Interference Alignment and Cancellation

Shyamnath Gollakota, Samuel David Perli and Dina Katabi
MIT CSAIL

ABSTRACT

The throughput of existing MIMO LANs is limited by the number of antennas on the AP. This paper shows how to overcome this limitation. It presents interference alignment and cancellation (IAC), a new approach for decoding concurrent sender-receiver pairs in MIMO networks. IAC synthesizes two signal processing techniques, interference alignment and interference cancellation, showing that the combination applies to scenarios where neither interference alignment nor cancellation applies alone. We show analytically that IAC almost doubles the throughput of MIMO LANs. We also implement IAC in GNU-Radio, and experimentally demonstrate that for 2x2 MIMO LANs, IAC increases the average throughput by 1.5x on the downlink and 2x on the uplink.

Categories and Subject Descriptors C.2.2 [**Computer Systems Organization**]: Computer-Communications Networks

General Terms Algorithms, Design, Performance, Theory

Keywords Interference Alignment, Interference Cancellation

1 Introduction

Multi-input multi-output (MIMO) technology is emerging as the natural choice for future wireless LANs. The current design, however, merely replaces a single-antenna channel between a sender-receiver pair with a MIMO channel. The throughput of such a design is always limited by the number of antennas per access point (AP) [5, 29]. Intuitively, if each node has two antennas, the client can simultaneously transmit two packets to the AP. The AP receives a linear combination of the two transmitted packets, on each antenna, as shown in Fig. 1. Hence, the AP obtains two linear equations for two unknown packets, allowing it to decode. Transmitting more concurrent packets than the number of antennas on the AP simply increases interference and prevents decoding. Thus, today the throughput of all practical MIMO LANs is limited by the number of antennas per AP.

This paper introduces Interference Alignment and Cancellation (IAC), a practical scheme to overcome the antennas-per-AP throughput limit in MIMO LANs. IAC synthesizes two interference management techniques: interference alignment and interference cancellation, showing that the combination improves performance in scenarios where neither interference alignment nor cancellation applies alone.

To get a feel for how IAC works, consider again a 2-antenna client that uploads two concurrent packets to a 2-antenna AP. Say we have

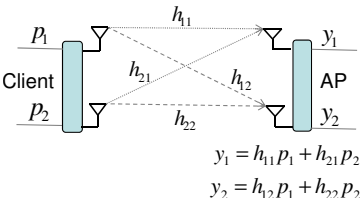


Figure 1: Throughput of current MIMO LANs is limited by the number of antennas per AP. The h_{ij}'s are known channel coefficients, and the p_i's are concurrent packets. The client transmits two concurrent packets. The AP receives a different linear combination of the transmitted packets on each antenna, which it solves to obtain the packets.

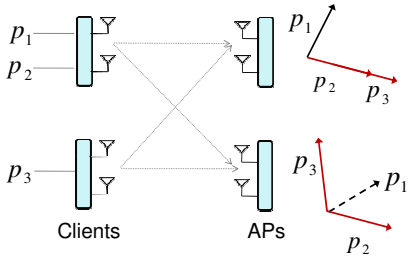


Figure 2: IAC Example. AP1 decodes packet p_1 and sends the decoded packet on the Ethernet to AP2 which then performs interference cancellation to subtract p_1. As a result AP2 can decode p_2 and p_3.

a second 2x2 client-AP pair on the same wireless channel and within interference range. Can the second client-AP pair concurrently upload a third packet? In existing MIMO LANs, the three concurrent packets interfere. As a result, each of the two APs gets two linear equations with three unknown packets, and hence cannot decode.

In contrast, IAC allows these three concurrent packets to be decoded. To do so, IAC exploits two properties of MIMO LANs: 1) MIMO transmitters can control the alignment of their signals at a receiver, and 2) APs are typically connected to a backend Ethernet, which they can use for coordination. Thus, in IAC, the two clients encode their transmissions in a special way to align the second and the third packets at AP1 but not at AP2, as shown in Fig. 2. As a result, AP1 can treat the second and third packets as one unknown; i.e., AP1 has the equivalent of two equations with two unknowns, allowing it to decode the first packet, p_1. AP1 then sends the decoded packet on the Ethernet to AP2, which can now perform interference cancellation to subtract the effect of the known packet. As a result, AP2 is left with two linear equations over two unknown packets, p_2 and p_3, which it can decode. The system delivers three packets per time unit. Hence, its throughput is not bounded by the number of antennas per AP.

Note the synergy between interference alignment and interference cancellation. Interference alignment aligns a subset of the packets at the first AP, allowing it to locally decode one packet and hence bootstrap the decoding process. Interference cancellation enables other APs to use the decoded packet to cancel its interference, and hence decode more packets. Neither interference alignment nor cancellation would be sufficient on its own to decode the three packets in Fig. 2.

SIGCOMM'09, August 17–21, 2009, Barcelona, Spain.

IAC has the following features:

- IAC brings in more gains than apparent in the above example and generalizes to any number of antennas. For a MIMO system with M antennas, we prove analytically that IAC delivers $2M$ concurrent packets on the uplink, and $\max(2M-2, \lfloor \frac{3}{2}M \rfloor)$ on the downlink – i.e., it doubles the throughput of the uplink, and almost doubles the throughput of the downlink for a large number of antennas.
- IAC delegates all coordination to the APs, which tell the clients how to encode their packets to produce the desirable alignment. Further, the channel estimates required for computing this alignment can be computed from ack packets with negligible overhead.
- IAC works with various modulations and FEC codes. This is because IAC subtracts interference before passing a signal to the rest of the PHY, which can use a standard 802.11 MIMO modulator/demodulator and FEC codes.

We have built a prototype of IAC in GNU-Radio and evaluated it using a testbed of 20 USRP nodes, each equipped with 2-antennas. Our results reveal the following findings:

- IAC improves the average throughput of our 20-node 2-antenna MIMO LAN by 1.52x on the downlink and 2.08x on the uplink. These experimental gains are slightly higher than the analytical ones because our analysis does not model IAC's diversity gains.
- IAC is fair in the sense that every client in our testbed benefits from using IAC instead of current MIMO.
- IAC provides a gain for any number of clients including a single active client. In this case, IAC exploits diversity to improve the throughput by 1.2x.

1.1 Contributions

This paper makes three main contributions:

- It presents interference alignment and cancellation (IAC), a new interference management technique that synthesizes interference alignment and interference cancellation, showing that the combination increases the throughput in scenarios where neither alignment nor cancellation applies separately.
- It analytically demonstrates that IAC almost doubles the multiplexing gain (i.e., number of concurrent transmissions) of flat-fading interference-limited MIMO LANs. The capacity of a distributed network can be written as [6]:

$$C(SNR) = d\log(SNR) + o(\log(SNR)),$$

  where d is the multiplexing gain and the capacity is computed as a function of the signal to noise ratio (SNR). At relatively high SNRs, the capacity is dominated by the first term and linearly increases with the multiplexing gain, d. We prove that IAC increases the multiplexing gain of flat-fading MIMO LANs, and thus provides a linear increase in the capacity characterization of these networks.
- It presents the first implementation of interference alignment demonstrating its feasibility. Our results show that in flat-fading channels, alignment can be performed without any synchronization even in the presence of different frequency offsets between concurrent transmitters.

2 Related Work

Related work falls in the following areas.

(a) MIMO Communication Theory. Our work builds on the theory of interference alignment. Recent work has argued that pre-processing signals at the senders in a manner that aligns interference at the receivers increases the total capacity of wireless networks [3, 6, 9, 21]. However, to the best of our knowledge, this paper is the first to present a system design and an implementation of interference alignment, showing that such idea works in practice. Further, this paper is the first to combine interference alignment with interference cancellation, showing that the combination, termed IAC, increases the throughput in scenarios where neither alignment nor cancellation helps alone.

Our work builds on recent advances in the theory of multiuser MIMO (MU-MIMO). MU-MIMO advocates having multiple clients concurrently communicate with a *single* AP or base station [11, 13, 29, 30]. Thus, the throughput of MU-MIMO is limited by the number of antennas on a single AP [13]. In contrast, this paper shows that IAC overcomes the antennas-per-AP throughput limit.

Our work is also related to Virtual MIMO [29, 20]. Virtual MIMO allows multiple transmitters to transmit concurrently and makes the receivers collaborate to jointly decode the concurrent transmissions. Virtual MIMO, however, remains a theoretical concept with no practical design because of two difficulties. First, it requires the transmitters to be synchronized to the symbol level. Second, it requires the receivers to communicate the raw received signal samples to be jointly decode. Communicating signal samples generates excessive overhead because to capture a signal without loss of information one needs to sample it at twice its bandwidth at each antenna, with each sample about 8-bit long. For example, to jointly decode three APs with four antennas each, one needs to send 6 Gb/s on the Ethernet. In contrast, IAC's receivers communicate decoded packets, and hence the Ethernet traffic remains comparable to the wireless throughput.

(b) Wireless Networks. Past work on single-antenna systems has proposed using multiple APs to improve coverage [26, 8], balance the load [22], or recover corrupted packets [24, 31]. This paper use multiple APs but focuses on MIMO networks, and introduces IAC, a new technique that enables MIMO LANs to support a larger number of concurrent transmissions than possible with existing designs.

Prior work has also advocated allowing concurrent transmissions in the context of single-antenna nodes. Some of these designs *prevent interference* by dividing the resources between users. For example, they might assign the different users different frequency bands [25, 26], or different codes [7, 17]. Other designs use interference cancellation to decode in the presence of interfering signals [14, 18]. IAC differs from this work in focus because it addresses MIMO networks. It also differs in mechanisms because IAC does not assign users different frequency bands or different codes and applies to scenarios where interference cancellation alone does not apply.

Finally, APs with directional antennas divide the space into sectors, each served by a different antenna. This prevents interference between nodes in different sectors, allowing multiple clients to communicate concurrently with the AP. Our approach is orthogonal to directional antennas since we can enable nodes in the same sector (i.e., nodes that interfere) to communicate at the same time.[1]

3 Interference Alignment and Cancellation

IAC's design targets MIMO wireless LANs in a university or corporate campus where APs are connected via a wired infrastructure (e.g., Ethernet). Today these networks use one AP to serve any particular area, and limit interference by assigning adjacent APs to different

[1] It is a common mistake to think that MIMO beam-forming is equivalent to directional antennas. Beam-forming allows the signal to constructively combine at the intended receiver, increasing its throughput. This however still creates interference at nodes that are not in the direction of the intended receiver. Hence, beam-forming cannot overcome the antennas-per-node throughput limit of MIMO LANs.

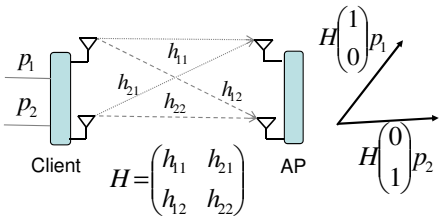


Figure 3: Two Packets on Uplink. The client transmits two packets, p_1 and p_2, from its two antennas. The packets arrive along the vectors $H[1\ 0]^T$ and $H[0\ 1]^T$, where H is the channel matrix and $[.]^T$ refers to the transpose of a vector. To decode p_1 and p_2, the AP projects along the vectors orthogonal to $H[0\ 1]^T$ and $H[1\ 0]^T$ respectively.

802.11 channels. Similar to the current architecture, in IAC, adjacent areas employ different 802.11 channels, but in contrast to the current architecture, each of these areas is served by a set of APs on the same channel, rather than a single AP. IAC allows this set of APs to serve multiple clients at the same time despite interference. To do so, it leverages the wired bandwidth to enable the APs to collaborate on resolving interfering transmissions.

IAC has three components: 1) a physical layer that decodes concurrent packets across APs, 2) a MAC protocol that coordinates the senders to transmit concurrently on the wireless medium, and 3) an efficient mechanism to estimate channel parameters.

4 IAC's Physical Layer

IAC modifies the physical layer to allow multiple client-AP pairs to communicate concurrently on an 802.11 channel. IAC operates below existing modulation and coding and is transparent to both.

For clarity, we present our ideas in the context of a 2-antenna per-node system, and assume nodes know the channel estimates. Later, we extend these ideas to any number of antennas and explain how we measure channel functions. Our presentation focuses on scenarios where interference from concurrent transmissions is much stronger than noise and is the main factor affecting reception.

(a) Two concurrent packets on the uplink: Let us start with the standard MIMO example in Fig. 3, where a single client transmits two concurrent packets to an AP. Say that the client transmits p_1 on the first antenna, and p_2 on the second antenna. The channel linearly combines the two packets (i.e., it linearly combines every two digital samples of the packets). Hence, the 2-antenna AP receives the following signals:

$$y_1 = h_{11}p_1 + h_{21}p_2$$
$$y_2 = h_{12}p_1 + h_{22}p_2,$$

where h_{ij} is a complex number whose magnitude and angle refer to the attenuation and the delay along the path from the i^{th} antenna on the client to the j^{th} antenna on the AP, as shown in Fig. 3.

Since the nodes have two antennas, the transmitted and received signals live in a 2-dimensional space. Thus, it is convenient to use 2-dimensional vectors to represent the system [29]. This representation will allow us to use simple figures to describe how a MIMO system works. We can re-write the above equations as:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = H\begin{pmatrix}1\\0\end{pmatrix}p_1 + H\begin{pmatrix}0\\1\end{pmatrix}p_2, \quad (1)$$

where H is the 2×2 uplink channel matrix (i.e., the matrix of h_{ij}'s). Thus, the AP receives the sum of two vectors which are along the directions $H[1\ 0]^T$ and $H[0\ 1]^T$ (where $[.]^T$ refers to the transpose of a vector), as shown in Fig. 3.

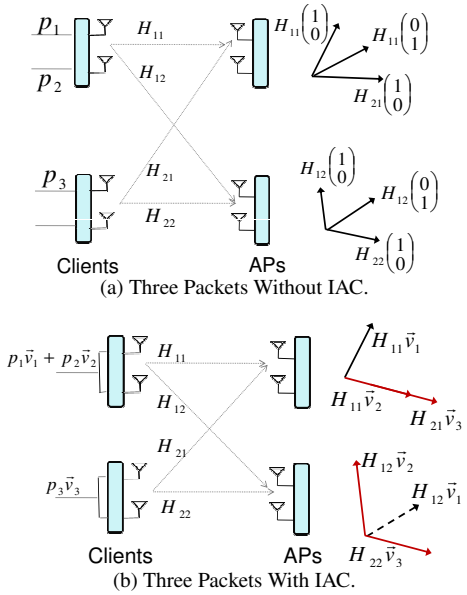


Figure 4: Three Packets with/without IAC. In (a), the clients transmit the packets without alignment. The packets combine at the APs along three different vectors and the APs cannot decode any packet. The second case shows how IAC delivers three packets on the uplink. Specifically, two of the three packets are aligned at AP1, allowing AP1 to decode one packet and send it to AP2 on the Ethernet. AP2 uses interference cancellation to subtract the packet and decode the remaining two packets.

Assume the AP knows the channel matrix, H, (we will see how to estimate it in §8). Decoding is easy; to decode p_1, the AP needs to get rid of the interference from p_2, by projecting on a vector orthogonal to $H[0\ 1]^T$. To decode p_2 it projects on a vector orthogonal to $H[1\ 0]^T$. We refer to the direction that a receiver projects on, to decode, as the **decoding vector**.

(b) Three concurrent packets on the uplink: Consider what happens if another client concurrently transmits a packet, as shown in Fig. 4a. Using the same derivation as above, AP1 receives:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = H_{11}\begin{pmatrix}1\\0\end{pmatrix}p_1 + H_{11}\begin{pmatrix}0\\1\end{pmatrix}p_2 + H_{21}\begin{pmatrix}1\\0\end{pmatrix}p_3,$$

where H_{11} and H_{21} are channel matrices from the first and second clients to AP1. Said differently, AP1 receives the combination of three packets p_1, p_2, and p_3, along three vectors $H_{11}[1\ 0]^T$, $H_{11}[0\ 1]^T$ and $H_{21}[1\ 0]^T$, as shown in Fig. 4a. Since AP1 has only two antennas, the received signal lives in a 2-dimensional space; hence AP1 cannot decode three packets. Said differently, for any packet p_i, the AP cannot find a projection (decoding vector) that eliminates interference caused by the other two packets. The second access point, AP2, is in a similar state, it receives three packets along three vectors $H_{12}[1\ 0]^T$, $H_{12}[0\ 1]^T$ and $H_{22}[1\ 0]^T$, and cannot decode for the same reason.

However, one advantage of MIMO is that a transmitter can control the vectors along which its signal is received. For example, when a transmitter transmits packet p_1 on the first antenna, this is equivalent to multiplying the samples in the packet by the unit vector $[1\ 0]^T$ before transmission. As a result the received vector at the AP is $H[1\ 0]^T p_1$, where H is the channel matrix from transmitter to receiver. If the transmitter, instead, multiplies the packet p_1 by a different vector, e.g., $\vec{v}$, the AP will receive the vector $H\vec{v}p_1$. Thus, instead of transmitting each packet on a single antenna, we multiply packet p_i by a vector $\vec{v}_i$ (i.e., multiply all digital samples in the packet by the vector) and transmit the two elements of the resulting 2-dimensional

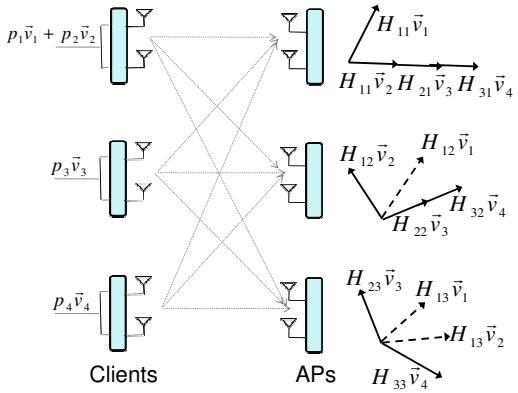


Figure 5: Four Packets on the Uplink. IAC allows AP1 and AP2 to decode one packet each, and AP3 to decode the two remaining packets. This requires three packets to be aligned at AP1 and two packets at AP2, which can be done by picking appropriate encoding vectors.

vector, one on each antenna. Thus, by changing $\vec{v}_i$, we can control the vector along which the AP receives the packet. We call the vector $\vec{v}_i$ the **encoding vector** of packet i.

Now, we can apply this method to the 2-client and 2-AP system to transmit three concurrent packets. In particular, the transmitters multiply packet i with vector $\vec{v}_i$, as shown in Fig. 4b. We want to pick $\vec{v}_2$ and $\vec{v}_3$ such that the second and third packets (i.e., p_2 and p_3) are aligned at AP1, as in Fig. 4b, that is:[2]

$$H_{11}\vec{v}_2 = H_{21}\vec{v}_3, \tag{2}$$

where H_{11} and H_{21} are the channel matrices from the first and second clients to AP1. This can be easily done by picking random (but unequal) values for $\vec{v}_1$ and $\vec{v}_2$ and substituting in the above equation to get $\vec{v}_3$ (i.e., $\vec{v}_3 = H_{21}^{-1}H_{11}\vec{v}_2$).[3]

In this case, AP1 receives the second and third packets aligned on the same direction as in Fig 4b. Thus, AP1 can decode the first packet, p_1, by projecting on a vector orthogonal to the aligned interference, i.e., a vector orthogonal to $H_{11}\vec{v}_2$ and $H_{21}\vec{v}_3$. Since these two vectors are already aligned, there is a vector that is orthogonal to both of them, and thus the AP can decode. Note that without alignment, AP1 could not decode because $H_{11}\vec{v}_2$ and $H_{21}\vec{v}_3$ would have different directions, and no vector will be orthogonal to both.

Note that aligning two vectors with respect to AP1 does not mean that they are aligned with respect to AP2. This is because the channels from the clients to the two APs are different and independent. However, we do not need to align the signals at AP2. AP1 can decode the first packet and send it to AP2 on the Ethernet. Now AP2 knows the first packet. It also knows the channel functions (see in §8 how we compute channel functions). Hence it can reconstruct the signal associated with the first packet and subtract it from what it received. This is standard interference cancellation [19, 29]. After cancellation, AP2 is back into a scenario similar to typical MIMO, namely two packets on two different directions, in a 2-dimensional space. Hence, it can decode. Thus, we obtained all three packets. AP1 decoded the first packet, and AP2 decoded the second and third.

(c) Four concurrent packets on the uplink: Let us try to increase the number of concurrent packets on the uplink to 4. We cannot do this with only 2 clients and 2 APs (This is because the system is already too constrained to produce the desirable alignment.) We need to add an additional AP-client pair. For example, consider the three APs and three clients, in Fig. 5. The first client transmits packets p_1 and p_2, the second client transmits p_3 and the third client transmits the fourth packet, p_4. Now that we have developed a vector representation, it is fairly simple to produce an IAC solution for any configuration. Specifically, as shown in Fig. 5, AP1 needs to align 3 out of 4 packets. This results in one free packet, e.g., p_1, which can be decoded with orthogonal projection, as we did earlier. From the perspective of AP2, p_1 is already decoded at AP1, and hence can be subtracted and removed from the signal. Thus, AP2 is left with three unknown packets. To decode one more packet, it needs to have 2 out of 3 packets aligned, as shown in Fig. 5. From the perspective of AP3, two packets are already decoded at AP1 and AP2, and their signal can be canceled using interference cancellation. Thus, AP3 is left with only two unknown packets, which it can decode. Hence, AP3 does not need to align any packets. We can achieve the desired alignment (i.e., the alignment in Fig. 5) by solving the following equations:

$$H_{11}\vec{v}_2 = H_{21}\vec{v}_3 = H_{31}\vec{v}_4 \tag{3}$$

$$H_{22}\vec{v}_3 = H_{32}\vec{v}_4, \tag{4}$$

where H_{ij} is the uplink channel matrix from the i^{th} client to the j^{th} AP. Eqs. 3 ensures the desired alignment at AP1 and Eq. 4 ensures the desired alignment at AP2. Effectively, this translates to three linear equations in three unknowns (the vectors), which can be solved. Thus, the APs can decode four concurrent packets.[4]

(d) The downlink: The discussion so far has focused on the uplink, what about the downlink? Clearly the downlink is more limited, since the clients cannot cooperate over a wired Ethernet. A client cannot decode one packet and send it to other clients for interference cancellation. The lack of cooperation means that the clients have to decode independently. So, we need to align the interference at each client to ensure that it can decode at least one packet. For a 2-antenna system, this means that we can at best deliver 3 concurrent packets on the downlink. This however is still higher than what can be delivered in today's point-to-point MIMO LANs.

Say that we want to deliver packets p_1, p_2, and p_3 to Client 1, Client 2, and Client 3 respectively. Each client needs to receive the two undesired packets aligned along the same vector and the desired packet along a different vector, as shown in Fig. 6. To achieve this behavior, each AP transmits one of the three packets. Now the roles are flipped: the APs are the transmitters and the clients the receivers. Hence, each AP multiplies the transmitted packet by a vector $\vec{v}_i$ that is carefully chosen to ensure the desired alignment. Specifically, we need to ensure:

$$H_{21}^d\vec{v}_2 = H_{31}^d\vec{v}_3 \tag{5}$$

$$H_{12}^d\vec{v}_1 = H_{32}^d\vec{v}_3 \tag{6}$$

$$H_{13}^d\vec{v}_1 = H_{23}^d\vec{v}_2, \tag{7}$$

where H_{ij}^d is the channel from the i^{th} AP to the j^{th} client, i.e., the downlink channels. The three equations above align the packets at each client to ensure that the two undesired packets are along the same vector. These are three linear equations over three unknown vector and can be solved using standard methods (similar to how we solved Eqs. §3 and §4). Hence, each client can decode its desired packet by orthogonal projection.

[2] In general, aligning the directions would mean $H_{11}\vec{v}_2 = \alpha H_{21}\vec{v}_3$, where α is a scalar. Also note that the vectors are normalized to satisfy the power constraints. But for clarity, we ignore these details in our description.

[3] Channel matrices are typically invertible because the antennas are chosen to be more than half a wavelength apart. If the matrix is not invertible, then you don't really have a MIMO system because the two antennas translate into just one equation.

[4] The solution to the alignment is $\vec{v}_4 = eig(H_{32}^{-1}H_{22}H_{21}^{-1}H_{31})$, where $eig(H)$ is an eigen vector of H, and $\vec{v}_2 = H_{11}^{-1}H_{31}\vec{v}_4$ and $\vec{v}_3 = H_{21}^{-1}H_{31}\vec{v}_4$.

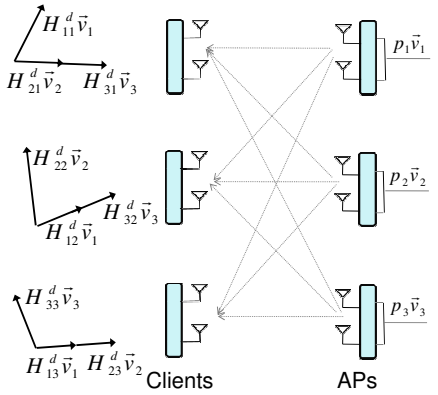


Figure 6: Three Packets on the Downlink. The APs deliver one packet to each client. To enable the client to decode its packet, all the undesired packets at the client must be aligned.

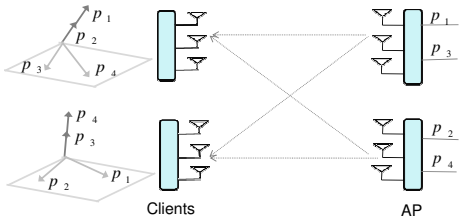


Figure 7: Four Packets on Downlink. At the first client, packets p_3 and p_4 are aligned along one dimension, allowing p_1 and p_2 to lie in a two dimensional space and hence be decoded. Similarly, at the second client, packets p_1 and p_2 are aligned, allowing p_3 and p_4 to be decoded.

5 Beyond Two Antennas

The previous section focuses on 2-antenna systems, but for the general case of M antennas per-node, what is the maximum number of concurrent packets that can be delivered? Further, how many APs are needed to support such a system?

Naively, it might seem that the number of concurrent packets is constrained only by the number of APs. Specifically, it might seem that one can align the received packets at every AP, allowing each of them to decode at least one packet, and hence one can keep increasing the number of concurrent packets by increasing the number of APs. This is however misleading because aligning a signal at one receiver limits the ability of the transmitter to freely align it at a second receiver. In particular, every alignment imposes new constraints on the encoding vectors at the transmitter. For a feasible solution, the constraints should stay fewer than the free variables in an encoding vector. Since the encoding vector has as many variables as there are antennas on the node, the number of constraints cannot exceed the number of antennas. Thus, using more APs is beneficial but only up to a point, after which one needs to increase the number of antennas. Below, we demonstrate that in IAC, the number of concurrent packets can be almost twice the number of antennas, and that this gain is achieved with a relatively small number of APs.

(a) Downlink. In [15], we prove the following:

Lemma 5.1 *In a system with M antennas per node, the maximum number of concurrent packets IAC can deliver on the downlink is* $\max\{2M-2, \lfloor \frac{3}{2}M \rfloor\}$. *For* $M > 2$, *IAC achieves this with* $M-1$ *APs.*

For $M = 3$, the above lemma tells us that we can achieve 4 concurrent packets on the downlink. Fig. 7 shows the downlink case. We have two APs and two clients. Each AP transmits two packets, one for each client. Since the clients have three antennas, the signal is in a three dimensional space. Thus, if we align two packets along one dimension, the other two packets are free of interference and can be decoded, resulting in 4 concurrent packets.

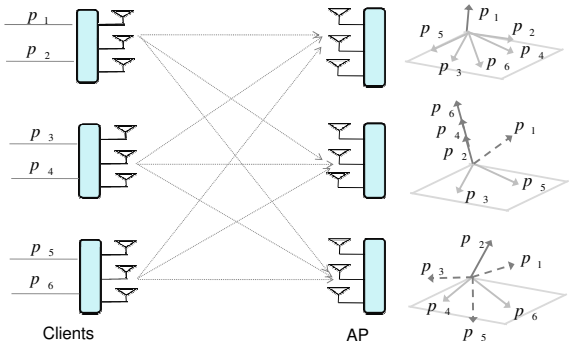


Figure 8: Six Packets on Uplink. At AP1, all the packets other than p_1 are aligned on a two dimensional plane, allowing p_1 to be decoded. At AP2, p_2, p_4 and p_6 are aligned along, allowing p_3 and p_5 to be decoded. At AP3, we cancel p_1, p_3 and p_5 leaving p_2, p_4 and p_6 to lie along three different dimensions and be decoded.

The above procedure can be generalized to any number of antennas. Specifically, if we have $M-1$ APs and two clients, a procedure that makes each AP transmit a packet to each client can deliver a total of $2M-2$ concurrent packets across the two clients. For a large M, this almost doubles the throughput of current MIMO LANs.

(b) Uplink. In [15], we prove the following:

Lemma 5.2 *For a M-antenna system, three or more APs, and at least two clients, IAC can deliver* $2M$ *concurrent packets on the uplink.*

For $M = 3$, the above lemma tells us that we can achieve 6 concurrent packets on the uplink. Fig. 8 shows three clients transmitting to three APs. At the first AP, five out of six packets are aligned in the same plane. This leaves one packet free of interference and hence can be decoded. From the perspective of the second AP, one packet is already decoded and hence can be eliminated from the received signal. Out of the five packets left, the second AP needs to have three packets aligned along one dimension and two free packets, allowing it to decode two packets. Finally, from the perspective of the last AP, three packets are already decoded and hence their interference can be eliminated. This leaves the last AP with three unknown packets in a three dimensional system and hence it can decode all of them.

Again, this procedure can be applied independent of the number of antennas. Specifically, one needs to align $2M$ packets such that the first AP can decode one packet, the second AP decodes $M-1$ packets and the last AP decodes M packets.

6 Practical Issues

The practicality of IAC relies on being able to implement interference alignment and interference cancellation. IAC uses only the subtraction step of interference cancellation. Interference cancellation typically involves two steps: first it decodes one of the concurrent packets in the presence of interference and second it subtracts the decoded packet from the rest to remove its contribution to interference, allowing the decoding of more packets. IAC replaces the first step with interference alignment to orthogonalize interference and eliminate its impact as it decodes one of the concurrent packets. It uses interference cancellation only to subtract the decoded packet. The subtraction step of interference cancellation is widely studied and has been shown to work in practical implementations [4, 14, 18].

Furthermore, the subtraction step does not require any synchronization between transmitters,[5] works with OFDM systems and various modulation schemes, and can accommodate single tap and multi-tap frequency selective channels [19, 10].

In contrast, prior to this paper, interference alignment has been a purely theoretical idea with no practical implementation. Thus, in this section, we focus on the practicality of performing alignment.

(a) Frequency offset: In practice, a transmitter-receiver pair always exhibits a small frequency offset, Δf. The frequency offset causes the phase of the received signal to increase linearly with time, i.e., the received vector rotates with time. Since the frequency offset is typically different for different sender-receiver pairs, signals from different transmitters that are aligned at the same receiver will rotate at different rates. Thus, it might seem that signals that are aligned at the beginning of a packet will lose alignment with time and be completely misaligned by the end of the packet. This reasoning however is incorrect because **interference alignment happens in the antenna-spatial domain and not the I-Q domain.**[6] Differences in frequency offset cause relative differences in how the signals rotate in the I-Q domain but only scale the direction of the vectors in the spatial domain by a complex number, leaving the alignment unaffected. Specifically, suppose the encoding vectors, $\vec{v_1}$ and $\vec{v_2}$, are picked to satisfy the equation $H_{11}\vec{v_1} = H_{21}\vec{v_2}$. As a result of the two frequency offsets, Δf_1 and Δf_2, the channel, $H_{i1}(t)$, changes as a function of time as $H_{i1}e^{j2\pi\Delta f_i t}$. Thus, these time varying channels satisfy the equation:

$$H_{11}(t)e^{-j2\pi\Delta f_1 t}\vec{v_1} = H_{21}(t)e^{-j2\pi\Delta f_2 t}\vec{v_2}$$
$$H_{11}(t)\vec{v_1} = e^{j2\pi(\Delta f_1 - \Delta f_2)t}H_{21}(t)\vec{v_2}$$

The complex function $e^{j2\pi(\Delta f_1 - \Delta f_2)t}$ scales the vector, $H_{21}(t)\vec{v_2}$, leaving its orientation unaffected. Since alignment only requires that the two vectors have the same orientation, the signals remain aligned through the end of the packets despite different frequency offsets. Realizing that signal alignment is unaffected by rotation in the I-Q domain is an important lesson that we learned from the implementation.

(b) Different Modulations: Interference alignment works independent of what constitutes the signal, i.e., independent of the modulation scheme (BPSK, QAM, or OFDM). It might seem that the modulation scheme, say QAM, changes the signal orientation and hence breaks the alignment. Again this argument is incorrect because modulation changes the signal's orientation in the I-Q domain, but interference alignment happens in the antenna spatial domain.

(c) Symbol Synchronization: One lesson that we learned from the implementation is that for relatively flat channels, you do not need to have symbol level synchronization. Specifically, if the channel between each transmit-receive antenna pair can be represented by a single complex number, h_{ij}, whose magnitude refers to the attenuation and phase refers to the delay along the path, interference alignment can then be implemented accurately without transmitter synchronization. This arises from two facts: 1) we perform interference alignment at the signal level and not symbol level, i.e., we align signal samples regardless of what symbol they represent, 2) the alignment occurs in the spatial antenna domain, not the I-Q domain, and hence though unsynchronized transmitters may not be aligned in the I-Q domain, this does not affect their alignment in the spatial antenna domain.[7]

[5] Once the receiver knows the bits and estimates the channel function from the preamble, it can reconstruct the corresponding continuous signal, sample it at the desired points, and subtract it from its received version.

[6] The I-Q domain is the 2-dimensional space that refers to the transmitted complex number.

[7] It should be noted that interference alignment is different from multi-user MIMO (which typically requires synchronization) in that not all signals need be decodable at a receiver. Specifically aligned interferers need not be decodable.

Note that modeling the channel between a pair of antennas as a single complex number is accurate for narrowband or flat channels, but becomes less so as the width of the channel increases. We conjecture that even if the channel is not quite flat, one can still do the alignment separately in each OFDM subcarrier without trying to synchronize the transmitters. In this case, there is some interference between the OFDM subcarriers, but given that nearby subcarriers typically have similar frequency response, for moderate width channels the resulting imperfection in the alignment stays acceptable. We cannot check this conjecture on USRP1 since their channel is fairly narrow and is accurately modeled with a single complex number.

7 Medium Access Control

Since IAC allows multiple clients and APs to transmit simultaneously, it changes the requirements of the MAC. The challenge in designing a MAC protocol for IAC arises not only from the need to enable multiple nodes to concurrently access the medium, but also from our desire to maintain minimal complexity at the clients. Specifically, a client should be oblivious to the number of APs in the system, and other clients who transmit concurrently. Finally, since traffic is bursty, we need to dynamically change the combination of concurrent clients to match instantaneous traffic demands, while respecting fairness.

The basic principle underlying our solution is to move complexity to APs, which arbitrate the medium among clients, and also provide each client with its encoding and decoding vectors. Our solution has two components: 1) a MAC protocol that allows multiple nodes to access the medium concurrently, and 2) a concurrency algorithm that decides which clients upload/download concurrently.

7.1 Accessing the Medium

Our design extends the 802.11 Point of Coordination Function (PCF) mode to allow it to support multiple concurrent senders. PCF is part of the standard [12]. It allows the AP to arbitrate the medium by polling the clients, and is originally designed to enable 802.11 networks to deal with time sensitive information.

(a) Contention-Free and Contention Periods. In IAC, one of the APs is designated as the leader. The leader AP acts as a coordinator. It polls the clients and grants access to those who have data to transmit [12]. Similar to PCF, we divide time into: Contention Free Period (CFP) and Contention Period (CP), as shown in Fig. 9. A contention-free period starts with the leader AP broadcasting a beacon that announces the duration of the current CFP. During a CFP, the leader AP coordinates access to the medium enabling the nodes to transmit using IAC. This is followed by a contention period, during which any node can contend for the channel using standard 802.11n.

The objective of this design is to use the contention period to allow new clients to associate with the APs, or to transmit after a long period of silence, using point-to-point MIMO. In contrast, the contention-free period (CFP) is used to pack transmissions as much as possible, increasing throughput. The duration of the contention period (CP) is constant, while the duration of CFP varies depending on congestion. During CFP, the APs serve one packet (on uplink and downlink) to each client that has pending traffic. Hence, when congestion is low and queues are empty, the CFP naturally shrinks, and clients spend more time in CP. When congestion is high, many clients have pending traffic and hence the CFP expands, which is desirable as this mode uses IAC to pack transmissions and increase efficiency.

(b) Acquiring Medium During CFP. Next, we explain how concurrent transmitters acquire the medium during a CFP. Clearly, this

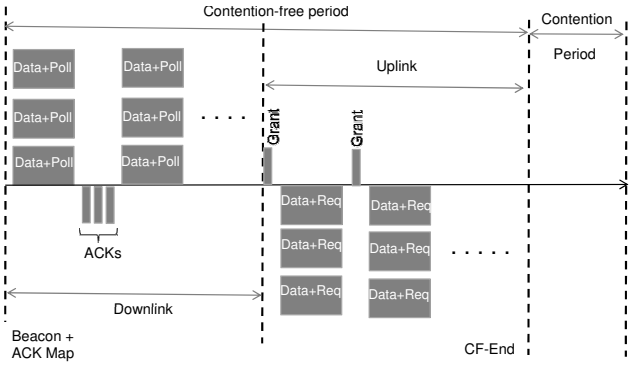


Figure 9: IAC's Extension to PCF.

Figure 10: The metadata in a `DATA+Poll` frame. This metadata is broadcast by the leader AP alone to inform the clients in a downlink transmission group of their decoding vectors and the other APs of their encoding vectors.

requires knowing which clients are served concurrently. This is the job of the concurrency algorithm, which divides clients with pending traffic into groups of concurrent transmissions that we call **transmission groups**. It further decides which AP serves which client in a transmission group, and the values of the encoding and decoding vectors. The process for deciding this is described in §7.2. In this section, we focus on how to deliver packets in each transmission group.

Fig. 9 shows the series of events during a contention-free period. At the beginning of a CFP, the leader AP sends a beacon. The leader AP then steps through the downlink transmission groups, one at a time, transmits their downlink packets with the help of other APs, and polls the corresponding clients for uplink traffic. This mode is similar to current PCF behavior, except that in the current PCF the AP steps through a list of individual clients, one at a time; whereas in IAC the leader AP steps through a list of transmission groups.

(b.1) Downlink. The leader AP first goes through the list of downlink transmission groups. With the help of other APs, it sends a `DATA+Poll` frame to each group. This frame has two parts. The first part, shown in Fig. 10, is broadcast by the leader AP alone, and contains the ids of the clients in the group and their encoding and decoding vectors. The ids are given to the clients upon association. The encoding and decoding vectors are computed by the concurrency algorithm which runs on the leader AP. The leader AP also includes a frame id, `Fid`, the number of APs and a checksum of its broadcast. Further, it sets the length of the `DATA+Poll` frame to the maximum length of the packets in the transmission group, so that all clients know when the frame ends. The second part of the frame is the combination of concurrent transmissions by all APs. For the example of three APs with 2-antennas each, this part has the three APs transmitting a packet to each of the three clients in a transmission group.

Note that both the clients and APs listen to the leader AP as it broadcasts the first part of the `DATA+Poll` frame. In order to transmit concurrently, the APs need to learn their encoding vectors. Similarly, the clients need to learn the decoding vectors to be able to decode their data. The clients and APs can use the checksum to test whether they received the correct information. Note that the transmissions still work fine if any of the APs or the clients failed to hear the leader AP. Specifically, the AP/client who failed to hear the leader AP, will not transmit. The other transmissions can go as desired.

After the `DATA+Poll` frame, the clients in the transmission group send their acks, one after the other, using traditional MIMO. The order in which they transmit these acks is the same as the order of their ids in the `DATA+Poll` frame. These acks are similar to synchronous 802.11 acks. In 802.11, they are sent one after each data packet. Here the data packets are sent concurrently and all acks follow.

(b.2) Uplink. After going through all downlink groups, the leader AP steps through the uplink groups. Similar to the downlink case, the leader AP first broadcasts a `Grant` frame specifying the ids of the clients that will transmit on the uplink, and the encoding and decoding vectors. The other APs listen to the encoding and decoding vectors and wait for clients' transmissions. The clients in an uplink group use their encoding vectors to transmit simultaneously on the uplink. Each client transmits a `Data+Req` frame. This frame contains the client's uplink data. If the client still has traffic to send, the frame will also contain a new request for transmission. Each AP listens to the `Data+Req` frame and projects the received signal on the proper decoding vector. This projection is orthogonal to the interfering signals and hence it allows each AP to receive its client of interest.

One difference between the uplink and downlink is that, while each client on the downlink can immediately ack its packet, the APs need to decode successively using interference cancellation and hence cannot send synchronous acks. The solution however is simple. During the following contention period, the APs inform the leader AP of successful receptions using Ethernet. The leader AP combines and sends all acks at the beginning of the next CFP, by embedding them in the beacon information as a bit map. This should not cause any significant delay since it allows all clients in the CFP mode to learn about their previous packet before they get to send the next packet.

At the end of CFP, the leader AP sends a `CF-End` frame. This allows the clients to go back to the contention mode, where they use traditional point-to-point MIMO. A few points are worth noting.

(a) How do we deal with lost packets and retransmissions? If a packet is lost on the uplink, the client discovers the loss from the lack of an ack (at the beginning of the next CFP) and asks for a new transmission slot next time it is polled. On the downlink, the corresponding AP discovers the packet loss immediately, from the lack of a client ack, and asks the leader AP to schedule a retransmission.

(b) Is it possible for various APs to make inconsistent decisions? Only the leader AP makes decisions, while other APs are dumb transmitters/receivers. Similar to clients, they receive their encoding and decoding vectors for each transmission group over the medium and use them without any modification. They only inform the leader AP in case a packet is lost, or the channel's estimate has changed.

(c) How often do APs need to communicate over the Ethernet and what do they exchange? As described in §4, APs exchange the decoded packets over the Ethernet to perform interference cancellation. Further, the subordinate APs need to tell the leader AP whenever a packet is lost or channel coefficients to a client changes by more than a threshold value. The APs can send this information as annotation on packets they exchange to perform cancellation.

(d) How large is the Ethernet overhead? To minimize Ethernet overhead, IAC connects the set of APs using a hub. This design ensures that every decoded packet is broadcast only once to all APs and to the switch that forwards the packet to its wired/final destination. In this design every packet is transmitted once and there is no extra overhead. While a hub is less efficient for a general Ethernet than a switch, it is a natural choice to connect the IAC APs. This hubbed network is then connected to the rest of the Ethernet via a switch.

(e) How large is the wireless overhead associated with IAC's MAC? IAC introduces metadata to coordinate clients and APs.

Specifically, concurrent transmissions are preceded by a short broadcast from the leader AP to inform the client-AP pairs of their encoding and decoding vectors. Such a broadcast message already exists in 802.11 PCF mode. [8] We only annotate these messages with extra information that is a few bytes per client-AP pair. Assuming 1440 byte packets, the overhead of the metadata amounts to 1-2%. In comparison, the throughput improvement expected from IAC is 1.5x to 2x, which more than compensates for the loss.

7.2 Concurrency Algorithm

The concurrency algorithm runs at the leader AP. The leader AP maintains a FIFO queue for traffic pending for the downlink and a similar queue for uplink requests learned from `DATA+Poll` frames (see §7.1). Given the queues of uplink and downlink traffic, the concurrency algorithm generates the uplink and downlink transmission groups. Without loss of generality, we will focus on the downlink.

There are multiple options for how to combine clients. The brute force approach considers all combinations of clients with queued packets and all different ways of assigning them to existing APs, computes the encoding and decoding vectors, and estimates the throughput of each combination. The throughput of a transmission group can be estimated without any transmissions as: $\sum_i \log(1+\|\vec{v_i^T} H_i \vec{w_i}\|^2)$, where the sum is over client-AP pairs, H_i is the channel for a pair, and $\vec{v_i}$ and $\vec{w_i}$ are the corresponding encoding and decoding vectors [29]. It then creates transmission groups for the queued packets which maximize throughput. There are two problems with such an approach. First, estimating the throughput for every combination of clients in the queue is a combinatorial problem in the number of clients. Second, since this approach focuses on maximizing throughput, it always prefers clients with good channels and hence is unfair. Alternatively, one can always create transmission groups by combining packets according to their arrivals in the FIFO queue. This approach is simple and gives each client a fair access to the medium, but is oblivious to the throughput of a particular grouping. In practice, different groups may yield significantly different throughput gains (see §10.3).

(a) The Best of Two Choices. IAC's concurrency algorithm balances the desire for high throughput with the need to be fair. To prevent starvation and reduce delay, it always picks the head of the FIFO queue as the first packet in the current transmission group. To reduce computational overhead, it picks other clients in the group using the best of two choices, a standard approach for reducing the complexity of combinatorial problems [23]. Say each group has three clients, and we already picked the first client in the group as the client whose packet is at the head of the transmission queue. We randomly pick two clients with queued packets as candidates for the second position in the group. Similarly, we also randomly pick two clients for the third position in the group. Now we estimate the throughput for the four transmission groups formed by these potential candidate clients and pick the group that optimizes throughput. As a result, instead of computing throughput for every possible combination of clients, we just compute it for four random client combinations.

Let us now consider the fairness of the approach. A client is considered for transmission either because it is at the head of the queue or because of a random choice. Both these cases give the client a fair access to the medium. However, since after picking the candidate clients, we still optimize for throughput, we need a mechanism to ensure that clients that never maximize throughput get picked. To do this, we assign a credit counter to each client. If the client is considered as a result of a random choice, and is ignored since it does not maximize throughput, the counter is incremented; but if it is picked for transmission the counter is reset. If the counter crosses a threshold, the client is selected as part of the group irrespective of the throughput. This mechanism ensures that every client is part of some group at least a minimum number of times.

[8] 802.11 calls the `Grant` frame `CF-Poll`, i.e., it is a poll without downlink data.

8 Channel Estimation

In IAC, the APs estimate and convey the channels to the leader AP as annotation on the decoded packets sent over the Ethernet.

(a) Uplink: To estimate the encoding and decoding vectors for the uplink, we need the physical channel from each concurrent client to each AP, as shown in Eqs. 3 and 4. In the absence of concurrent clients, estimating this channel is a standard MIMO technique [2]. Thus, the first time a client broadcasts an association message, all APs estimate the channel from that client to themselves. Once the APs have an initial estimate, they need to track it. This is done using the client's ack packets from the contention-free period, and its data packets from the contention period. Both packet types are transmitted without any concurrent transmissions. Hence they can be processed using standard MIMO channel estimation [2].

Since the APs can estimate the channel from every ack the client transmits, they obtain a frequent estimate of the channel. In static environments the channel is relatively stable and can be easily tracked at this estimation frequency. Slight inaccuracy in estimating the channel only means that the interference is not fully eliminated after applying the encoding and decoding vectors. As long as most interference is eliminated, the loss in throughput stays negligible.

(b) Downlink: Channel estimation is typically done at the receiver [7]. Thus, we have two options: either have clients estimate and convey the channels to the leader AP when it polls them, or try to have APs estimate the channel by exploiting reciprocity between uplink and downlink channels. In our measurements, the latter option worked with sufficient accuracy and hence we adopt it. Reciprocity means that the channel from node A to node B is the transpose of the channel from B to A. Thus, an AP can use the uplink channel from a particular client to infer the downlink channel to that client.

It is important to understand that channel reciprocity does not mean that the link between two nodes A and B is symmetric. Reciprocity (i.e., the kind that we care about in this paper) means that the channel coefficients are the same, but the noise or interference could be vastly different. For example, if A transmits symbol x, node B receives $y_B = Hx + n_B$. Similarly, in the opposite direction, node A receives $y_A = Hx + n_A$. The channel multiplier, H, is the same, but the noise could be much higher at A if it is close to a microwave oven. Hence, one may see many packet drops at A but not at B, but this does not contradict reciprocity. Reciprocity has been confirmed in measurements [16, 28, 27] and is used in QUALCOMM's 802.11n proposal [2].

Reciprocity cannot be applied directly without calibration to account for hardware differences between the tx and rx chains. The calibration however can be computed once and does not change for the same sender receiver pair. IAC uses a calibration method from the QUALCOMM's 802.11n proposal [2]. Let H^d be the channel between a particular AP and client pair, and H^u the uplink channel from that client to the same AP. Then:

$$(H^d)^T = C_{Client,rx}\, H^u\, C_{AP,tx}, \tag{8}$$

where H^T refers to the transpose of H, and $C_{Client,rx}$ and $C_{AP,tx}$ are constant diagonal matrices that describe the extra attenuation and delay observed by the signal in the transmit and receive hardware chains on the client and the AP respectively.

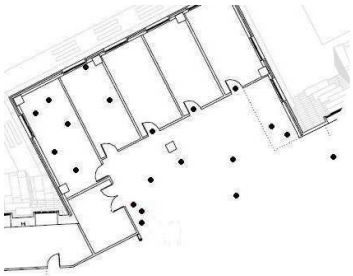


Figure 11: Testbed Topology.

9 Complexity

IAC multiplies each packet with an encoding vector at the transmitter and projects on a direction orthogonal to interference at the receiver. Both pre-coding and projection are general operations in MIMO designs [2]. IAC also performs interference cancellation, which is linear in the number of cancelled packets. Since, the packets cancelled at an AP are already decoded at prior APs, all the packets can be cancelled in parallel. Hence, the delay from cancellation can be made independent of the number of cancelled packets.

10 Performance Evaluation

We evaluate IAC in a testbed of MIMO software radios. Each node is a laptop connected to a 2-antenna USRP radio board and runs the GNU-Radio software. To create a MIMO node, we equip each USRP with two RFX2400 daughterboards. We also set the MUX value in software to allow the FPGA to process samples from both antennas.

(a) Topology. Our testbed, shown in Fig. 11, has 20 nodes. Each node has two antennas. All nodes are within radio range of each other to ensure that concurrent transmissions are enabled by the existence of multiple antennas, not by spatial reuse.

(b) Modulation. IAC uses the modulation/demodulation module as a black-box and hence works with a variety of modulation schemes. Our implementation, however, uses BPSK, which is the modulation scheme that 802.11 uses at low rates.

(c) Parameters. We use the default GNU-Radio parameters. However, in order to drive two antennas at the same time, we double the interpolation and decimation rates at the transmitter and the receiver. Each packet consists of a 32-bit preamble, and 1500-byte payload.

(d) Compared Schemes. We compare the following:

- **IAC:** This is our implementation of IAC.
- **802.11-MIMO:** There are multiple proposals for 802.11n [2, 1]. These schemes are all point-to-point, i.e., they allow only one transmitter to access the medium at any point in time. They however differ in the amount of channel information available to the transmitter, with more channel information leading to better performance [29]. Since IAC uses full channel information, we compare it with an 802.11 MIMO design with full channel information available to both sender and receiver. This design is based on QUALCOMM's eigenmode enforcing [2] and uses an approach that is proven optimal for point-to-point MIMO [29].

(e) Setup. In each experiment, we randomly pick some nodes to act as APs and others to act as clients. We repeat the same experiment with IAC and 802.11-MIMO. Three points are worth noting.

- First, we allow 802.11-MIMO access to the same number of APs as IAC. Though 802.11-MIMO cannot use the additional APs for concurrent transmissions, it can use them to increase diversity. For example, if there are three APs, each 802.11-MIMO client communicates with the AP to which it has the best SNR.
- Second, we use a simplified TDMA MAC for both IAC and 802.11-MIMO. The MAC assigns the same number of transmission timeslots to the two schemes. Consider an uplink scenario that involves three clients and three APs. We start with the 802.11-MIMO experiment and assign each client to its best AP. Each client transmits for 100 time slots, for a total of 300 time slots for the 802.11-MIMO experiment. We follow with an IAC experiment where clients transmit together for a total of 300 time slots. We then repeat the experiment for a different client set. This simplified MAC allows for a fair comparison between IAC and 802.11-MIMO because it assigns the medium equally to each scheme. Implementing the MAC in §7 requires access to accurate timing information, and the ability to quickly switch the board from a transmit mode to a receive mode. These requirements are not supported by the current USRP-GNU-Radio platform.
- Finally, both IAC and 802.11-MIMO use the GNU-Radio basic decoding modules (e.g., packet detection, clock recovery, synchronization, and channel estimation) and the same system parameters.

(f) Metric. It is typical in the networking community to compare the throughput of various designs. Throughput results, however, do not bring much insight for radios that do not have proper rate adaptation. Specifically, both in theory and practice, wireless systems (e.g., 802.11a/b/g/n cards, WiMax, etc.) can exploit a higher SNR to use denser modulation and coding schemes, and hence increase their throughput. GNU-Radios however do not yet support rate adaptation. In this case, it is not sufficient to compare throughput because two systems may have the same throughput yet one of them has a higher SNR. In an actual wireless product, the higher SNR system would use better modulation and coding schemes to achieve a higher throughput but current GNU Radios cannot exploit this higher SNR. Another way to look at the problem is as follows. Say we take a 2-antenna system and show that IAC can decode four concurrent packets, while 802.11-MIMO decodes only 2 concurrent packets. In this case, the throughput of our system will be double the throughput of 802.11-MIMO. Such a result however is ambiguous because it is not clear whether the 802.11-MIMO system has a higher SNR. If it does, then 802.11-MIMO could have used denser modulation and coding schemes, potentially doubling its throughput, or maybe tripling it. Because of this ambiguity, it is preferable to measure performance at the physical layer in terms of SNR or a function of it.

Thus, for both 802.11-MIMO and IAC, we measure the signal to noise ratio, $SNR_{Measured}$, for each transmitted packet. We compute the achievable rate, i.e., the rate that could be achieved in the presence of optimal rate adaptation [29]:

$$Rate = \sum_i \log_2(1 + SNR^i_{Measured})[bit/s/Hz], \tag{9}$$

where the sum is over all concurrent packets. For each scheme, we average the above rate over the whole experiment, and compute the gain as the ratio of the average rate of IAC to that of 802.11-MIMO:

$$Gain = \frac{Rate_{IAC}}{Rate_{802.11-MIMO}}. \tag{10}$$

10.1 IAC's Multiplexing Gain

The main advantage of IAC is that it increases the number of concurrent packets, i.e., it provides a multiplexing gain. In §5, we have demonstrated this gain analytically. Here, we check it in practice.

Experiment with 2-by-2 Uplink. We randomly pick two clients from the testbed to upload traffic to two APs, then repeat the experiment with different clients and APs. We compare IAC to 802.11-MIMO. In 802.11-MIMO, each client uses its best AP and transmits two packets simultaneously, and the two clients alternate in using the medium. In IAC, the two clients simultaneously transmit three

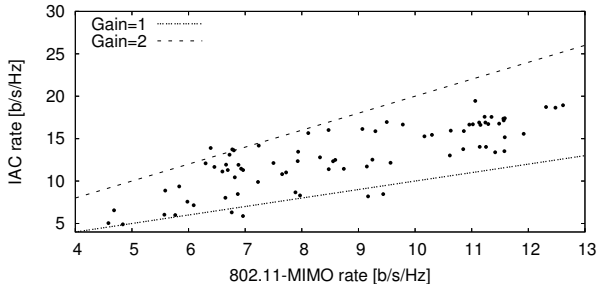


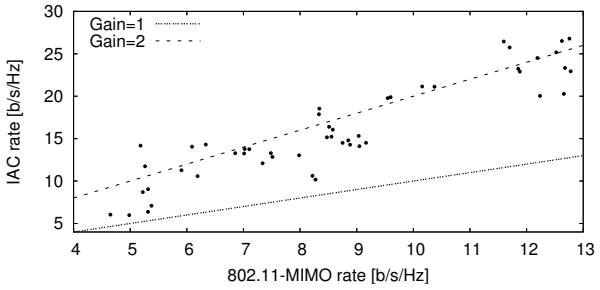
Figure 12: 2-Client and 2-AP Uplink. The figure shows a scatter plot of the average rate under IAC and 802.11-MIMO. The rate is measured as bits per second per a Hz of frequency bandwidth. The two lines are for reference; they illustrate the cases of no-gain in transfer rate, i.e., "Gain=1" and a doubling of transfer rate, i.e., "Gain=2". The figure shows that for the 2-client and 2-AP uplink scenario, on average, IAC increases the transfer rate by 1.5x over 802.11-MIMO.

packets to both APs, but in one time slot, client 1 uploads a single packet and client 2 uploads two packets, while in the next slot, client 1 uploads two packets and client 2 uploads one packet.

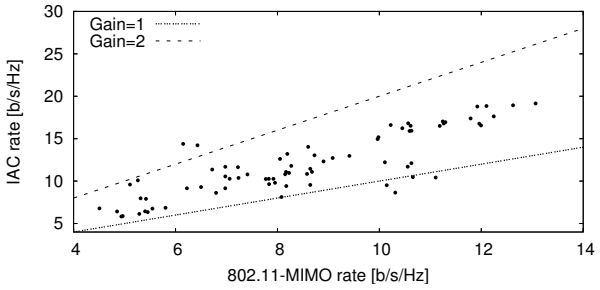
Results for 2-by-2 Uplink. Fig. 12 shows a scatter plot of the average rate under IAC and 802.11-MIMO. Each point on the graph corresponds to a particular 2-client and 2-AP choice, and is generated as follows. We first transmit packets using 802.11-MIMO and measure the received SNR for each packet. We compute the 802.11-MIMO achievable rate according to Eq. 9. Second, we repeat the experiment for the same 2-client and 2-AP choice but with IAC. Similarly, we measure the SNR for each received packet and compute the achievable rate under IAC using Eq. 9. For each point in the figure, the x-axis shows the average rate under 802.11-MIMO while the y-axis shows the average rate under IAC.

The figure supports the analysis, showing that for the 2-client and 2-AP uplink, IAC's transfer rate is on average 1.5x higher than 802.11-MIMO. The figure also shows a significant variance around the average gain, i.e., in certain experiments the gain is less than 1.5 and in others it is more. The variance in the gain is partially due to channel and noise variations over the duration of an experiment. More importantly, the variance is mainly due to relative differences between the channels of the two clients in an experiment. In particular, IAC's gain is typically lower when the channel matrices of the two clients are similar. To see why this is the case, consider the extreme scenario when the two clients have exactly the same channels to the two APs (i.e., $H_{11} = H_{21}$ and $H_{12} = H_{22}$). In this case, aligning the two clients at one AP implies aligning them at the other AP, and hence you cannot decode. In practice, two clients are unlikely to have the same channel to both APs. However, the more similar their channel matrices, the more the alignment is affected by noise and imperfection of channel estimates, and hence the less the gain from IAC. On the other hand, IAC's gain may exceed $1.5x$ because of spatial diversity. Specifically, in IAC, one of the concurrent clients uploads two packets and the other uploads one. The client that uploads one packet uses both antennas to transmit. This creates a diversity gain that increases the received SNR and the achievable rate. Thus, in addition to its multiplexing gain, IAC can exploit diversity to achieve a higher rate for this packet, getting a higher gain over 802.11-MIMO than analytically demonstrated. This diversity gain is further studied in §10.2.

Experiment with 3-by-3 Uplink and Downlink. Next, we want to check whether IAC can further increase the multiplexing gain. In §5, we found the bound on the number of concurrent packets. Since our nodes have 2 antennas each, we expect IAC to multiplex 4 packets on the uplink and 3 packets on the downlink. We examine whether our implementation can deliver these rates.



(a) Uplink.



(b) Downlink.

Figure 13: 3-clients and 3-APs. The figure shows a scatter plot of the rate under IAC and 802.11-MIMO, for 3-client and 3-AP scenarios. The two lines are for reference; they illustrate the cases of no-gain in transfer rate, i.e., "Gain=1" and a doubling of transfer rate, i.e., "Gain=2". The results show that, on average, IAC increases the rate by 1.8x on the uplink and 1.4x on the downlink.

Each experiment involves three clients and three APs, and is run for 802.11-MIMO and then for IAC. In the 802.11-MIMO experiments, each client accesses the medium alone and uploads/downloads 2 packets per timeslot. The medium is arbitrated between the three clients. In IAC, all clients access the medium concurrently. In downlink experiments, each client transmits 1 packet per timeslot. In uplink experiments, in every timeslot, one of the clients transmits 2 packets, the other clients transmit one packet each. We choose the client that transmits the two packets in each timeslot in a round robin manner.

Results for 3-by-3 Uplink and Downlink. Figs. 13a and 13b show scatter plots of the rate under IAC and 802.11-MIMO for the 3-client and 3-AP scenario. For each point, the x-axis shows the total rate of the three clients involved in that experiment when they use 802.11-MIMO, whereas the y-axis shows the total rate of the same clients when they use IAC. The figures show that IAC provides about 1.4x increase in transfer rate on the downlink and 1.8x on the uplink. Furthermore, these gains are achieved at both low and high rates (i.e., low and high SNRs).

10.2 IAC's Diversity Gain

Our discussion so far has focused on scenarios with multiple clients, where IAC provides a multiplexing gain over 802.11-MIMO. But, what if there is only one client? In this case, IAC has no multiplexing gain over 802.11-MIMO, i.e., in both schemes, the maximum number of concurrent packets that can be communicated to/from one client is two (since it has 2 antennas). However, because of its ability to coordinate multiple APs over the Ethernet, IAC still exhibits a diversity gain over 802.11-MIMO. This diversity gain arises from the ability to choose between transmit-receiver antenna pairs. For example, consider the downlink when there is one client and two APs. We want to deliver two concurrent packets to the client. 802.11-MIMO can exploit diversity by selecting the best among the two

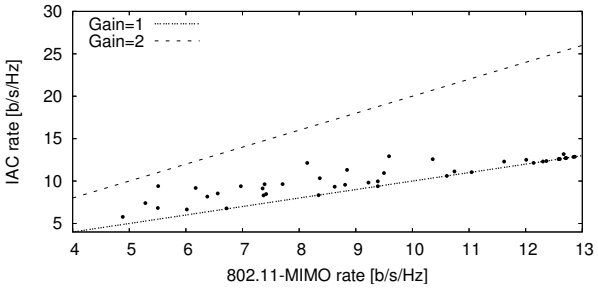


Figure 14: 1-client and 2-APs. The figure shows that IAC is beneficial even when the network has one active client. In this case, IAC provides a diversity gain over 802.11-MIMO because it allows the client to choose between downloading two concurrent packets from one of the two APs, or using both APs concurrently, downloading one packet from each.

APs.[9] IAC however has more options because it can exploit diversity across APs; for example it can use one antenna from each AP or use all four antennas together or any option in between.[10]

Experiment with 1-client and 2-APs. Every experiment uses one random client and two APs. The client downloads 100 packets. We compare an 802.11-MIMO design where the client downloads its packets from the best AP (i.e., the AP that delivers the highest SNR to the client) with a IAC design where the two APs cooperate on downloading the packets to the client. Specifically, the leader AP compares the following options: transmit one packet from each AP, and transmit both packets from one of the two APs. It picks the option that has a better throughput. In both 802.11-MIMO and IAC, two packets are transmitted simultaneously in every timeslot.

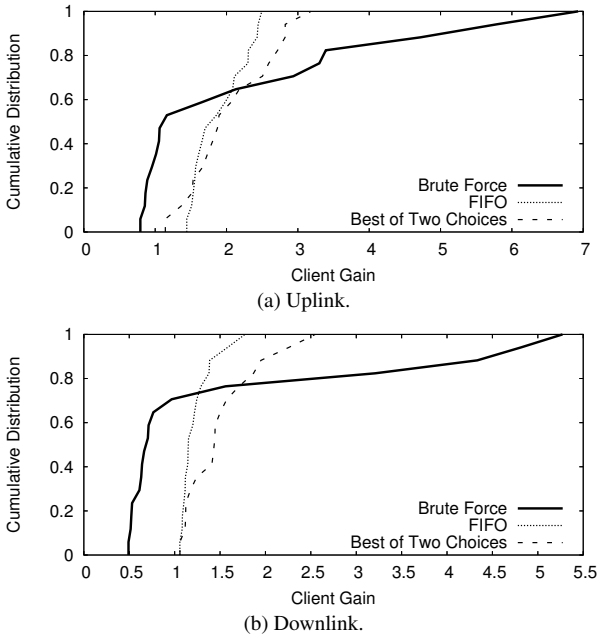
Results for 1-client and 2-APs. Fig. 14 plots the increase in download rate achieved with IAC in comparison with 802.11-MIMO. The figure reveals the following:

- IAC is beneficial even when the network has only one active client.
- IAC has a diversity gain over 802.11-MIMO. This is because 802.11-MIMO can choose only between APs, but IAC can exploit antenna diversity across APs.
- Diversity is particularly beneficial at low rates (i.e., low SNRs), where the rate could double with IAC. This is expected since having two diverse choices typically gives an SNR improvement of about 1-3 dB [29]. This translates to high relative gains at low SNRs, but relatively low gains at higher SNRs.

10.3 IAC in a Large Network

We investigate IAC's performance in a large network with many active clients (e.g., a large conference room). When the number of clients is larger than the maximum number of concurrent packets, one has many options for which clients transmit concurrently, both on the uplink and downlink. Choosing a particular option impacts both fairness and the total rate. In fact, in any wireless network, there is always a tension between maximizing transfer rate and ensuring fairness because the best option in terms of rate would always transmit to the client with the best channel and starve the others. Thus, we want to look at the performance in terms of both fairness and rate maximization.

In particular, we compare 802.11-MIMO against three algorithms for picking concurrent clients. All three algorithms pick the packet at the head of the queue for transmission; however, they differ in the choice of which other packets to concurrently transmit with the head-of-the-queue packet. The first algorithm is a *brute force* search that finds packets in the queue that maximize the rate.[11] The second algorithm, which we refer to as *FIFO*, combines the packets according to their arrival order. The third algorithm is the *best of two choices*, which is explained in §7.2. This is the choice that IAC adopts.



(a) Uplink.

(b) Downlink.

Figure 15: Gains in Transfer Rate for the Whole Testbed. The figure shows CDFs of client gains for three IAC concurrency algorithms. Each CDF is taken over 17 active clients. The figure shows that the three variants of IAC behave differently. IAC+brute-force delivers extreme gains to some clients while reducing the rate of other clients below their rates with 802.11-MIMO. IAC+FIFO is fairer but has low overall gains. IAC+best-of-two has the best fairness-throughput tradeoff.

Experiment. We use all nodes in the testbed in Fig. 11. We pick three nodes to be APs and let the other 17 nodes be clients. Each client has infinite demands. This ensures that a client's throughput is not limited by its own demands but by how the concurrency algorithm chooses to serve the client. Packets from different clients arrive at the system in random order. Each run involves using the medium for 1000 timeslots, and we repeat a run 3 times to compute the average rate per client. We run the experiment with four designs: 802.11-MIMO, IAC+best-of-two, IAC+brute-force, and IAC+FIFO. For each client, we compute the average rate it achieves under 802.11-MIMO and the three variants of IAC. For each variant of IAC, we compute the change in client transfer rate in comparison to 802.11-MIMO, i.e., the gain seen by each client. We compare the three variants of IAC by comparing their gains over 802.11-MIMO.

Results. Figs. 15a and 15b show the CDFs of the gains of the three IAC concurrency algorithms with respect to 802.11-MIMO, both on the uplink and downlink. The figures reveal the following findings:

- All three approaches for choosing concurrent packets provide a significant gain over 802.11-MIMO. The average gain on uplink is: 2.32x for the brute force approach, 1.9x for the FIFO approach, and 2.08x for the best-of-two approach. Similarly, on the downlink, the average gain is: 1.58x for the brute force approach, 1.23x for the FIFO approach, and 1.52x for the best-of-two approach. (Note that while IAC's multiplexing gain is bounded by 2x, the total gain can be larger because it includes diversity gains.)

[9] The literature also presents another way in which 802.11-MIMO can exploit diversity. In this case, the two antennas are used to send/receive the same packet/symbol. This approach however is less desirable since it requires 802.11-MIMO to give up on sending two concurrent packets, and repeat the same packet on both antennas [29].

[10] Note that comparing these options to find the best can be done merely by computing the capacity using our knowledge of the channel matrices [29].

[11] The relative rate can be estimated without transmitting the packets as $\sum_i log(1 + ||\vec{v}_i^T H_i \vec{w}_i||^2)$, where the sum is over client-AP pairs, H_i is the channel for a pair, and $\vec{v}_i$ and $\vec{w}_i$ are the corresponding encoding and decoding vectors [29].

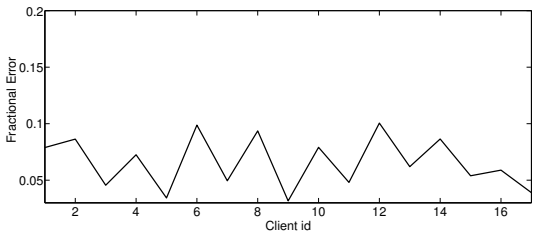


Figure 16: Channel Reciprocity. This figure plots the fractional difference between the direct estimate of a downlink channel and its estimate based on reciprocity. The x-axis refers to runs at different clients/locations. The figure shows that estimates based on reciprocity provide a reasonable accuracy and can be used in IAC.

- The three approaches differ widely with respect to fairness. In particular, the brute-force approach is significantly unfair. A few clients get a humongous boost in transfer rate, while many clients have a gain smaller than 1, i.e., their rates are better with 802.11-MIMO. The other schemes have a better fairness, with the best-of-two approach having the best fairness-throughput tradeoff.
- Thus, IAC, which employs the best-of-two approach, provides good fairness and high throughput. It delivers an average rate increase of 2.08x on the uplink and 1.52x on the downlink. Further, no client suffers a notable reduction in rate in comparison to 802.11-MIMO.

10.4 Channel Reciprocity

Finally, we check whether channel estimates based on reciprocity are accurate enough to be used in IAC.

Experiment. We take 17 random client-AP pairs from the testbed, and measure their uplink and downlink channels. We compute the calibration matrices according to Eq. 8. For each pair, we then fix the AP and move the client. This causes the uplink and downlink channels to change (but the calibration matrices stay the same.) We now make the AP measure the uplink channel, H^u, and multiply it by the calibration matrices to estimate the downlink channel $H^d_{reciprocity}$. We compare this estimate with the downlink channel as estimated at the client, H^d_{true}. We compute the fractional error in the AP's estimate as $Err = \frac{\|H^d_{true} - H^d_{reciprocity}\|}{\|H^d_{true}\|}$. We repeat the experiment 5 times for each client, where each run is done in a new location. For each of the 17 client-AP pairs, we plot the average fractional error in Fig. 16.

Results. The figure shows that reciprocity holds to a large extent. The fractional error between the actual downlink channel and the estimate based on reciprocity stays small. Note that since the client changed location between the estimation of the calibration matrices and their later application to estimate the downlink channel, reciprocity is reasonably accurate despite client movement. This result does not contradict prior measurements which show that links could be highly asymmetric in their loss rate. Reciprocity refers only to the channel matrix, but the performance of a link depends also on the noise level at the receiving node, which could be highly asymmetric.

11 Conclusion

This paper introduces interference alignment and cancellation (IAC). IAC weaves two signal processing techniques: interference alignment and interference cancellation, such that the combination applies to new scenarios that could not have benefited from either technique alone. We show both analytically and via a prototype implementation that IAC doubles the throughput of MIMO LANs.

We believe that IAC can provide benefits in scenarios other than those explored in the paper. For example, IAC also extends to clustered MIMO networks, which can occur in ad-hoc and mesh settings, like that in Fig. 17, where links within a cluster are strong (i.e., high bitrate) and links across clusters are weak (i.e.,low bitrate). The throughput of clustered networks is bottlenecked by the low bitrate inter-cluster links. IAC can double the throughput of the inter-cluster bottleneck links. In fact, this scenario is analogous to a WLAN where nodes in the same cluster can be thought of as being connected with a high bandwidth Ethernet. We believe that IAC can naturally increase throughput in these settings. Further exploration of IAC in ad hoc settings is left for future work.

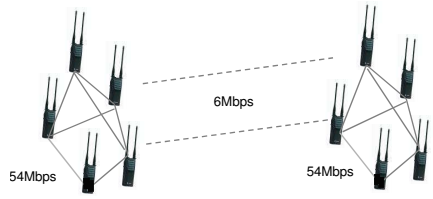


Figure 17: Clustered MIMO Ad Hoc Networks. Links within a cluster have high rates, while links across clusters have low rates and hence are the bottleneck. IAC doubles the throughput over these bottleneck links, hence increasing the overall network throughput.

Acknowledgments: We thank Nate Kushman and David Malone for their comments. This work is funded by DARPA ITMANET.

References

[1] Antenna selection and RF processing for MIMO systems. *IEEE 802.11-04/0713r0*, 2004.

[2] System Description and Operating Principles for High Throughput Enhancements to 802.11. *IEEE 802.11-04/0870r*, 2004.

[3] M. A. Ali, S. A. Motahari, and A. K. Khandani. Communication over MIMO X Channels: Interference Alignment, Decomposition, and Performance Analysis. *Tran. on Info. Theory*, 2008.

[4] J. Andrews. Interference cancellation for cellular systems: A contemporary overview. *IEEE Wireless Communications*, 2005.

[5] D. Bliss, K. Forsythe, and A. Chan. Mimo Wireless Communications. *Lincoln Journal*, 2005.

[6] V. Cadambe and S. Jafar. Interference Alignment and the Degrees of Freedom for the K User Interference Channel. In *Trans. on Information Theory*, 2008.

[7] P. Castoldi. *Multiuser Detection in CDMA Mobile Terminals*. Artech house Publishers, 2002.

[8] R. Chandra, P. Bahl, and P. Bahl. MultiNet: Connecting to Multiple IEEE 802.11 Networks Using a Single Wireless Card. In *INFOCOM*, 2004.

[9] S. Changho and D. Tse. Interference Alignment for Cellular Networks. In *Allerton*, 2008.

[10] J. M. Cioffi. *Multi-channel Modulation*. Stanford University.

[11] A. E. Gamal and T. Cover. Multiple user information theory. In *Trans. on Info. theory*, 1980.

[12] M. Gast. *802.11 Wireless Networks*. O'Reilly, 2005.

[13] D. Gesbert, M. Kountouris, R. W. Heath, C. Chae, and T. Salzer. Shifting the MIMO Paradigm: From Single User to Multiuser Communications. In *Sig. Proc. Mag.*, 2007.

[14] S. Gollakota and D. Katabi. ZigZag Decoding: Combating Hidden Terminals in Wireless Networks. In *Sigcomm*, 2008.

[15] S. Gollakota, S. Perli, and D. Katabi. Overcoming the antennas-per-node throughput limit in mimo lans. Technical report, MIT, 2009.

[16] M. Guillaud, D. Slock, and R. Knopp. A practical method for wireless channel reciprocity exploitation throught relative calibration. In *Sig. Process. and Apps*, 2005.

[17] R. Gummadi and H. Balakrishnan. Wireless Networks should Spread Spectrum Based on Demands. In *Hotnets*, 2008.

[18] D. Halperin, T. Anderson, and D. Wetherall. Taking the sting out of carrier sense: Interference Cancellation for wireless LANs. In *ACM Mobicom*, 2008.

[19] J. Hou, J. Smee, H. D. Pfister, and S. Tomasin. Implementing Interference Cancellation to Increase the EV-DO Rev A Reverse Link Capacity. *IEEE Communication Magazine*, 2006.

[20] C. Huang and S. Jafar. Degrees of Freedom of the MIMO Interference Channel with Cooperation and Cognition. In *arxiv: 0803.1733*, 2008.

[21] S. Jafar and S. Shamai. Degrees of Freedom of MIMO X Channel. In *Tran. in Info. Theory*, 2008.

[22] S. Kandula, K. Lin, T. Badirkhanli, and D. Katabi. FATVAP: Aggregating AP BackHaul Bandwidth. In *NSDI*, 2008.

[23] M. Mitzenmacher. *The Power of Two Choices in Randomized Load Balancing*. PhD thesis, 1991.

[24] A. Miu, H. Balakrishnan, and C. E. Koksal. Improving Loss Resilience with Multi-Radio Diversity in Wireless Networks. In *Mobicom*, 2005.

[25] T. Moscribroda, R. Chandra, Y. Wu, S. Sengupta, P. Bahl, and Y. Yuan. Load-Aware Spectrum Distribution in Wireless LANs. In *ICNP*, 2008.

[26] R. Murty, J. Padhye, R. Chandra, A. Wolman, and B. Zill. Designing High Performance Enterprise Wi-Fi Networks. In *NSDI*, 2008.

[27] C. Qiu, C. Zhou, G. Nan, and J. Zhang. Time Reversal with MISO for ultra-wideband Communications: Experimental Results. In *Letters on Antennas and Propogation*, 2006.

[28] C. Qiu, C. Zhou, J. Zhang, and G. Nan. Channel reciprocity and time-reversed propagation for ultra-wideband communications. In *Symp. on Antennas and Propagation*, 2007.

[29] D. Tse and P. Vishwanath. *Fundamentals of Wireless Communications*. Cambridge Press, 2005.

[30] P. Viswanath and D. Tse. Sum capacity of the vector gaussian channel and uplink-downlink duality. In *Trans. on information theory*, 2003.

[31] G. Woo, P. Kheradpour, and D. Katabi. Beyond the Bits: Cooperative Packet Recovery Using PHY Information. In *ACM MobiCom*, 2007.