble is interfered, both the CFO and the start of frame are unknown.

So MUC proposals require clean (i.e., non-overlapping) preambles to extract physical layer information for MIMO decoding. Enforcing such non-overlapping preamble transmissions degrades the channel utilization as coordination overhead is incurred among senders and receivers.

In this work, we aim to propose a MIMO receiver system that is fully capable of decoding packets without coordination. The system thus requires the acquisition of those parameters from interfered preambles. In the following, we entail about each parameter and discuss corresponding challenges more in detail.

**Channel state information**: Suppose there are two transmitters, each equipped with one antenna, and a receiver with two antennas. When both transmitters transmit, their signals are summed up linearly over the air and the following linear equations are derived:

$$y_1 = h_{11}x_1 + h_{21}x_2 + w_1 \quad (1)$$

$$y_2 = h_{12}x_1 + h_{22}x_2 + w_2, \quad (2)$$

where $y_j$ are outputs from receiving antennas, $h_{ij}$ is the channel coefficient (i.e. CSI) between a transmitter $i$ and a receiver antenna $j$, and $w_j$ is the noise. Given a receiver has CSI, two linear equations have two unknowns ($x_1$ and $x_2$), so we can solve them.

The CSI can be obtained via the channel training process, in which a receiver compares an incoming packet preambles with its knowledge. Typically either linear correlator [6] or minimum mean squared error (MMSE) [5] criteria is used.

The conventional channel training method uses an MMSE-based technique where channel coefficients from all transmitters are explicitly estimated. To estimate the coefficients, MMSE-based channel estimation uses a short training sequence (around 5 symbols). Here it is required that the training sequence is sent in a clear channel without any interference. The challenge is that in uncoordinated concurrent transmissions, the entirety of the preamble may be overlapped with interfering signals. Unless the interfering signals are explicitly modeled and cancelled from the received signals, we cannot use MMSE. Therefore, most of the existing MUC techniques use interference cancellation to remove the interferences. But the problem is that in many situations, the interfering signal cannot be exactly modeled. And it is the reason why MUC protocols coordinate the senders to avoid preamble overlaps.

**Start of frames**: When there is only a single transmission, a receiver can immediately know the start of that frame by observing an energy level. In case multiple packets arrive at a receiver concurrently, the starting positions should be accurately detected for measuring CSI as a next step. It is because, the receiver needs to know when to start the training process between incoming signals and its known sequences.

Time domain correlation is one of the techniques that can be used to find the starting positions. The correlation

observes similarity between incoming signals and a priori known symbol sequence. For a given known sequence of samples, $x = x_0, x_1, \ldots, x_p$ and an input timed series of symbols at time $t$, $y_t, y_{t+1}, \ldots$, correlation multiplies each component independently and sums them up as follows:

$$C = \sum_{i=0}^{p} x_i y'_{j+i}. \quad (3)$$

Then it shifts the alignment by one sample of the input and obtains another correlation value. The start of a frame is estimated to be the sample index of the first sample that produces a spike, the maximum correlation value (e.g., [6], [5]). As we will see, the correlation does not work well with carrier frequency offsets. Bottom of Figure 5 shows the preamble correlation result when multiple packets overlap. The correlation is performed without CFO compensation. As shown, the correlation does not yield a distinguishable peak if without proper CFO compensation.

There are other techniques for collision detection. SoftPHY [7] detects the increase of sample dispersion in the constellation map when a packet is overlapped with another. This technique is effective in detecting the presence of errors incurred by packet collision, but cannot be used to find the collision points of more than two overlapped packets since dispersions from two packet collision are hard to be distinguished from three or more packet collision. It is proposed [8], [9] to use the increase in the amplitude variance within received signals when packet collision occurs. This technique is effective only if signal strength is very strong; otherwise, the amplitude variance is easily influenced by the background noise.

**Carrier frequency offset**: CFO is caused by (1) natural differences in the carrier frequencies, generated by respective oscillators of a sender and a receiver or (2) doppler shifts due to movements [10], [11]. IEEE 802.11n [12] defines the allowable CFO range of $\pm$60KHz at 2.4GHz, $\pm$100KHz at 5GHz and 802.15.4 [13] allows $\pm$96KHz at 2.4GHz channel. CFO incurs *phase rotations* in the received samples. For instance, a sinusoidal wave with a frequency $f_{Tx}$ sampled at a periodic interval $1/f_{Tx}$ will produce the same value. But with CFO, the values sampled at every $1/f_{Rx} = 1/(f_{Tx} + \Delta f)$, even starting from the exact same phase, will have displacements and eventually produce different values. In order to decode a packet, a receiver should accurately model the CFO.

Correlation is not effective in the presence of CFO. A correlation value at a start of a frame will be given as:

$$C(t) = h \sum_{k=0}^{p-1} |x_k|^2 e^{2\pi i k \Delta f T}, \quad (4)$$

where $p$ is the preamble length, $x_k$ is the $k$-th symbol in a preamble, $h$ is the channel coefficient vector, $T$ is the symbol duration and $\Delta f$ is the CFO. With the exponential term $e^{2\pi j k \Delta f T}$, CFO produces a phase rotation of each symbol $x_k$ which cancels each other during the summing process. Finding the correlation peak over the time domain, therefore, must entail cancelling out the exponential term, which is not possible without knowing $\Delta f$.

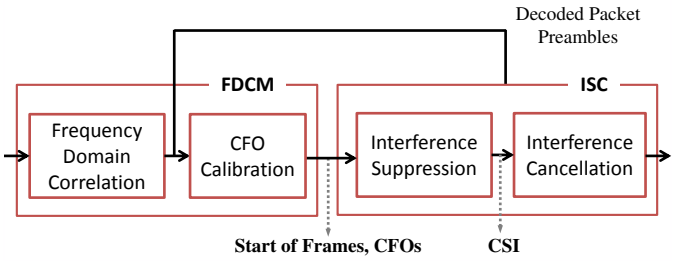



Fig. 2. Block diagram of the ADOPT decoder.

CFO can be obtained via measuring the amount of phase rotation per symbol within a known symbol sequence. However in case a receiver receives concurrently transmitted packets, the process is not trivial. While the receiver has to determine the starting position of a packet to compute CFO, the correlation technique does not work without compensating for CFO.

It has been suggested that CFO is precomputed for all the possible senders when a packet from each is sent alone in a clear channel [6], [3], [14]. The problem is that CFO can be incurred by doppler shifts and in this case precomputed values will be inaccurate. But more fundamentally, when several packets overlap in their preambles, there is no way that the receiver can identify the transmitter of each received packet and thus which CFOs to apply accordingly. A brute force technique is to try all the possible CFOs, but it is too time-consuming and prohibitive in terms of computational power. So this approach might be applicable to a small, static network. Another approach lets all the senders tune their carrier frequency to the respective receiver when they transmit [15], [4], [16], [17]. But the synchronization overhead could be non-negligible. Also in case there could be errors in the synchronization, the receiver still needs the ability to compensate for CFOs.

## 3 Proposed System: ADOPT

ADOPT decodes packets from the overlapped signals using the information obtained from overlapped preambles. The ADOPT architecture mainly consists of two components: *frequency-domain correlation and matching* (FDCM), and *interference suppression and cancellation*. FDCM is a novel technique that performs 2-dimensional search to find both (1) CFO and (2) starts of packets from incoming overlapped preambles, with significantly low computational overhead. With these parameters, we combine interference suppression and cancellation techniques to decode the concurrently transmitted packets. ADOPT is deliberately designed to serve as a black box decoder, that operates independently of upper layer protocols. Figure 2 shows our system design.

### 3.1 FDCM

FDCM utilizes frequency spectrum of a packet preamble and that of the incoming signals. We have timed input samples $y = y_0, y_1, y_2, \ldots$ and samples from the preamble $x = x_0, x_1, \ldots, x_{p-1}$. We take $p$ samples from both and apply FFT to obtain frequency spectrum of each. Correlation is performed

between frequency spectrum of the two sample sequences. This is a two dimensional search. We find a frequency and time pair $(\delta, \tau)$, where $\delta$ is the frequency value that produces a correlation spike and $\tau$ is the time instance when $\delta$ has been found, respectively. This process is repeated until the end of the received samples by shifting one sample of the input signals at a time to get another $p$ samples.

The intuition behind FDCM is simple: an exponential change in the time domain becomes linear in the frequency domain. This is also known as *shift theorem*. So the exponential term $e^{2\pi ik\Delta fT}$ in Eq. (4) becomes a constant frequency shift in the frequency domain. If the frequency spectrum of the preamble sequence has a low autocorrelation in the frequency domain, then the unknown frequency offset $\Delta f$ can be estimated by finding the peak in frequency domain correlation. The intuition is analytically explained by the following.

When DFT (discrete Fourier transform) is applied to a fixed number of received signal samples of a packet, the frequency spectrum at each frequency $f_k$ is represented as

$$S_r(f_k) \quad = \quad H(f_k) \cdot S_p(f_k - \Delta f), \tag{5}$$

where $S_r(f_k)$, $S_p(f_k)$, $H(f_k)$ and $\Delta f$ denote the frequency component of the received signals at frequency $f_k$, the original frequency of the preamble (i.e., DFT results of the preamble), the $k$-th frequency response of the wireless channel and the CFO, respectively. Note that $f_k$ is a discrete value such that $f_k = \frac{kR}{2N}$ Hz, where $k \in \{0, \ldots, N-1\}$, $N$ is the number of samples used for DFT (i.e., the number of frequency bins) and $R$ is the sampling rate.

By taking correlations between the DFT results of received signal and the known preamble, it is possible to reliably detect both the start of frames and CFOs. The correlation is computed for each $\delta \in \{f_0, \ldots, f_{N-1}\}$ as below.

$$\begin{aligned} C(\delta) \quad &= \quad \sum_{k=0}^{N-1} S_r(f_k + \delta) \cdot S_p^*(f_k) \\ &= \quad \sum_{k=0}^{N-1} H(f_k) S_p(f_k + \delta - \Delta f) \cdot S_p^*(f_k), \end{aligned} \tag{6}$$

where $S_p^*(f_k)$ is the complex conjugate of the $k$-th frequency component of the preamble sequence.

Now, suppose a receiver has received packets $P_a$ and $P_b$ simultaneously, whose frequency spectrum are frequency spectrum $S_a$ and $S_b$, respectively. Then $S_r(f_k)$ is the linear sum of each packet's frequency components:

$$\begin{aligned} S_r(f_k) \quad &= \quad H_a(f_k) \cdot S_p(f_k - \Delta f_a) \\ &+ \quad H_b(f_k) \cdot S_b(f_k - \Delta f_b) + W(f_k) \end{aligned} \tag{7}$$

where $H_a(f_k)$ and $H_b(f_k)$ indicate the $k$-th frequency responses, $\Delta f_a$ and $\Delta f_b$ are the CFOs of packets $P_a$ and $P_b$, respectively. $W(f_k)$ represents the $k$-th frequency response of the background noise at the receiver. We take correlation in frequency domain and from Eq. (6) and Eq. (7), the result at $\delta = \Delta f_a$ is computed as:

$$\begin{aligned} C(\Delta f_a) \quad &= \quad \sum_{k=0}^{N-1} S_r(f_k + \Delta f_a) \cdot S_p^*(f_k) \\ &= \quad \sum_{k=0}^{N-1} [H_a(f_k) S_p(f_k) \cdot S_p^*(f_k) \\ &+ \quad H_b(f_k) S_p(f_k + \Delta f_a - \Delta f_b) \cdot S_b^*(f_k) \\ &+ \quad W(f_k + \Delta f_a)) \cdot S_p^*(f_k)]. \end{aligned} \tag{8}$$

If the preamble has a low autocorrelation and a cross-correlation in the frequency domain, the second and third terms will cancel out and Eq. (8) is simplified as:

$$C(\Delta f_a) \quad \simeq \quad \sum_{k=0}^{N-1} H_a(f_k) \cdot |S_p(f_k)|^2, \tag{9}$$

which produces a spike at $\delta = \Delta f_a$. We argue that our assumption of low autocorrelation and low cross-correlation of preambles in the frequency domain is valid in reality. We have found experimentally that the packet preambles of CSMA standards such as ZigBee and WiFi satisfy the same property in the frequency domain. Figure 3 shows the autocorrelation of the ZigBee preamble in the frequency domain.

FDCM can reliably detect the CFOs of packets as well as starting points even when two or more packets arrive at the receiver exactly at the same time instances. Such cases frequently happen in a dense network where the contention is severe. If we replace $S_b$ in Eq. 8 with $S_p$, we will see two

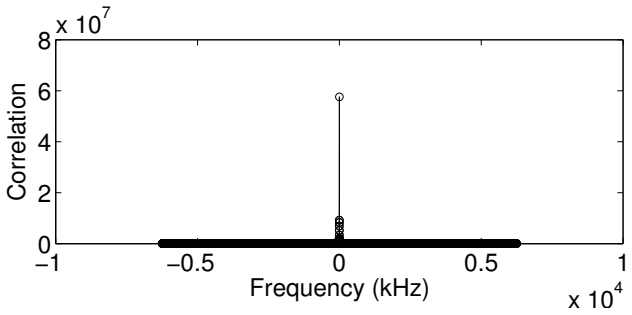



Fig. 3. Autocorrelation of ZigBee preamble in frequency domain.

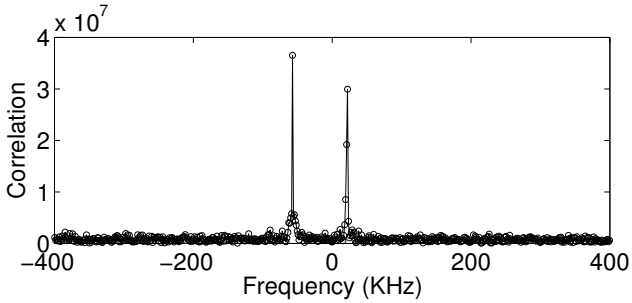



Fig. 4. Two ZigBee packets are transmitted at the same time slot. Each packet has CFO of -55.5KHz and 21KHz, respectively. ADOPT reliably detects perfectly aligned packets and their CFOs.

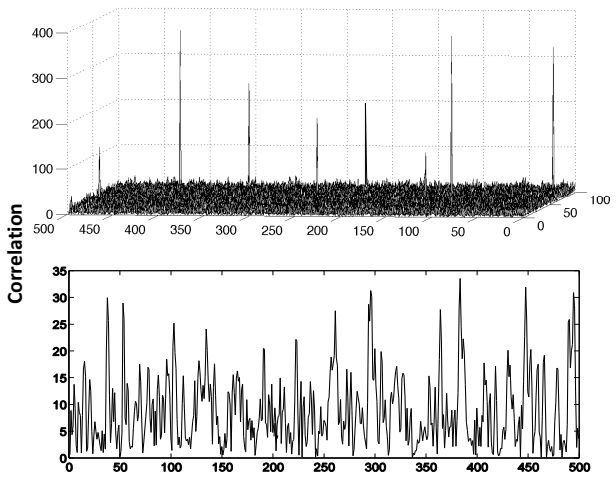



Fig. 5. Top: FDCM results of 8 concurrently transmitted ZigBee packets. 8 peaks identify both CFOs (y-axis) and starting positions (x-axis) of 8 packets. Bottom: correlation of the same signal in time domain. No distinct peaks are visible.

spikes at the frequency index of $\Delta f_a$ and $\Delta f_b$. As Figure 4 shows, FDCM produces multiple spikes at the start of each packet when the CFOs of the senders are different from each other. This is very likely in reality. After detecting multiple packets and their CFOs, the receiver can start to decode each packet. Our technique of decoding the overlapped packets is explained in Sec. 3.2.

Figure 5 compares the outpus from the proposed FDCM and the conventional time domain correlation. Our test is performed on the received signal dump consisting of eight collided ZigBee packets. In the conventional time domain correlation, the CFOs of collided packets have not been compensated. As Eq. (4) describes, the time domain correlation value is substantially distorted and the result does not easily identify any start of packet. On the other hand, FDCM clearly identifies starts of packets as well as CFOs by the $(time, frequency)$ coordinates of the peaks.

To the best utilization of FDCM, the threshold for detecting the peaks needs to be determined carefully. If the threshold is too low or high, the false positive or negative probability becomes too high. In Section 5, we empirically provide a threshold value which gives very low false negatives while maintaining a reasonable level of false positives.

### 3.1.1 Complexity Reduction

Despite its efficacy, high complexity of FDCM would make it less practical. For each incoming sample, FDCM computes FFT on $N$ samples and perform correlations. While the FFT takes $O(N \cdot logN)$, the correlation operation compares the $N$ samples at $N$ different frequencies and thus involves $O(N^2)$ computation.

Now we propose FFDCM (fast FDCM) that substantially reduce the complexity and the computational overhead. FFDCM exploits following observation: the frequency spectrum of a packet preamble exhibit high energy only at a few frequency

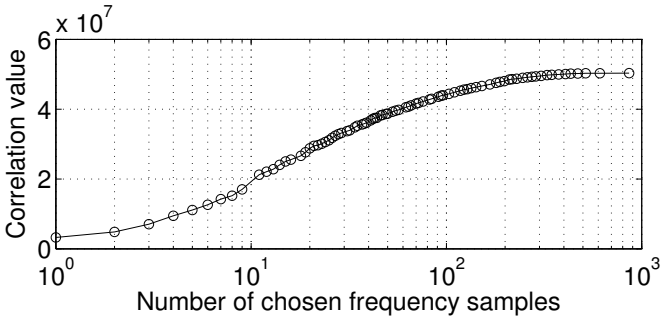



Fig. 6. The maximum correlation value when the number of chosen frequency samples is varied.

bins. But most of the frequency bins at other frequencies carry energy that almost amounts to zero. This is due to the repetitive nature of symbol patterns in the preamble. For example, ZigBee preamble repeats the same sequence of 32 chips 8 times and WiFi (802.11b) preamble repeats identical 11 chips 192 times. By using only the FFT points with high energy, FFDCM can find the almost the same correlation results while having the significantly reduced computational overhead.

FFDCM computes the correlation in the frequency domain at selected $(f_i, S_r(f_i))$ and $(f_i, S_p(f_i))$ pairs. $f_i$ is chosen such that $|S_p(f_i)| > thresh$. Now Eq 6 rewrites as

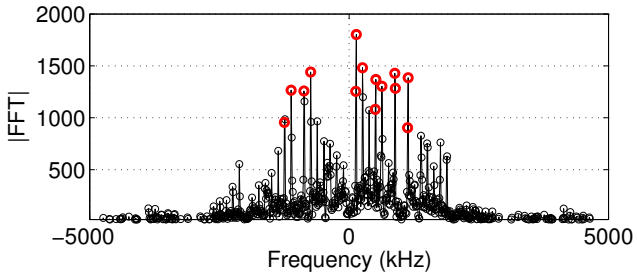
$$\begin{aligned} C(\delta) &= \sum S_r(f_i + \delta) \cdot S_p^*(f_i),\ f_i \in \{f_0 \dots f_{k-1}\} \\ &= \sum H(f_i) S_p(f_i + \delta - \Delta f) \cdot S_p^*(f_i). \end{aligned}$$

Given $k$, the total number of chosen FFT points, FFDCM takes $O(k \cdot N)$ computations at each incomping signal sample. So the overall complexity is upperbound by FFT, which takes $O(NlogN)$. Figure 6 shows our experimental result from ZigBee testbed, where the the maximum correlation values from FFDCM are shown with the varying number of chosen frequency We see that use of 20 frequency samples still yields around 60% correlation peak compared to the case when all of 1024 samples are used. As an example, in figure 7(a) we take 14 of the most significant FFT points out of total 1024 points. Despite the complexity reduces by 80 folds, figure 7(b) clearly shows that the detection performance is still maintained.

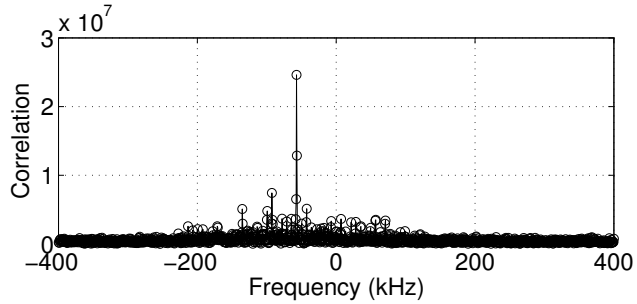
### 3.1.2 CFO Calibration

The number of samples used for FFT determines the resolution of CFO estimation. However as the length of preamble is fixed and cannot be extended arbitrarily, there always exists a certain range of estimation errors. We call this *residual CFO*. For example, as for IEEE 802.11b, we have $144 \times 11 = 1584$ symbols in the preamble (144 symbols each being spread using 11 chips). If we use 20MHz radio receiver, we can detect CFO with $20M/2/1584 = \pm 3.16$KHz resolution. We can further refine the estimate by increasing the number of frequency bins, for example, using interpolation or oversampling. But it could significantly increase the complexity or the energy consumption in the receiving device.

For the remedy, we calibrate the coarse CFO as follows. The calibration is a two step process: a receiver (1) decodes the




(a) Frequency spectrum of the ZigBee preamble. Red circles shows total 14 points that have higher amplitudes.




(b) Correlation result of the captured ZigBee signals (SNR = 2dB) and the selected FFT points of ZigBee preamble.

Fig. 7. Using only the significant FFT samples, FFDCM is able to reliably perform correlation in the frequency domain with substantially reduced computational overhead.

preamble using the coarse CFO estimates and (2) estimate the residual CFO using the decoded preamble. Before attempting to decode data portion of a packet, we apply MIMO decoding only to the preamble and extract the preamble symbols. Since the preamble size is small and the current CFO estimate is obtained using the entirety of that preamble, the coarse CFO is *always* good enough to decode it. Note that the symbol phase rotation due to the residual CFO error is limited to at most one during that time. With those symbols completely known, the residual CFO can be estimated by measuring the amount of phase rotation between the first and last symbols of the preamble. We extract a preamble using an adaptive filter, which is explained in Sec. 3.2. As the adaptive filter has the linear complexity, it takes $O(P)$ computations where $P$ is the preamble length.

## 3.2 Interference Suppression

A typical MIMO receiver requires CSI to solve Eq. (2), for the decoding of concurrent packets. However explicit modeling of CSI is often not feasible in uncoordinated networks where preambles overlap. In order to avoid the explicit estimation of CSI, ADOPT uses a linear adaptive filter. The adaptive filter forms $\mathbf{W}$ directly from the received signals without explicitly modeling individual CSI [18]. Here $W$ is defined as $n \times m$ matrix such that

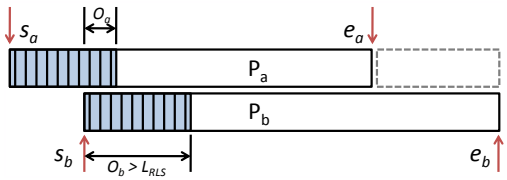



Fig. 8. Interference suppression using RLS; RLS can be applied to decode $P_a$ and $P_b$.

$$x = W \cdot y, \tag{10}$$

where $y$ is the received symbol vector with length $m$ and $x$ is the transmitted symbol vector with length $n$. The main benefit of using the adaptive filter is that it can extract a packet in the presence of unknown interference signals. $W$ converges via an iterative process to produce minimum estimation error, even when the sequence is overlapped with other signals [18].

The adaptive filter has not been actively used for the channel training purposes in the context of concurrent transmissions. It is because of its requirement of a relatively long symbol sequence [19]. However, given that most wireless communication standards (e.g., 802.11 and 802.15.4) have sufficiently long preambles with low autocorrelation, we can make use of adaptive filters for decoding concurrent packets.

Among several different choices (e.g., LMS, Kalman, etc.), we choose *recursive least square* (RLS) for our implementation and evaluation. RLS has the smallest error in a static environment where the channel state changes relatively slowly (e.g., nodes are stationary) [1]. RLS uses the following iterative algorithm:

$$\begin{aligned} W_i &= W_{i-1} + (x_i - W_{i-1} y_i) y_i^* \rho_i \\ \rho_i &= \lambda^{-1}[\rho_{i-1} - \frac{\lambda^{-1}\rho_{i-1} y_i y_i^* \rho_{i-1}}{1 + \lambda^{-1} y_i^* \rho_{i-1} y_i}], \end{aligned}$$

where the initial value of $\rho_i$ is given as $\rho_{-1} = \epsilon^{-1} I$ with $\epsilon$ being a very small constant. $\lambda$ is a forgetting factor, $0 \ll \lambda < 1$. $W_i$ converges iteratively with an error feedback $x_i - W_{i-1} y_i$ [20], [18].

### 3.2.1 Decoding of two packets in overlap

Now consider a scenario where two packets $P_a$ and $P_b$ are overlapped as in Figure 8. We show how packet $P_b$ can be decoded in the presence of $P_a$. Let $L_{\mathrm{RLS}}$[2] be the minimum preamble length required for training an RLS filter, and $O_i$ be the time duration that the preamble of packet $i$ overlaps with the other packet. To simplify the discussion, we assume that starts of packets and CFOs have been already identified via FDCM.

ADOPT always first decodes a packet that has arrived later. The reason is that, in order to remove certain interfering signals from a packet, $W$ should be built upon the preamble

1. We can consider switching to LMS (least meansquares) for a fast time-varying channel, but we leave it as future work.

2. We interchangeably use $L_{\mathrm{RLS}}$ in the unit of symbols or time unless it makes confusion. Actual $L_{RLS}$ value is studied in Sec. 5

that is interfered by those signals. In Figure 8, the preamble of $P_b$ fully overlaps with $P_a$. So RLS can train $W$ such that it can cancel out the interference of $P_a$. However if $W$ is trained using the preamble of $P_a$ and if $O_a < L_{\text{RLS}}$, RLS cannot secure a sufficient number of iterations to obtain the information about $P_b$'s interference. Thus decoding of both $P_a$ and $P_b$ will fail. After deciding to decode $P_b$ first, ADOPT adjusts received signals to compensate for the CFO of $P_b$.

Now denote $\mathbf{y}$ as an $m$-rank vector of received symbol values given that a receiver has $m$ antennas. Since the number of concurrent transmitters is unknown, we set $\mathbf{x}$ to have the same rank as $\mathbf{y}$ such that $x_i = [s_{1,i}, \ldots, s_{m,i}]^T$ where $s_{k,i}$ represents the $i$-th preamble symbol from the $k$-th transmitter. Suppose that $P_b$ is transmitted by a node $k$. ADOPT treats all the interferences such as $P_a$ as unknown during the decoding process of one packet, $P_b$. So all the symbols from the other nodes are assumed to be zero. So we have $x_i = [0, 0, \ldots, s_{k,i}, 0, \ldots, 0]^T$. Training $W$ over $L_{\text{RLS}}$, we decode $P_b$ from the simple operation $\hat{x} = W \cdot y$. Note that the later portion of $P_b$, not overlapped with $P_a$, can be treated as if $P_a$ is transmitting virtual null signals (denoted by a dotted box in Figure8). This portion does not affect the result of operations. Recall that if a receiver knows about all the coefficients in Eq. (2), it can solve the equation to obtain $x_2$ even when $x_1$ is 0.

This technique is considered as *interference suppression* because interfering signals are not explicitly modeled, but instead treated as unknown interferences. RLS filtering can be understood as a receiver side beamforming [21] that adjusts the antenna gains to suppress the interferences without individually modeling each signal.

After $P_b$ is successfully decoded, it is possible to cancel out the contribution of $P_b$ from superimposed signals. Of course, if $O_a > L_{\text{RLS}}$, it is possible to directly apply RLS to decode $P_a$. Otherwise ADOPT utilizes the interference cancellation technique [8], [3], [5]. Since $P_b$ is decoded using RLS, it can be subtracted from the mixed signals to expose $P_a$ in a clear channel. As this technique is extensively studied in the literature, we omit the detail.

We considered an example where two packets have been transmitted at different time instances and thus have distinct starting points $s_a$ and $s_b$. This is a common colliding pattern for hidden terminals. In this case, RLS could use an identical preamble as training sequences for the two different packets. It is because the preamble has the low autocorrelation property and the packets overlap at different symbol indices.

But what happens if two senders with identical preambles begin transmissions at a same time instance and so the starts of the packets are perfectly aligned? Note that the two packets usually have different CFOs. As ADOPT extracts one packet at a time, say $P_b$, all the incoming signals are adjusted to compensate for the CFO of $P_b$. Then as Eq.(4), the preamble of another packet, $P_a$, will look like a random pattern when seen in the time domain. RLS can decode $P_b$ by considering $P_a$ as an unknown interference. In an actual network, it is extremely rare that two transmitters have the same CFO and also their packets are received in a completely aligned manner.

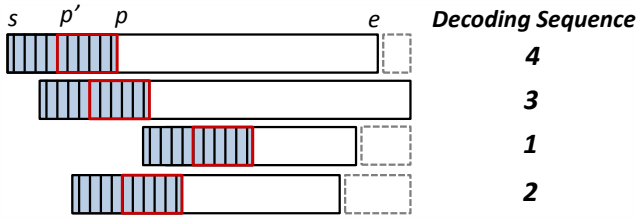



Fig. 9. There is always at least one RLS-decodable packet in any overlapping set. A packet with the latest time interval $[p', p]$ is always RLS-decodable.

### 3.2.2 Decoding of three or more packets in overlap

Now we consider cases when more than two packets are overlapped. Figure 9 shows an examples. We show that interference suppression and cancellation discussed in Section 3.2 can be extended to decode more than two concurrent packets.

We denote the start and end times of a packet $P_i$ to be $s_i$ and $e_i$, and the time at which its preamble ends to be $p_i$. So $s_i < p_i < e_i$. We say that two packets $P_i$ and $P_j$ are *directly overlapped* if $s_i \leq s_j \leq e_i$ or vice versa. $P_i$ and $P_j$ are *transitively overlapped* if (1) there exists a sequence of packets $P_1, \ldots, P_l$ such that $P_x$ and $P_{x+1}$ $(1 \leq x < l)$ are directly overlapped and (2) $P_i$ and $P_1$, and $P_l$ and $P_j$ are directly overlapped respectively. We define an *overlapping set* $\mathcal{S}$ to be the set of packets such that any two packets, $P_i$ and $P_j$, are directly overlapped or transitively overlapped.

*Lemma 3.1:* A packet $P_k$ in an overlapping set $\mathcal{S}$ is decodable by RLS if there is no directly overlapped packet in $\mathcal{S}$ that starts after $p'_k := p_k - L_{\text{RLS}}$.

*Proof:* Here $p_k$ is the ending point of the preamble. The lemma can be informally shown as follows. By the definition, all the packets in the overlapping set of $\mathcal{S}$ are directly or transitively overlapping with $P_k$. Recall that RLS filtering can train $W$ if the training duration is longer than or equal to $L_{\text{RLS}}$. As no directly overlapped packet starts after $p'_k$ by the assumption, the RLS filter can be sufficiently trained during a period $[p'_k, p_k]$ and successfully decode $P_k$. However, if there is any directly overlapped packet starting after $p'_k$, RLS does not have enough training symbols to converge to a good filter that is able to decode $P_k$. □

We say that $P_k$ in Lemma 3.1 is *RLS-decodable*. In any overlapping set, we can prove that there is at least one packet that is RLS-decodable.

*Lemma 3.2:* In any overlapping set $\mathcal{S}$, there exists at least one RLS-decodable packet.

*Proof:* We can prove this by contradiction. Suppose that every packet has some directly overlapping packet starting after the time instance $p'_k = p_k - L_{\text{RLS}}$. So none of the packets in $\mathcal{S}$ is RLS-decodable. Consider a packet $P_l$ whose ending time $e_l$ is the last among those in $\mathcal{S}$. By the contradiction hypothesis, there is at least one directly overlapping packet $P_j$ starting after $p'_l$. Then since $P_l$ finishes last, $P_j$ must end before or at $e_l$. Then the length of $P_j$ is less than $P_l$ because $P_j$ starts later than $P_l$ and ends before $P_l$ or at the same time with $P_l$. By the same logic, $P_j$ must have another directly overlapping packet starting after $p'_j = p_j - L_{\text{RLS}}$ and ending

before or at $e_j$. Again, the length of that packet will be less than $P_j$. This argument continues to reduce the packet length further, and eventually we will have a packet whose length is less than the length of preamble. This is a contradiction because the length of a packet must be larger than that of preamble. □

In general, a packet $P_k$ with the latest time interval $[p'_k, p_k]$ is always RLS-decodable. After the RLS-decodable packets are decoded, we can cancel out their signals from the original signals so that no packets in the residual signals are overlapping with the decoded packets. Then we have $\mathcal{S}' = \mathcal{S} - \{P_k\}$. Lemma 3.2 also applies to $\mathcal{S}'$ and it has at least one RLS-decodable packet. By induction, we can claim that all packets in $\mathcal{S}$ can be sequentially decoded. Figure 9 shows examples of overlapping sequences of packets which illustrates the order of RLS-decoding.

#### 3.2.3 Complexity

RLS is a linear filter so it will take $O(P)$ for training where $P$ is the preamble length. The actual filtering will take $O(n)$ for $n$ samples in a packet. Interference cancellation also takes $O(n)$.

## 4 Implementation and Setup

In this section, we describe our implementation of ADOPT using 8 USRP2s running GNU radio platforms, and the configuration of our test. ADOPT is implemented using 8 USRP2s with GNU radio platform. We test ADOPT in ZigBee (IEEE 802.15.4) and WiFi (IEEE 802.11b). While ADOPT can work with any type of channel access schemes, we choose these two networks to demonstrate the general applicability of ADOPT as they are two of the most common CSMA networks.

### 4.1 Hardware and Software Setup

An ADOPT receiver consists of eight USRP2, each attached with one RFX2400 daughter board. The system forms an 8 antenna MIMO receiver. Each USRP2 is connected to a PC through a gigabit Ethernet. An USRP2 dumps the incoming digital samples received from its transceiver to the PC. We operate USRP2s at the sampling rate of 25 million samples per second. This signal dump is analyzed offline. Figure 10 (a) shows the ADOPT receiver. Note that we do not necessarily synchronize those USRP2s because it is possible to accurately detect start of each frame in each signal dump using FDCM.

To investigate the performance of ADOPT in testbeds, we deployed 18 MICAz nodes each with a CC2420 ZigBee compliant transceiver running TinyOS and 40 WiFi nodes running MadWiFi driver over Atheros 802.11b cards. Note that because of its communication overhead with PCs the USRP2 receiver cannot communicate with senders in realtime (e.g., cannot ACK). Instead, clients are associated with a native AP (or sink) which is different from the ADOPT receiver and they directly communicate with that AP. An ADOPT receiver is placed right next to the AP and it passively receives incoming packets. For the simplicity, we do not test multiple AP scenarios, but instead we adjust the carrier sensing thresholds of clients in order to manually create hidden terminals.

The deployment maps of ZigBee and WiFi networks are shown in Figure 10. The received SNR at the AP is between -7dB to 12dB, and 2dB to 35dB for the ZigBee and WiFi testbeds, respectively. ZigBee nodes have CFOs in the range between -55KHz and 40KHz and WiFi nodes have between -40KHz and 25KHz with normal distribution around the center frequency.

### 4.2 Physical Layer Characteristics

Both ZigBee and WiFi use CSMA/CA for media access control and transmit in the 2.4GHz ISM band. However, they have completely different physical layer characteristics that include physical rate, coding, modulation and preamble length. While CC2420 transmits at a fixed data rate of 250Kbps, IEEE 802.11b can transmit at various data rates ranging from 1Mbps to 11Mbps. We choose 11Mbps for WiFi and disable the rate adaptation.

ZigBee maps every 4 bits into 32 chips of pseudo noise codes. Then every two chips are modulated into one OQPSK symbol by the CC2420 chipset. 11Mbps IEEE 802.11b uses DSSS with CCK of 8 chips per symbol and every two chips are converted into a DQPSK symbol. While DQPSK is a differential modulation/demodulation technique designed for the non-coherent detection, we manually map it to QPSK for the coherent detection.

ZigBee uses a base bandwidth of 5MHz and WiFi (IEEE 802.11b) uses 20MHz. ZigBee typically has higher effective SINR than WiFi 11Mbps because of longer spreading codes. The preamble length is also different between ZigBee and WiFi. They use 256 chips (ZigBee) and 616 chips (802.11b 11Mbps), respectively. As will be shown through experiments, these preamble sizes are sufficiently long for RLS filtering since RLS performs well with a preamble size of 20 chips or longer.

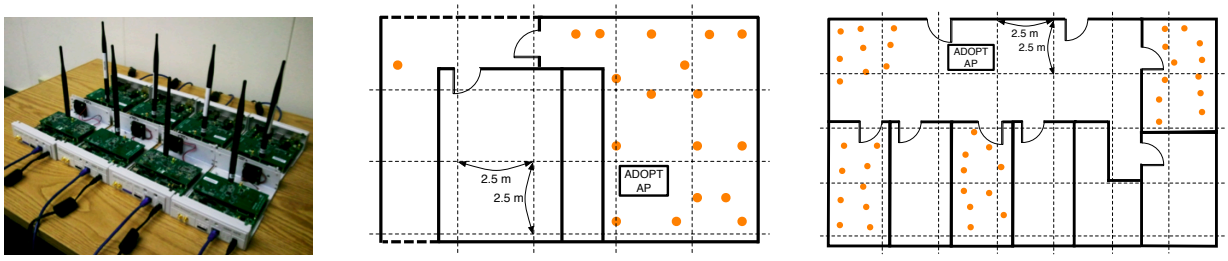
## 5 Evaluation

In this section we evaluate the performance of ADOPT using ZigBee and WiFi senders. We first discuss about factors that affect the performance of ADOPT, and see BER and normalized throughput in controlled environments. Then we perform testbed evaluation to show that ADOPT can work with non-cooperative protocols and ADOPT increases the network performance. In all tests, We use native CSMA/CA clients. Unless specified, we use 8 USRP2s for the ADOPT receiver. We use UDP packets of 128 bytes for ZigBee and 1500 bytes for WiFi.

### 5.1 Performance in Controlled Environments

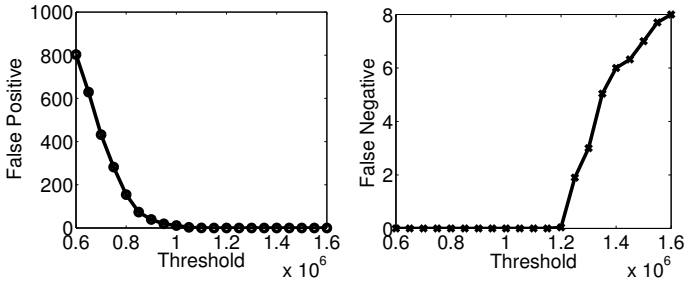
#### 5.1.1 FDCM threshold

FDCM requires a threshold value that provides a good separation of collision points and CFOs. We place 8 MICAz nodes to transmit a set of data packets back to back to their respective AP. FDCM is performed using the ZigBee preamble, together with the various threshold values. All nodes are configured to transmit packets to have the same SNR of 15dB at the receiver. Figure 11(a) plots the false positive and negative instances for




(a) ADOPT receiver (b) Map of 18 ZigBee nodes (c) Map of 40 WiFi nodes

Fig. 10. ADOPT hardware setup and two testbed networks.




(a) False positive and negative for various correlation thresholds

Fig. 11. The detection performance of FDCM for various correlation threshold values.

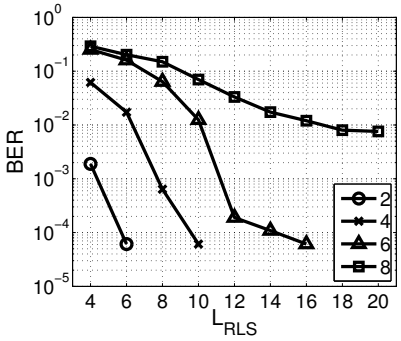



Fig. 12. BER of ADOPT with different $L_{RLS}$.

various threshold values. Having too many false positives costs computation while having many false negatives reduces the number of decodable packets. In the figure, the threshold is computed over the signals after AGC (automatic gain control) is applied. We observe that the threshold value around $1.1\times10^6$ reliably detects CFOs and start of frames. We use this value as the threshold throughout the experiments in this section.

### 5.1.2 $L_{RLS}$

Large value of $L_{RLS}$ is required at the receiver when the interference is strong or there are many concurrently transmitted packets. But if $L_{RLS}$ is too large, the number of RLS-decodable packets within an overlapping set could decrease and computational complexity increases. We change

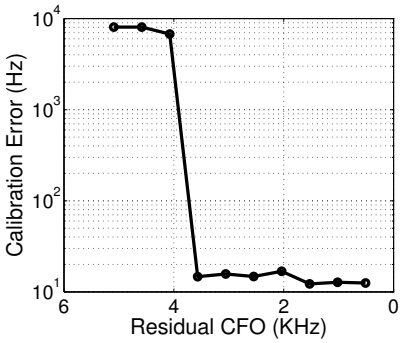


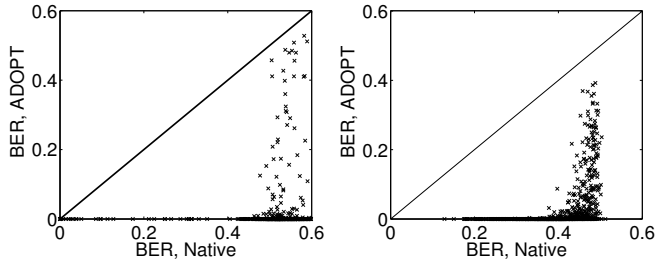

Fig. 13. ADOPT can reliably decode packet preambles with the coarse CFO estimates if the error is within 4KHz. Then it can reuse the preamble to fine-tune the CFO estimation.

the number of concurrent MICAz packets from 2 to 8, and vary $L_{RLS}$ value to measure the decoding performance. In this test, each packet has 15dB SNR at the receiver. Figure 12 shows that increase in the number of concurrent transmissions requires larger $L_{RLS}$ value to achieve the similar level of BER performance. With $L_{RLS}$ larger than 16, ADOPT decodes 2, 4 and 6 packets reliably. However when 8 packets overlap, the SINR becomes very low and even the large $L_{RLS}$ does not guarantee a sufficient BER performance for decoding packets. We use $L_{RLS}$ of 20 in the rest of our experiments.

### 5.1.3 CFO calibration

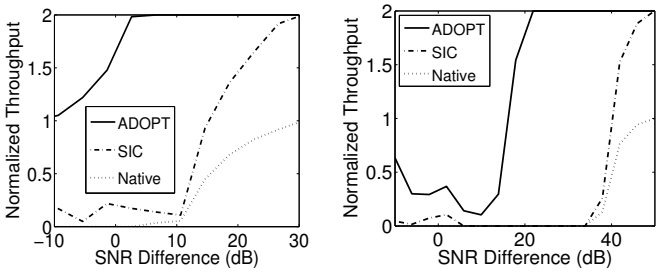
CFO estimation error (i.e., residual CFO) is caused because of limited number of signal samples used for FFT. CFO calibration process fine-tunes the coarse estimates of CFOs obtained via the frequency domain correlation. However if the estimation error is too large, there is a possibility that the following calibration process cannot sufficiently compensate for the error.

Here, we see how large residual CFO a receiver can tolerate during the CFO calibration process. First, we precisely measure the CFOs of every node during its single transmissions. Then 6 MICAz senders send packets to the ADOPT receiver. At the receiver, we manually adjust the amount of residual CFOs by multiplying arbitrary exponential terms to time domain symbols. Figure 13 shows the final CFO estimation




(a) ZigBee (b) 802.11b 11Mpbs

Fig. 14. BER of ADOPT and native APs. Two nodes transmit to one AP having two antennas.




(a) ZigBee (b) 802.11b 11Mpbs

Fig. 15. Throughput of ADOPT, SIC and native APs with two antennas as we vary the SNR difference of two colliding packets.

error after the calibration process, according to the residual CFO. With the residual CFO less than 4KHz, the estimation error is in the order of 10Hz. Note that the estimation error of FDCM cannot be larger than 4KHz, as the chip rate of CC2420 is 2Mcps and the number of chips within the preamble is 256 (i.e., $2M/256 = \pm 4K$). This means that the suggested CFO calibration can accurately estimate and compensate for the CFOs. The test results with WiFi shows larger tolerance to the residual CFOs because of its longer preamble size. We omit the result here.

#### 5.1.4 Bit error rates

We combine the above-mentioned parameters and briefly test the feasibility of ADOPT in the link level. Two nodes concurrently transmit back-to-back UDP packets and an ADOPT receiver decodes them using the signals received from two antennas. The native AP gets only an SNR gain (about 3 dB) from the two antennas. The signal strength of each packet is varied from -5 dB to 10 dB in the receiving SNR. Each data point in the Figure 14 represents the BER for each received frame. The straight line in the figures represents a region where the BER performance of ADOPT and the native AP becomes exactly the same. Thus, if the data points are located below the line, it tells that the BER performance of ADOPT is better. When the native AP experiences up to 40% BER, ADOPT experiences almost zero BER.

#### 5.1.5 Normalized throughput

Normalized throughput is defined as the average number of successfully decoded frames per collision. It does not take MAC layer overheads into account. We measure the normalized throughput of ADOPT and native APs when two nodes transmit to one AP with two antennas. Additionally we compare the performance of SIC [8]. The SIC depends upon the SNR difference among collided packets. If the received SNR of a packet is sufficiently larger than another, the receiver can decode the packet and cancel it out. Then it can decode a packet with weaker SNR.

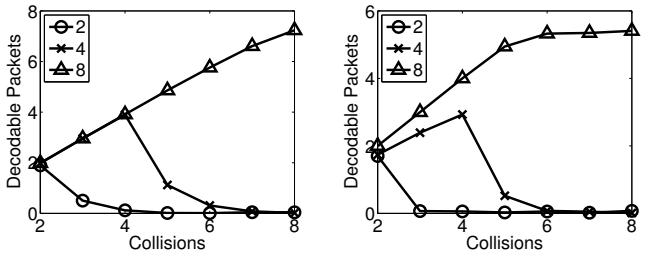
One node transmits at a fixed SNR of 15dB while the other node varies its power such that its received SNR ranges from -10dB to 65dB. The native AP has a SNR gain of 3 dB from the two antennas. With high SNR differences, SIC performs as well as ADOPT because of the capture effect – at this SNR regime, the native AP can recover the high SNR packet which also allows SIC to recover another packet with lower SNR as well. But in the other regime, SIC and native APs perform poorly. ADOPT performs well when the SNR difference is larger than 3 dB for ZigBee and 20 dB for WiFi. Below those levels, the performance of ADOPT is rather erratic, especially WiFi. That is because the transmission power of those packets is already very low.

#### 5.1.6 Performance scaling with more antennas

We evaluate the performance of ADOPT as we add more antennas. In the experiment, we use the deployed testbed where two to eight senders continuously transmit to the AP. We measure the average number of decoded packets by the ADOPT receiver for a given number of concurrent packets. We set all nodes to deliver similar receiving SNR to the AP. ZigBee nodes have approximately 10dB and 802.11b nodes have 15dB as SNR values. Figure 16 shows that the collided packets can be decoded with the highest probability when the number of collided packets matches the number of antennas. When it is larger than the number of antennas, the decoding performance drops quickly. With an eight-antenna ZeeBee ADOPT receiver, we can decoded up to 7.2 packets. Note that this high number of decoded packets is partly due to the ZigBee's reliable coding scheme at the physical layer (e.g., ZigBee uses 8 times redundancy for encoding a packet compared to 802.11b 11Mbps bit rate). However with the WiFi AP, that number stops at 5 packets. This is because WiFi nodes send at 11Mbps and their SNRs are not strong enough to push the number of decoded packets beyond 5.

### 5.2 Performance in Testbeds

We measure the performance of an eight antenna ADOPT receiver in the ZigBee and the WiFi testbeds. The performance of the native AP is also measured and compared. The native AP gets 8dB SNR gains from the antennas. We run the tests for 30 seconds and calculate the aggregate throughput. The aggregate throughput is calculated based on the total number of successfully decoded packets at the AP.




(a) ZigBee (b) 802.11b, 11Mbps

Fig. 16. The average number of decoded packets. X-axis shows the number of concurrently transmitted packets. Different lines show the number of antennas at the receiver. We vary the number of antennas from two to eight for a given number of collided packets.

| Scenario | CW size | Hidden terminal |
|---|---|---|
| 1 | Auto | No |
| 2 | Maximum 8 (16) | No |
| 3 | Auto | Yes |
| 4 | Maximum 8 (16) | Yes |

TABLE 1
Testbed evaluation scenarios.

| Scenario | Total tx. | Single tx. | Coll. | Concur. tx. |
|---|---|---|---|---|
| 1 | 16,061 | 12,620 | 3,441 | 1.25 |
| 2 | 16,145 | 9,099 | 7,046 | 1.62 |
| 3 | 10,729 | 5,106 | 5,623 | 1.90 |
| 4 | 21,370 | 1,903 | 19,476 | 3.78 |

TABLE 2
Packet transmission statistics in WiFi testbed during 30 seconds. The average number of collided packet increases with the scenario index.

Four different scenarios are tested as follow. In scenario 1, we do not change the medium access. In scenario 2, we manually adjust the maximum size of the contention window ($CW_{max}$) to 8 in the ZigBee testbed and 16 in the WiFi testbed such that the packets are transmitted more aggressively. We also differentiate the existence of hidden terminals. In scenario 3, we increase the carrier sense threshold of nodes such that half of the nodes are hidden to the another half. We do not modify contention window in scenario 3. Finally in scenario 4, we both modify the carrier sense threshold and size of the contention window such that the number of concurrent transmissions is maximized. The scenario setups are summarized in Table 1 and Table 2 presents the packet transmission statistics (obtained in the WiFi testbed). Each column shows the number of total packet transmission attempts, the number of single node transmissions, the number of collisions and the average number of concurrent transmissions per each transmission attempt.

Figure 17 shows the results. In scenario 1, where we do not modify any setting of standard protocols, carrier sensing prevents nodes to transmit concurrently. ADOPT still achieves about 30% to 46% improvement over the native AP. As we force the senders to transmit more aggressively in scenario 2, more packet collisions are introduced and the performance improvement by ADOPT becomes more significant. ADOPT shows 570% higher throughput in ZigBee testbed and 340% in WiFi testbed. Note that in both scenario 1 and 2, most of the packet preambles are perfectly aligned but ADOPT can reliably decode them.

In scenario 3 and 4 where hidden terminals exist, collisions are much more common. Therefore the performance of the native AP severely degrades. In scenario 3, size of each sender's contention window is around the maximum because of the repeated collisions. So the transmission attempts are largely discouraged. As we see from Table 2, nodes transmit the least number of packets. But the average number of collided packets per each transmission is still larger than scenario 1 and 2. By reducing the $CW_{max}$, we again push the senders to transmit more aggressively in scenario 4. Table 2 shows that the number of concurrent transmissions are doubled compared to scenario 3.

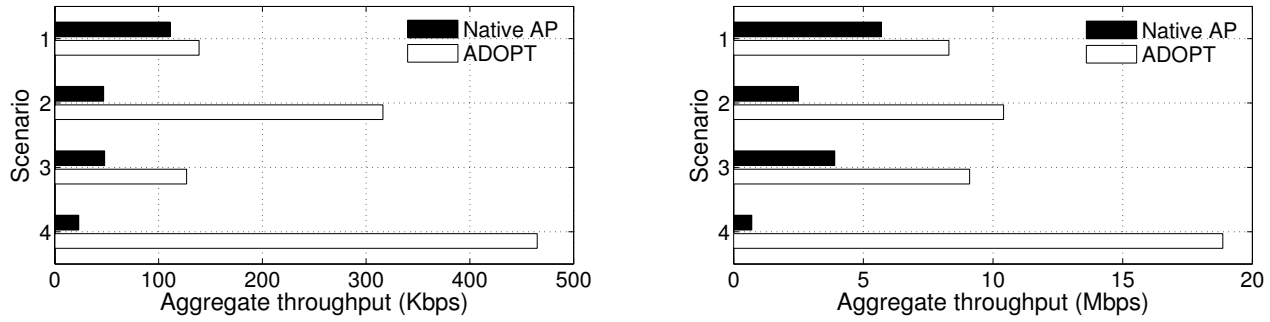
We can interpret scenario 1 and 3 as the general corporate networks. Scenario 2 and 4 are more close to the networks running MUC protocols where concurrent transmissions are largely encouraged. Without any explicit cooperation among the transmitters, ADOPT increases the throughput. It is because ADOPT can decode concurrent transmissions without restriction in their collision patterns. Comparing the throughput of the native AP in scenario 1 with the throughput of ADOPT in scenario 4, we can claim that ADOPT can achieve up to 319% in ZigBee and 231% in WiFi networks.

## 6 Related Work

SIC[8] exploits the capture effect arising in wireless networks to decode collided packets. When two or more packets collide, a receiver first decodes a packet with the strongest signals, subtracts them from the received signals. Then it decodes the weaker ones from the residue. This process is repeated to decode all the collided packets. SIC does not necessarily utilize multiple antennas at the receiver. The biggest limiting factor of SIC is its requirement of significant signal strength differences among overlapped signals. Sen *et al*.[14] argue that from the perspective of MAC design, the gain achievable by SIC is marginal because of the requirement.

ZigZag [6] was designed to solve the hidden-terminal problem. ZigZag exploits a repeated incidence of collision by the same set of packets which can happen through retransmission by hidden terminals. While ZigZag can significantly reduce MAC layer overhead, it requires $n$ retransmissions to decode $n$ packets.

IAC [3] enables MUC in a network of multiple APs even when the number of senders is larger than the number of receiving antennas at an AP. For this, multiple clients cooperate to align their signals using transmit beamforming so that their transmitted packets are detected as if coming from a single source. IAC can operate even when the number of antennas in each AP can differ from each other. $N^+$ [4] proceeds one




(a) ZigBee

(b) WiFi 802.11b, 11Mbps

Fig. 17. Aggregate Throughput of various MUC schemes. (a) no hidden terminal. (b) hidden terminals. The APs have 8 antennas. All nodes in the testbeds are transmitting.

step further and can serve as a general framework for networks where devices having different number of antennas interoperate. $N^+$ helps to fully utilize the channel by increasing the number of data streams when additional degrees of freedom are available. Additionally, $N^+$ uses an innovative technique, lightweight RTS/CTS, that can significantly reduce the channel training overhead. For both IAC and $N^+$, we believe that ADOPT can further help to improve the efficiency in obtaining the physical layer parameters.

SAM [5] implements MUC using interference nullifying, cancellation and media coordination techniques. SAM modifies CSMA clients to implement a channel access that ensures that the preamble transmissions of the collided packets are serialized so that any two or more preambles are not overlapped. While the proposed interference nullifying and cancellation is still effective, ADOPT can contribute to reduce the coordination overhead.

Aryafar *et al*. [22] implement a downlink MUC scheme using MIMO APs, which enables the AP to send multiple packets concurrently, each destined to a different receiver. It provides a practical implementation and an experimentation of zero forcing beamforming. ADOPT is rather an uplink solution while it solves downlink MUC.

Techniques have been suggested [15] that synchronize transmitters such that received signals at a receiver have similar physical layer characteristics (e.g., arrival time, carrier frequency offset or CSI). SourceSync [16] implements such an idea with FPGA hardware programming. This type of sender cooperation can significantly improve the quality of signals received at the receiver. Still, there could be diverse devices that do not support the cooperation and ADOPT can handle them

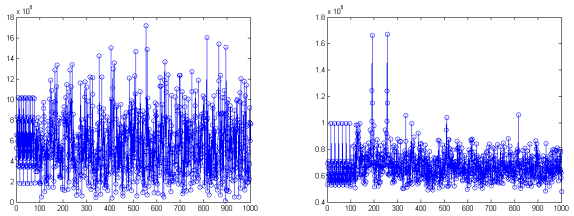
## 7 Conclusion and Discussion

Current WiFi or sensor network performance in a dense network suffer from severe packet collisions. Many MIMO based MUC solutions have been proposed to improve the capacity. Still, there is a limitation in the way those solutions obtain essential information for the MIMO decoding. The coordination is necessitated in order for the senders to transmit the packets in a way that the packet preambles are not overlapped with each other. The exclusive transmission not only lowers the channel utilization but reduces the interoperability. ADOPT is a practical solution that can work as an add-on, black box decoder that does not require such an explicit coordination between the senders. Our evaluation shows that ADOPT can achieve about 46% throughput improvement over the current WiFi and ZigBee standards. But when we are allowed to modify medium access and control the contention window sizes, which is easily achieved in the driver (Aethros and TinyOS allow such a control), we can achieve over 319% throughput improvement for ZigBee and 231% for WiFi 11Mbps. ADOPT can work with any kind of MAC schemes irrespective of their ability to coordinate packet transmissions. Further ADOPT could provide a level of flexibility in designing a new MAC protocol for the concurrent transmissions because it is free from such constraints.

There are further issues that we have not discussed so far, but need some attention.

**Rate adaptation**: A fundamental tradeoff exists between transmission rates and the number of concurrent transmissions. When multiple packets overlap, their decodability is limited by their transmission rates since a higher data rate is more susceptible to channel errors. So, which data rate should a client send when it can also control the number of concurrent transmissions (e.g., by adjusting its contention window sizes)? We can send at a high rate by reducing the concurrency, and also vice versa; as both data rate and concurrency affect throughput, this is a difficult decision. Developing an optimal rate adaptation algorithm for MUC is an area of future research.

**OFDM (IEEE 802.11a/g/n)**: We implemented ADOPT in ZigBee and 802.11b testbed which are single-carrier systems. But ADOPT can also be used with a multiple sub-carrier system like OFDM. We test FDCM using two collided IEEE 802.11a packets captured from SoRa radio boards [23]. FDCM works well with OFDM signals, without any modication from existing ADOPT code. Figure 18 is the example snapshots of



(a) Time domain cross correlation

(b) Frequency domain correlation and matching

Fig. 18. The FDCM technique works well with the OFDM signals (802.11a) packets.

the correlation results in time domain and frequency domain, respectively.

The steps to perform interference suppression and cancellation over OFDM signals (802.11a/g/n) is fundamentally similar. The only difference is that bits are modulated into frequency domain symbols.

## REFERENCES


[1] Y.-C. Cheng, J. Bellardo, P. Benkö, A. C. Snoeren, G. M. Voelker, and S. Savage, "Jigsaw: Solving the puzzle of enterprise 802.11 analysis," in *Proceedings of ACM SIGCOMM*, 2006.

[2] 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," 2009.

[3] S. Gollakota, S. D. Perli, and D. Katabi, "Interference alignment and cancellation," in *Proceedings of ACM SIGCOMM*, 2009.

[4] K. C.-J. Lin, S. Gollakota, and D. Katabi, "Random access heterogeneous mimo networks," in *Proceedings of ACM SIGCOMM*, 2011.

[5] K. Tan, H. Liu, J. Fang, W. Wang, J. Zhang, M. Chen, and G. M. Voellker, "SAM: Enabling practical spatial multiple access in wireless lan," in *Proceedings of ACM MOBICOM*, 2010.

[6] S. Gollakota and D. Katabi, "ZigZag decoding: Combating hidden terminals in wireless networks," in *Proceedings of ACM SIGCOMM*, 2008.

[7] K. A. Jamieson, "The softphy abstraction: From packets to symbols in wireless network design," in *Ph.D Dissertation, MIT*, 2008.

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

[9] S. Katti, S. Gollakota, and D. Katabi, "Embracing wireless interference: Analog network coding," in *Proceedings of ACM SIGCOMM*, 2007.

[10] Y. Zhao and S.-G. Haggman, "Sensitivity to doppler shift and carrier frequency errors in ofdm systems-the consequences and solutions," in *Vehicular Technology Conference, 1996. 'Mobile Technology for the Human Race'., IEEE 46th*, apr-1 may 1996, pp. 1564 –1568 vol.3.

[11] P. Moose, "A technique for orthogonal frequency division multiplexing frequency offset correction," *Communications, IEEE Transactions on*, vol. 42, no. 10, pp. 2908 –2914, oct 1994.

[12] "IEEE standard 802.11n," 2009.

[13] "IEEE standard 802.15.4," 2006.

[14] S. Sen, R. Roy Choudhury, and S. Nelakuditi, "CSMA/CN: Carrier sense multiple access with collision notification," in *Proceedings of ACM MOBICOM*, 2010.

[15] R. Mudumbai, D. R. Brown, U. Madhow, and H. V. Poor, "Distributed transmit beamforming: Challenges and recent progress," *IEEE Communications Magazine*, vol. 47, no. 2, pp. 102 – 110, 2009.

[16] H. Rahul, H. Hassanieh, and D. Katabi, "Sourcesync: A distributed wireless architecture for exploiting sender diversity," in *Proceedings of ACM SIGCOMM*, 2010.

[17] K. Tan, J. Fang, Y. Zhang, S. Chen, L. Shi, J. Zhang, and Y. Zhang, "Fine-grained channel access in wireless lan," in *Proceedings of ACM SIGCOMM*, 2010.

[18] S. Haykin, *Adaptive Filter Theory, 4th Ed.* Prentice Hall, 2001.

[19] R. Rao, S. Lang, and B. Daneshrad, "Overhead optimization in a mimo-ofdm testbed based on mmse mimo decoding," in *Proceedings of IEEE VTC*, 2004.

[20] I. Barhumi, G. Leus, and M. Moonen, "Optimal training design for mimo ofdm systems in mobile wireless channels," *IEEE Transaction on Signal Processing*, vol. 51, no. 6, pp. 1615–1624, June 2003.

[21] D. Tse and P. V. Fundamentals, "Fundamentals of wireless communication, cambridge university press, 2005." in *Cambridge University Press*, 2005.

[22] E. Aryafar, N. Anand, T. Salonidis, and E. W. Knightly, "Design and experimental evaluation of multi-user beamforming in wireless lans," in *Proceedings of ACM MOBICOM*, 2010.

[23] "Sora : Microsoft research software radio," http://research.microsoft.com/en-us/projects/sora/.