This paper was presented as part of the main technical program at IEEE INFOCOM 2011

Contrabass: Concurrent Transmissions without Coordination for Ad Hoc Networks

Sungro Yoon, Injong Rhee, Bang Chul Jung†, Babak Daneshrad‡ and Jae H. Kim*
North Carolina State University, †Gyeongsang National University,
‡University of California, Los Angeles, *The Boeing Company
{syoon4, rhee}@ncsu.edu, bcjung@gnu.ac.kr, babak@ee.ucla.edu, jae.h.kim@boeing.com

***Abstract*—A practical protocol jointly considering PHY and MAC for MIMO based concurrent transmissions in wireless ad hoc networks, called *Contrabass*, is presented. Concurrent transmissions refer to simultaneous transmissions by multiple nodes over the same carrier frequency within the same interference range. Contrabass is the first-to-date open-loop based concurrent transmission protocol which implements simultaneous channel training for concurrently transmitting links without any control message exchange. Its MAC protocol is designed for each active transmitter to independently decide to transmit with near optimal transmission probability. Contrabass maximizes the number of successful concurrent transmissions, thus achieving very high aggregate throughput, low delays and scalability even under dynamic environments. The design choices of Contrabass are deliberately made to enable practical implementation which is demonstrated through GNURadio implementation and experimentation.**

I. Introduction

An important attribute of multi-antenna processing is the ability to null out co-channel interference for improved signal quality. Interference can be canceled at the transmitters by using pre-coding techniques, at the receivers by adjusting antenna weights, or both. This ability enables *concurrent transmissions*, defined to be simultaneous transmissions over the same carrier frequency by multiple transceivers residing in the interference ranges of each other.

Consider a network where each node has m antennas. In theory, up to m concurrent transmissions can be successfully decoded at receivers using the multiple input multiple output (MIMO) detection techniques such as zero forcing (ZF), minimum mean squared error (MMSE) and maximum likelihood (ML). Any concurrent transmissions by more than m transceivers are considered as *collision*. The recent standard for MIMO communication, namely IEEE 802.11n [1], adopts only a special case of concurrent transmissions in which only one node is involved in the transmission at a time, using all or part of its antennas and transceivers. The CSMA/CA is used as an MAC protocol to ensure only one node within an interference range to transmit at a time. We refer to this type of concurrent transmissions as *exclusive transmissions* in this paper, which is also called single-user MIMO in the literatures.

The exclusive transmissions miss out many opportunities of concurrent transmissions. With exclusive transmissions, a packet can be transmitted at a rate proportional to the number of transceivers (i.e., theoretically m times a link rate), so the transmission time of data is much shorter than using a single transceiver. However, control overhead such as time duration due to collisions, channel training, SIFS and DIFS, and ACK transmissions must still be incurred per packet. To mitigate the effect of this overhead, IEEE 802.11n aggregates multiple packets and transmits them as a single large frame. Although the aggregation amortizes the overheads over multiple sub-packets, there are situations where aggregation may not be possible due to the delay constraint of the packet. Concurrent transmissions, on the other hand, can naturally implement this amortizement without incurring delays because they permit multiple nodes to transmit at the same time. Furthermore, the transmission policy of FCC for unlicensed spectrum bands limits the per-node transmission power and in the exclusive transmissions, all antennas are co-located in the same node and the transmission power per antenna should be decreased as the number of transmit antennas increases [2]. However, in concurrent transmissions the transmission power per node remains constant independently of the number of concurrently transmitting links as long as the transmitters are in different nodes. The total power used for data transmission is increased as the number of concurrently transmitting nodes and the total data rate also increases compared to that of the exclusive transmissions with the same number of transmit antennas. See Sundaresan et al. [3] for initial performance comparison between concurrent transmissions and exclusive transmissions.

Realizing the full potential of concurrent transmissions requires an effective co-design of PHY and MAC protocol where interference cancelation and collision avoidance should work toward the same goal. The biggest challenge is to minimize the control overhead of the protocol. There have been a number of proposals for concurrent transmission protocols including [3]–[7]. Unfortunately, none of them is used in real systems because of high control overhead for supporting concurrent transmissions which diminishes the performance gain of concurrent transmission. Most overhead is incurred by coordination (e.g., using RTS and CTS) among concurrent transmitters to enable exclusive channel training.

Most of the earlier concurrent transmission protocols for ad hoc networks run in two phases. In the first phase, they perform RTS and CTS handshaking during which they perform exclusive channel training. During the transmission of RTS and CTS, no other transmitters can transmit so that each transmitter and receiver pair can learn the channel state information (CSI) without any interference from other transmitters. Typically, RTS and CTS embed training sequences needed for channel training. Another important task that RTS-CTS coordination

978-1-4244-9920-5/11/$26.00 ©2011 IEEE

performs is to determine the set of transmitters that can transmit during the second phase. The total number of concurrent transmissions in the second phase should not exceed the number of antennas at the receivers. In the second phase, concurrent transmissions by the chosen set of transmitters take place to transmit data packets. Since RTS and CTS must be sent at a low rate, this handshaking incurs very high overhead.

In this paper, we design a joint PHY/MAC protocol for MIMO based concurrent transmissions in ad hoc networks which does not involve infrastructures and operates without any coordination. By no coordination, we mean that the protocol must not introduce control frames other than ACK. The protocol is called *Contrabass* which is named after CONcurrent TRAnsmission.

There are a number of challenges for coordination-free concurrent transmissions. First, a receiver needs to estimate the channel state from a transmitter without any explicit coordination. The difficulty lies in that a receiver does not know in advance who will send without prior coordination. Second, the number of concurrently transmitting transceivers must be kept to less than or equal to m to avoid collision. To solve these problems, Contrabass implements simultaneous channel training by developing a set of innovative PHY-level techniques. Simultaneous channel training ensures channel training of concurrently transmitting channels without explicit and separate coordination. The accompanying MAC protocol of Contrabass achieves near optimal channel utilization by ensuring, again with no coordination, the number of concurrent transceivers as close as m for high channel utilization. Such a control is non-trivial because no prior information is given about the time-varying number of contemporarily competing transceivers, potentially located at different nodes.

We have implemented the PHY portion of Contrabass in the GNURadio platform [8] and conducted a proof of concept experiment (Section IV). Our experiment indicates that Contrabass successfully decodes signals from concurrent transmissions using its PHY protocols. Since GNURadio do not permit an effective implementation of real-time carrier sensing (see [9]), we implemented MAC of Contrabass and other MIMO protocols using NS-2. Here, Contrabass and two of existing concurrent transmission protocols, and IEEE 802.11n are tested under diverse network and traffic conditions. We verify through simulation that Contrabass yields high scalability under various traffic loads and outperforms the existing concurrent transmission protocols. Compared to IEEE 802.11 with frame aggregation, it achieves about 60 to 70% performance improvement with aggregated throughput under high load and about 4x performance improvement with VoIP traffic. The performance results are presented in Section V.

II. Related Work

[3], [4] perform simultaneous transmission of RTS and CTS, while [6], [7], [10], [11] use separate staggered RTS and CTS message exchange. When simultaneous RTS and CTS transmissions are allowed, either the perfect knowledge of CSI [3] or use of spreading coding [4] is assumed for the decoding of the overlapped control message signals. While the assumption on the perfect knowledge of CSI is unrealistic, use of spreading coding incurs significant overhead because RTS and CTS are transmitted at a low data rate and spreading each symbol consumes too much channel resource. Although Contrabass also uses PN-sequence for channel training, its use is limited to training sequences (about 31 symbols). On the other hand, encoding the entire frames (typically 14 bytes) of RTS and CTS incurs too high overhead (assuming 8 symbols/bit spreading factor, 1792 symbols ($14\times2\times8\times8$)).

The existing protocols also differ in the scheduling algorithms that select concurrently activated links in the second phase. This is based on various factors such as traffic demands and diversity gains which are gathered using the control messages. While some protocols [6], [7], [10], [11] limit the degree of concurrency, [4] [3] propose optimal scheduling algorithms that fully exploit both spatial gains, array and diversity gains. However, most of these scheduling algorithms require complicated computations at the time scale of per-packet transmissions.

III. Contrabass

In this section, we present our protocol, called *Contrabass* that implements coordination-free concurrent transmissions. Contrabass consists of PHY and MAC components.

A. Assumptions and Definitions

Each node is assumed to have m antennas. We refer to an antenna and its corresponding transceiver as a *transmitter*. The m different transmitters in concurrent transmissions may or may not belong to the same node. A transmitter is *active* if it has a packet to transmit. We define a *channel* to be the wireless link between a transmitting antenna and a receiving antenna. There are n active transmitters in the same interference range and a subset of transmitters transmit simultaneously. We assume that the number of concurrently transmitting nodes is equal to n_{tx}. A node cannot receive and transmit at the same time (i.e., half-duplex node).

Consider from a perspective of a receiver receiving signals through m antennas. Let us denote the channel coefficient from transmitter i to the jth antenna of the receiver as $h_{i,j}$. Then the MIMO channel is described as $y = Hx + w$ where y is an m dimensional receive vector, x is an n_{tx} dimensional transmit vector, w is an m dimensional additive white Gaussian noise vector and H is the $m \times n_{tx}$ channel response matrix.

Given a rich scattered environment such as Rayleigh fading channel, wireless signals experience diverse paths to the destination and the channel coefficients become linearly independent from each other. With $n_{tx} \leq m$, it is possible for a receiver to recover the original input vector x. Normally a channel filter is built for the process. The linear channel filter W is an $n_{tx} \times m$ matrix such that $\hat{x} = Wy$ where $\hat{x}$ is the estimation of the original input vector x by the receiver. The receiver must estimate all channels of n_{tx} transmitters through channel training in order to obtain W.

B. Physical Layer Protocol

Contrabass implements simultaneous channel training by embedding a unique training sequence in the preamble of each

data packet. We perform a set of PHY-level signal processing techniques to implement simultaneous training using the training sequences. A receiver performs RLS (recursive least squares) filtering on the received signal (containing the training sequences super-imposed together through simultaneous transmission) to create a channel filter for a receiver without explicitly and separately estimating all the contributing interfering channels. Since in concurrent transmission, it is possible that multiple data packets are transmitted to the same receiver, a receiver needs to build multiple channel filters, each tuned to the channel state from each contributing transmitter. To enable this, we use two additional signal processing techniques called RIC (random index correlation) and SIC (successive interference cancelation). RIC is used to identify the number of concurrent transmitters to that receiver and extract their training sequences. Since multiple transmitters might be located at different distances from the receiver, received signals may have different powers. This is commonly called the near-far problem. SIC is applied to to enhance the performance of training sequence extraction even under differing received powers of multiple signals. With exception of RIC, RLS and SIC are previously developed [12]–[14]. Our key contribution in the PHY layer is the innovative combination of these techniques to enable simultaneous channel training without coordination.

1) Training Sequence: In wireless communication, training sequences (TS) are commonly used for time and frequency synchronization and channel estimation. Typically TS is embedded in the preamble of each data packet. In Contrabass, to enable simultaneous channel training, we use training sequences with low cross correlation. For channel training, a receiver needs to know in advance the TS embedded in received frames. There are two choices of training sequences that a transmitter embeds in its transmitted frame to a receiver. The first choice is for a transmitter to compose a unique TS and embeds it into the frame. This requires its receiver to know in advance which transmitter is transmitting to it. The second choice, adopted by Contrabass, is that each receiver generates its unique TS from a well-known unique ID such as its MAC address. Since a transmitter knows the MAC address of its receiver, it can also generate and embed the TS of the receiver in the preamble of its frame. The receiver simply needs to match the received training sequences with its own. One issue with this approach is that concurrent transmitters sending to the same receiver will use the same TS. In order to distinguish these overlapped TS' without coordination, we devise a novel technique called RIC (described further next section) where each Contrabass transmitter varies its starting index of TS randomly. A Contrabass receiver correlates its TS with received signals by rotating the starting index of its TS. To maximize the chance of identifying the TS' of those concurrent transmitters, we use training sequences with very low auto-correlation. As a result, we adopt gold sequences which have very low cross/auto-correlation [15].

2) Random Index Correlation: Contrabass uses 31 symbol gold sequences for channel training. Denote by $f(i)$ the symbol at the ith position of a TS. Each transmitter picks a random index j from 0 to 30 and use the index as a starting position to transmit to its receiver in the following way. The symbols are rotated by j so that $f(i)$ is placed at $(i+j) \bmod 31$. This rotated TS is embedded in the preamble.

Suppose that n_{tx} transmitters simultaneously send their frames. By the MAC protocols, all these frames are synchronized and exactly overlapped. The receiver performs a cross-correlation function $C(t)$ by performing a convolution starting from index 0 to index 30.

$$C(t) = \sum_{k=0}^{30} f^*((t+k) \bmod l) \cdot g(k), \tag{1}$$

where f^* is the complex conjugate of the training symbol and g is the received symbol such that $g(k) = h_1 x_{1,k} + \cdots + h_{n_{tx}} x_{n_{tx},k}$, where h_i is the channel coefficient between transmitter i and the receiver, and $x_{i,k}$ is the kth symbol of the training sequence received from transmitter i.

RIC finds index s such that $C(s) = |\max_{0,30} C(t)|$. When a number s coincides the starting index of a training sequence from a certain transmitter, say 1, the correlation value will be

$$C(s) \quad = \quad \left| h_1 \sum_{k}^{30} |x_{1,k}|^2 + \cdots + h_{n_{tx}} \sum_{k}^{30} \overline{x_{1,k}} \cdot x_{n_{tx},k} \right|,$$

where $\bar{x}$ is the complex conjugate of the symbol x. With the TS having a pseudo orthogonality, it will have $\sum_{i \neq 1,k} \overline{x_{1,k}} \cdot x_{i,k} \simeq 0$. Hence, we will have the correlation peak of

$$C(s) \simeq |h_1| \sum_{k}^{30} |x_{1,k}|^2. \tag{2}$$

3) Recursive Least Squares Filtering: RLS filtering [12] trains a channel filter W directly from the input signal without explicitly estimating CSI and CVM (covariance matrix) using the following iterative algorithm.

$$W_i = W_{i-1} + (x_i - W_{i-1} y_i) y_i^* P_i,$$

$$P_i = \lambda^{-1} [P_{i-1} - \frac{\lambda^{-1} P_{i-1} y_i y_i^* P_{i-1}}{1 + \lambda^{-1} y_i^* P_{i-1} y_i}].$$

x_i indicates an m-dimensional vector defined as $[s_1 s_2 \ldots s_{n_{tx}} 0 \ldots 0]^T$ where s_j represent the ith symbol of the TS from the jth concurrent transmitter and n_{tx} is the number of the received frames to the same receiver. Therefore, by using RLS filter, up to m frames can be decoded and unknown interferences from other concurrently transmitting nodes to other receivers can also be effectively suppressed at the same time. The initial value of P_i is given as $P_{-1} = \epsilon^{-1} I$ where ϵ is a very small constant. λ is a forgetting factor, $0 \ll \lambda < 1$. W_i converges iteratively to W with an error feedback $x_i - W_{i-1} y_i$ without the need for estimating H and its covariance. RLS is known to have a SNR performance same with MMSE [16].

4) Successive Interference Cancelation: After a training sequence of a frame is correctly identified and used for channel training via RIC and RLS filtering, a Contrabass receiver cancels out the contribution of that training sequence from the received signals in the following way. First, the average

channel coefficient of the decoded symbols, $\bar{h}$, is obtained (e.g. h_1 in Eq. (2)). Then, the recovered symbols in the TS and the data frame are multiplied by $\bar{h}$ and subtracted from the mixed signals. On the resulting signals, Contrabass performs RIC and RLS to extract the second packet. This process is repeated until the extracted payload does not pass the integrity test or the receiver could not find a correlation peak.

In general, SIC is effective only when the signal strength ratio among interfering frames is large enough for the demodulator to decode symbols. This requirement interferes with rate adaptation [17]. In our protocol, this problem does not occur since SIC is applied only for training sequences and training sequences are always transmitted at a fixed rate. In addition, a receiver already knows its own training sequence which will be subtracted from the original signals. After finding the starting index using RIC, indicated by the correlation peak with the known TS, the known TS can be subtracted from the received signals. Since no decoding is involved in this stage, the signal strength ratio among interfering frames does not need to be large. Furthermore, since SIC is applied only to TS, it consumes very little computational resource. Note that the data portion of a frame is decoded using the channel filter obtained from RLS filtering without using SIC.

C. MAC Layer Protocol

The MAC protocol of Contrabass needs to achieve the following goals. (1) Contrabass requires every concurrent transmitter within an interference range to transmit at the same time and (2) the total number of concurrent transmitters must be less than or equal to m for collision avoidance, but as close to m as possible for high channel utilization.

To achieve the first goal, Contrabass adopts CSMA where each active transmitter performs carrier-sensing before transmission. After the medium becomes idle, each node waits for a fixed time interval (typically called DIFS in IEEE 802.11 standard). This ensures that all the concurrent transmitters start their transmission right after the DIFS and no new transmission can start in the middle of the other concurrent transmissions in the same range. As interference ranges are different from sensing ranges, this does not necessarily guarantee simultaneous transmission within an interference range. However, this can be considered as a good approximation. Note, if there is a hidden terminal outside the sensing range, it is possible that a new transmission may start in the middle of the other concurrent transmissions. Note that the same problem may also occur with IEEE 802.11n. A conventional technique to handle hidden terminal is to use RTS and CTS. But in practice, RTS and CTS are not commonly used because of their overhead. Furthermore, there are studies indicating that hidden terminals do not frequently occur [18]. If hidden terminals happen, their impact is no worse in Contrabass than in IEEE 802.11n without RTS and CTS.

At each time of transmission, an active transmitter transmits a new frame with a probability τ. Fig. 1 illustrates the basic structure of Contrabass MAC. To achieve the second goal, Contrabass adjusts τ in a way that the probability of successful transmissions is maximized. In this section, we first

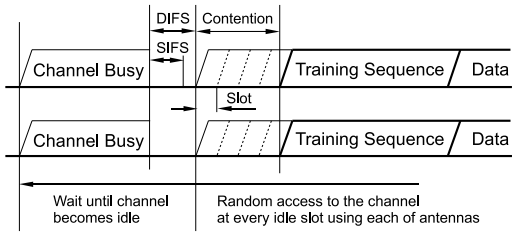

Fig. 1. The basic media access mechanism of Contrabass MAC

mathematically derive the optimal transmission probability, τ_{opt}, that maximizes the success probability given n. Second, we discuss a *binary search technique* that adjusts τ based on the recent history of transmission successes and idle slots so that it quickly converges to τ_{opt} without the knowledge of n.

After the successful receptions, ACK frames are sent back to their transmitters. As the duration of packet transmissions could vary due to different payload sizes and data rates, all the receivers wait until the last data transmission finishes. If the channel becomes idle again, within a pre-defined time interval called *short inter-frame space* (SIFS), all the successful receivers send ACK frames back to their original transmitters. Those receivers that receive multiple frames send ACK frames using their multiple transceivers. To ensure that the transmitters decode the ACK frames correctly, the receivers attach the unique training sequence that is generated from the MAC addresses of the transmitters. A receiver sends the training sequence again starting from a random index so that the transmitter can decode concurrently received ACK frames.

1) Optimal Transmission Probability: Suppose that the number of active transmitters n is known. Recall that the transmission is successful only when the number of simultaneous transmissions within the interference range is less than or equal to m. Given that the transmitters transmit their frames with probability τ, we can express the probability of successful transmission, p_S by adding the probabilities of all the events that the number of concurrent transmissions is less than or equal to m:

$$p_S = \sum_{i=1}^{m} \binom{n}{i} \tau^i (1-\tau)^{n-i}, \tag{3}$$

$$p_I = (1-\tau)^n, \tag{4}$$

$$p_C = 1 - p_S - p_I. \tag{5}$$

As in Fig. 2(b), the transmission success probability is a concave function of τ. We can find the τ_{opt} by finding τ that makes the first derivation of Eq. (5) 0. The first derivation of Eq. (5) with respect to τ gives

$$\begin{aligned} p'_S &= \sum_{i=1}^{m} \binom{n}{i} (i\tau^{i-1}(1-\tau)^{n-i} - \\ &\quad (n-i)\tau^i(1-\tau)^{n-i-1}). \end{aligned} \tag{6}$$

We can observe that the second term iteratively cancels out the first term in Eq. (6). Now, Eq. (6) can be simplified into

$$p'_S = n(1-\tau)^{n-1} - n\binom{n-1}{m}\tau^m(1-\tau)^{n-m-1}.$$

Finally, by finding τ that renders p'_S to 0, we easily obtain the optimal transmission probability τ_{opt} that maximizes Eq. (5):

$$\tau_{opt} = \frac{1}{\sqrt[m]{\binom{n-1}{m}} + 1}. \quad (7)$$

2) Binary Search Algorithm: In this section, we present a binary search technique that dynamically adjusts τ to converge to τ_{opt} for an unknown n.

The intuitions of the algorithm start from two observations. First, Fig. 2 (a) shows the values of τ_{opt} as n changes. It also plots the corresponding success, slot idle and collision probabilities under a given τ_{opt} value which are denoted by p_S^{opt}, p_I^{opt} and p_C^{opt}, respectively. These probabilities remain almost constant for large n. This property allows each transmitter to adjust its τ so that its observed numbers of idle slots, transmission success and collision approximates those observed when τ is equal to τ_{opt}. Second, in Fig. 2, we observe that p_I decreases and p_C increases monotonically as τ increases. By observing p_I and p_C, we can tell whether τ is larger or smaller than τ_{opt}. τ is increased when p_I is larger than p_I^{opt} and decreased when p_C becomes larger than p_C^{opt}. Note that p_I^{opt} and p_C^{opt} are known from Fig. 2 as their values are fixed independent of n for a sufficiently large n.

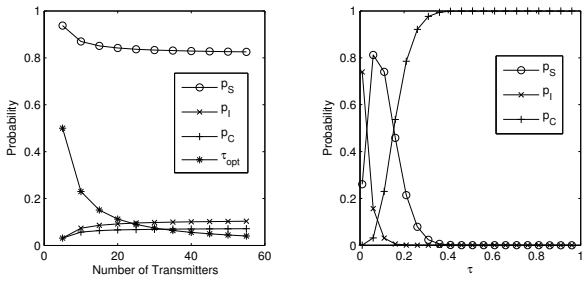

Fig. 2. (a) When τ_{opt} is applied, p_I, p_S and p_C is almost constant with different number of transmitters. (b) p_I and the p_C are used for estimating the current level of transmission attempt probability as they are monotonic functions, respectively.

In Contrabass, a node observes the results of any transmissions occurring within its carrier sensing range by detecting an ACK frame. If a data frame is followed by an ACK frame, it can be deduced that the transmission was successful. Otherwise, collision has occurred. Although a node may not participate in the transmission, it still overhears the channel to see if an ACK frame is transmitted.

Each node records the sequence of past N events, $e_0, e_1, \ldots, e_{N-1}$ that it has observed, in a reverse chronological order (with e_0 being the most recent). We call this an *event window*. There are three types of events: *idle*, *collision* and *success*. We assign *event values* 1, 0 and -1 to each idle, success and collision events respectively and E is a weighted moving average of event values: $E = \sum_{k=0}^{N-1} \lambda^k \cdot v(e_k)$ where λ is a forgetting weight < 1 and $v(e_k)$ is the value of the currently observed event. E gets larger as a node sees more idle events recently, and smaller as it sees more collision events recently. E summarizes the idleness of the channel. If E is larger than a threshold T_i, we say that the channel is in the *idle state* and if it is less than T_c, then the *collision state*. Note that T_i (resp., T_c) is set to the value of E when the most recent $N \cdot p_I^{opt}$ (resp., $N \cdot p_C^{opt}$) events in the event window are idle (collision) and the remaining events are success. In the idle state, τ must be increased and in the collision state, τ must be decreased.

$\tau \leftarrow \tau_{init}$; $\tau_{max} \leftarrow 0$; $j \leftarrow 0$
loop
  if $E \geq T_i$ **then**
    if $\tau + \delta_{min} \leq \tau_{max}$ **then**
      {Binary Search Phase}
      $\tau \leftarrow \frac{\tau + \tau_{max}}{2}$
    else
      {Exponential Increase Phase}
      $\tau \leftarrow \alpha \cdot 2^j + \tau$; $j \leftarrow j + 1$
    end if
  else
    if $E \leq T_c$ **then**
      {Multiplicative Decrease}
      $\tau_{max} \leftarrow \tau$; $\tau \leftarrow \beta \cdot \tau_i$; $j \leftarrow 1$
    end if
  end if
end loop

Fig. 3. Transmission probability control algorithm

To guarantee fast convergence to τ_{opt}, we use a binary search algorithm based on E. Suppose that when a node determines $E < T_c$, its current value of τ is assigned to τ_{max}. It is certain that τ_{opt} should be below τ_{max}. The node reduces τ by a multiplicative amount: $\tau = \beta \cdot \tau$ where $\beta < 1$ is a decreasing factor. If the node finds that $E > T_i$, then the target transmission probability (τ_{opt}) must be somewhere in between τ_{max} and current τ. So the node performs binary search by setting its τ to $\frac{\tau + \tau_{max}}{2}$. When τ becomes very close to τ_{max} (the difference is less than a small constant, δ_{min}), it implies that τ_{max} is less than τ_{opt}. The node starts probing for a new value of τ_{max} by exponentially increasing τ until the state changes to collision. A similar control algorithm is used in a recent version of TCP called, BIC [19], and proven to have fast and stable convergence to the target value. Fig. 3 shows the algorithm specification.

3) Dynamic Adjustment of Active Transmitters: For the flexibility to adapt to various network topologies and conditions, a transmitting node may adjust the number of its transmitters to use for packet transmission. If a node finds the channel is not too busy which can be determined by a large value of τ, it can use more transmitters for transmission. Otherwise, it may use less transmitters. We use a simple algorithm for this. Each node maintains a single transmission queue of packets for each transmitter which becomes active with a random uniform probability τ. An active transmitter takes a packet from the transmission queue and transmits the packet. Therefore, when the channel is mostly idle (with large τ) and enough packets are in the transmission queue, the node performs exclusive transmissions.

D. Synchronization

Decoding concurrent frames at a receiver has two practical challenges; timing mismatch and carrier frequency offset. The former is incurred because of random hardware delays at

senders. It results in the misalignment of frames. Sender synchronization techniques such as [20] can solve the problem. The different distances to the receiver is another source of the timing mismatch. However, if amount of the mismatch is smaller than the guard time of an OFDM symbol, it does not degrade the signal detection performance [21], [22]. For example, the guard time specified in the IEEE 802.11n specification is set to $0.8\mu s$ which is sufficiently large to handle any mismatch occurring within a range of $250m$. Since most ad hoc network scenarios consider an interference area within a radius of $250m$ [23], time synchronization between concurrent data streams can be easily guaranteed. The timing mismatch can be also relaxed by use of OFDM cyclic prefix.

The latter can be solved by two different methods. First, a receiver broadcasts a control packet for transmitters to force them adjust the carrier frequency offsets (CFO) [24]. However, this method cannot be used for ad hoc networks because two different transmissions destined to different receivers do not necessarily have the same offset. Second, multiple CFOs from different transmitters can be effectively compensated at a receiver without any control messages by using specialized training structures and additional computations such as ML estimation or PN sequence despreading at the receiver [25], [26]. In Contrabass, gold sequences are used for simultaneous channel training and they are also useful for estimating multiple CFOs. Therefore, we adopt the second method for frequency synchronization.

IV. GNU Radio Experiment

We implement the PHY protocol of Contrabass on the GNURadio platform [8]. Our purpose is to verify that our protocol can be implemented in real systems and can correctly function in a real signal scattering environment. The Universal Software Radio Peripheral (USRP) is used for the evaluation [27]. We form a MIMO link with 4 nodes, each containing two RFX2400 daughter boards operating in the 2.5 GHz range. The experiment is conducted indoor and the setup is shown in Figure 4. Two nodes (3 and 4 in the figure) are transmitting and the other two nodes (1 and 2) are acting as receivers. The two transmitting nodes use only one antenna for transmission, but the receiving nodes use both antennas for receiving.

We use the binary phase shift keying (BPSK) for modulation and use the default GNURadio configuration for DAC ($128e6$ samples/s), interpolation rate (128), ADC ($64e6$ samples/s). We use a 32-bit preamble (31 gold training sequence plus one bit padding), and a 1000 byte payload and 32-bit CRC. As the channel state might drift in the middle of a frame transmission, we perform a periodic channel matrix adjustment by inserting a pilot sequence (the same gold sequence as in the preamble) in every 64 bytes of the payload. At the receiver side, the wave form from the PHY layer is dumped to a log file in the PC, connected via USB interface to the receiving GNURadio nodes. This log file is analyzed offline using MATLAB where we perform timing synchronization, channel training, interference cancelation and payload decoding. Since GNURadio cannot implement carrier sensing without modification in FPGA, we did not implement a real-time MAC protocol, but instead, we force both transmitters to transmit at the same time.

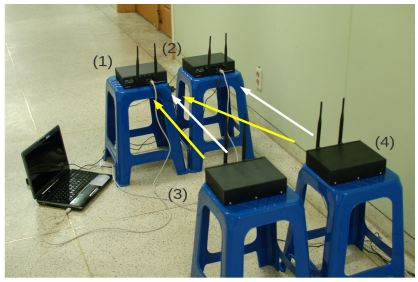

Fig. 4. Contrabass GNURadio experiment setup. Two nodes are set up to be transmitters and the other two nodes are receivers. Each node have two antennas and a transmitter uses one antenna for transmission and a receiver uses both antennas for receiving. A PC connected to radios via USB interfaces controls the experiment, and receives the dumped wave forms from the radio and performs offline signal processes.

A. MIMO Decoding and Interference Cancelation

In this experiment, we measure the performance of Contrabass under two different scenarios. We first test the scenario where two different transmitters send frames to the same receiver with two antennas (nodes 3 and 4 sending to node 1 in Figure 4). This scenario is called *MIMO decoding* scenario. In this scenario, the receiver builds two different channel filters using RIC, RLS and SIC. We also test another scenario where two transmitters send to two different receivers (nodes 3 and 4 sending to nodes 1 and 2, respectively). This scenario is called *interference cancelation* scenario. In this scenario, using RLS, a receiver builds only one channel filter for the signals coming from the intended transmitter and treats the signals from the other transmitter as unknown interference. In both experiments, each transmitter sends frames 500 times, each with a different transmission power. We measure the bit error rates (BER) over received SNR. Figure 5 shows the result of the experiment. We compare the performance of Contrabass under these scenarios with the performance of a single stream with one or two receiving antennas. The performance of two streams to one receiving antenna is used as the baseline which shows that all the frames are un-decodable (with 0.5 BER). Our experiment is performed without any channel coding and 1% BER in an un-coded system is considered acceptable.

The performance of MIMO decoding is comparable to the single streaming performance with one or two antennas up to 12 dB SNR. For SNR higher than 12 dB, its performance drifts a little away from the single stream/two antenna case, but still remains comparable to the one antenna case. The performance of interference cancelation is worse than that of MIMO decoding because it uses only one channel filter treating the other signal (from node 4) as interference and performs interference suppression. But its performance is still within acceptable ranges compared to the performance of single stream cases.

B. Timing Synchronization

We measure the capability of MIMO decoding under timing differences when two transmitters send from different distances to a receiver. We vary the amount of timing offsets

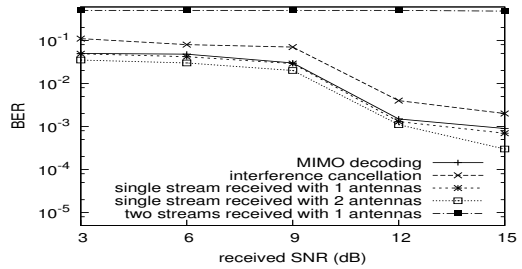

Fig. 5. BER performance of MIMO decoding and interference cancelation.

manually by time-shifting the wave form at one of the transmitters. The receiver receives the superimposed symbols as shown in Figure 6 and oversamples the incoming symbols 7 times instead of using 2 times to improve the performance of MIMO decoding under timing difference. In this experiment, we measure the *success ratio* of MIMO decoding - when a frame is received without any CRC error, we say it is a success. We verify that with $1.14\mu s$ offset, the MIMO decoding of Contrabass achieves about 90% success rate and the success rate drops to 70% with $2.28\mu s$ offset. $1.14\mu s$ offset is possible when the two transmitters (after synchronizing at the boundary of channel idle through carrier sensing) are 342 meters apart. These results indicate that Contrabass is able to handle a reasonable amount of timing errors arising from transmitters at different distances in rich scattering environments.

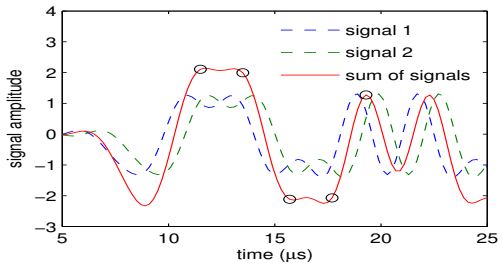

Fig. 6. The receiver performs oversampling and perform the PHY takes the optimal sampling index to resolve the timing error of multiple frames.

V. Simulation

A. Simulation Setup

GNU radio does not permit effective implementation of real-time carrier sensing [9]. Thus, for the network level evaluation, we have implemented Contrabass and several existing MAC protocols in NS-2. Among the concurrent transmission protocols, we choose DMUMSS [4] and SPACE-MAC [6] for the comparison. DMUMSS is the most recent distributed scheduling-based protocol that has been shown to outperform many prior protocols. When implementing DMUMSS, we use 8-bit spreading coding for the concurrent transmissions of RTS and CTS. DMUMSS assumes that a node has the full knowledge of the active neighbors and their traffic load prior to each transmission. Hence we allow DMUMSS to exchange Hello messages and to obtain the information. SPACE-MAC is a unique protocol that allows concurrent transmissions to occur in the middle of other on-going transmissions using precoding technique. We modified basic CSMA/CA such that nodes can transmit and receive packets during transmission of other nodes. We also implement the MAC feature of IEEE 802.11n, adding the frame aggregation technique (A-MSDU) to the existing NS-2 implementation of IEEE 802.11. In our implementation, a node picks several packets with the same destination from its transmission queue. Then it aggregates the packets into one single frame. The receiver returns a block ACK for the successfully decoded packets.

Number of antennas	4
Basic data rate	6 Mbps
RLS training sequence length	31 symbols
MMSE training sequence length	4 symbols
Maximum aggregated frame size	8000 bytes
RTS CTS	14 bytes
Routing protocol	AODV
Size of area	100m x 100m
Transmission range	30m
Data transmission rate	90 Mbps for concurrent tx 180 Mbps for exclusive tx
Channel	Rayleigh fading

TABLE I
Simulation parameters

In most experiments, we populate 100 nodes within a square of 100 m by 100 m area. We set the transmission power such that communication range is 30 meters in the Rayleigh fading channel. In this scenario, hidden terminals are possible. The nodes operate in the ad hoc mode and AODV is used for routing [28]. We derive the data transmission rate of MIMO nodes from [29]. The data rate for concurrent transmission is set to 90 Mbps per antenna and 180 Mbps in total for exclusive transmissions. The simulation parameters are listed in Table I and they are derived from the IEEE 801.n standard [1].

We test Contrabass under three types of traffic: web, FTP and VoIP. For the web traffic, we implement a *web user model* in which each user sends an HTTP request and the server returns a response. After receiving the response, the user sleeps for a random think time. We use the data from a measurement study [30] to select the realistic input distributions. The request size is a bimodal distribution of 100 bytes for 75% and 1500 bytes for 25%. We use Pareto distributions for response file sizes and think time. We set the mean file size to 8800 and the slope to 1.2. When the FTP traffic is used, a node continually downloads a large file from randomly selected users. Both the web and the FTP models run on top of TCP. A VoIP source models the pulse-code modulation (PCM) scheme with sending rate of 64 Kbps and 128 byte packets. We construct a VoIP user model where each user makes a call to a random receiver. The call duration follows a lognormal distribution whose parameters are set following data from a VoIP measurement study [31]. After the call terminates, the user waits for a random think time and again makes another call to a random receiver. VoIP data are always bi-directional. VoIP traffic is delivered using UDP.

B. Performance Evaluation

The web traffic result is plotted in Figure 7 (a). We adjust the mean think time of each user from 0.5 seconds to 0.1 seconds to vary the web traffic load. Contrabass shows very good scalability maintaining its performance with high offered load. The reason is that Contrabass effectively amortizes protocol overhead by minimizing control overhead. The other

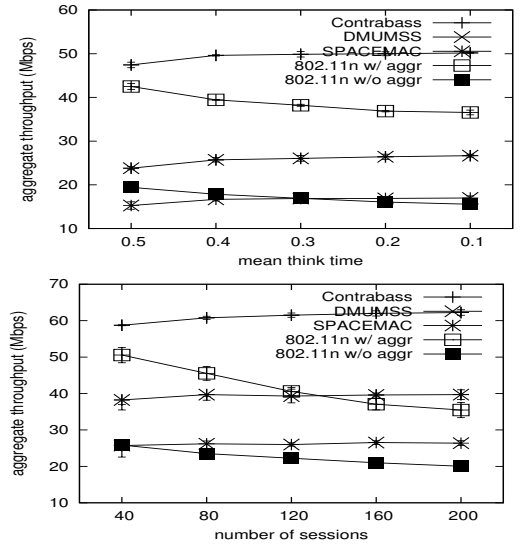

Fig. 7. The aggregate network throughput when the traffic load is varied. (a) Web traffic. (b) FTP traffic

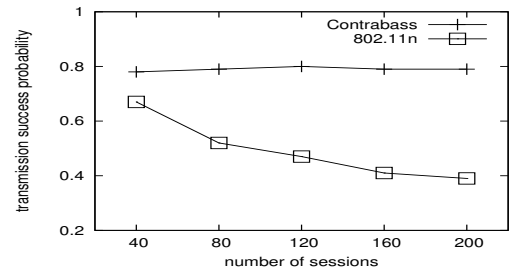

Fig. 8. Transmission success probability of Contrabass and 802.11n.

concurrent transmission protocols incur significant control overhead for the coordination. While 802.11n shows very good throughput under low traffic load, as the traffic load increases, its performance drops because of contention.

Figure 7 (b) shows the aggregate throughput when the FTP traffic is used. Contrabass again outperforms other protocols. Unlike the web traffic scenario where the traffic is very bursty and hence the contention between the transmitters is naturally mitigated, all the nodes keep sending packets with FTP traffic. Note that the performance of Contrabass is rarely affected by the increased contention. The reason is that Contrabass has a very high transmission success probability even with the large number of active transmitters. To illustrate this point, Figure 8 shows the transmission success probability of protocols. The transmission success of Contrabass is around 80% even when the number of transmitters is 200. This indicates that the transmission probability control algorithm of Contrabass works very effectively. On the other hand, 802.11n protocol experience severe performance degradation due to increased packet collisions.

The performance of SPACE-MAC and DMUMSS is almost independent of contention since these protocol exchange control messages to keep the total number of subsequent data transmissions under the limit of receiving antenna numbers. Hence all transmissions are successful unless channel error occurs due to fading. But their performance is poor because of the heavy control overhead incurred by the RTS/CTS

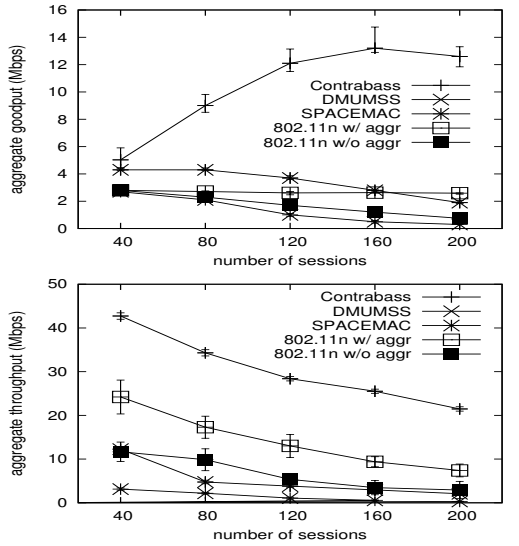

Fig. 9. The result of the VoIP experiment. (a) VoIP goodput. (b) Aggregated throughput of the background FTP traffic

message exchanges. DMUMSS encodes these messages with spreading codes and incurs about 224 byte overhead for the control messages exchange. Moreover these control messages are sent at the basic rate. Contrabass uses the basic rate for the transmissions of preamble and ACK. SPACE-MAC does not incur as much overhead. But its maximum possible concurrent transmissions are only two packets regardless of the number of receiving antennas.

We test VoIP performance under loaded conditions, adding FTP as the background traffic. The number of VoIP sessions is varied. The goodput of VoIP traffic measures the throughput of packets received within deadline (200ms in PCM) after their departure from the application. Figure 9 (a) shows the VoIP goodput. The performance of Contrabass scales extremely well to the traffic load. While the performance of other protocols begins to drop as soon as the load increases, Contrabass continues to increase its goodput until the number of sessions reaches 160. This indicates that Contrabass effectively utilizes the remaining capacity without much overhead. Both background and VoIP traffic under Contrabass achieve much higher throughput and goodput than the other protocols. The high goodput of VoIP also comes from low delay arising from low packet loss rates due to collision and absence of any control messages (e.g., RTS and CTS). We measured the average delay of successfully delivered packets in this scenario in Figure 10.

The 802.11n frame aggregation technique does not effectively improve the performance of VoIP traffic. This is because VoIP data arrive at a fixed interval and there are not many packets to aggregate at each instance of transmission. Since VoIP data and background traffic are likely to be destined to different receivers, aggregation is not activated most of time. In this simulation, an 802.11n node aggregated average 4.7 packets when the number of active sessions is 40 but it dropped to 1.0 with 200 active sessions. Since aggregation cannot function, the control overhead per VoIP packet (Block ACKs, MAC layer retransmissions, SIFS and DIFS) is fully accrued.

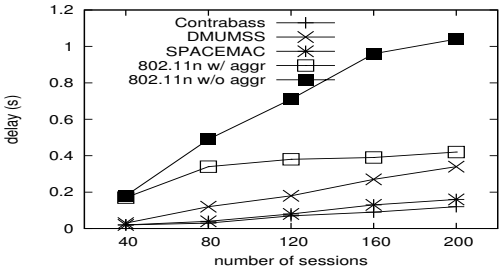

Fig. 10. Average packet delay of protocols.

VoIP packets are very small and the relative control overhead is substantially large. This effect is also clearly visible from the DMUMSS performance as it has the largest control overhead.

In Figure 9 (b), the aggregate throughput of the background FTP traffic is plotted. It is well known that CBR traffics harm the performance of TCP based applications [32]. The reason is that TCP senders continuously reduce their c ongestion window sizes when the packet is lost, while CBR traffic maintains its sending rate, finally filling up the network capacity. Contrabass experiences performance degradation as well, but the problem is considerably alleviated as TCP packet loss rate is much lower than using the other protocols.

VI. Conclusion

Contrabass implements the first-to-date open-loop based MIMO PHY/MAC protocol for concurrent transmissions. Its operation is tuned toward ad hoc network environments. It employs an innovative combination of novel and existing PHY layer MIMO techniques to implement simultaneous channel training for effective interference cancelation and MIMO decoding for concurrent transmission. This results in elimination and minimization of coordination and control overhead. The MAC protocol of Contrabass harmonizes with the PHY layer protocols to implement the required synchronization for simultaneous channel training. More important, although being completely open-loop with no handshaking such as RTS and CTS, it can effectively adjust its transmission probability to achieve near-optimal throughput performance with low delay. Our design choices are deliberately made to enable practical implementation which is demonstrated through GNURadio implementation and experimentation. Being open-loop, we admit that Contrabass might be susceptible to hidden terminal problems, but this is a deliberate design choice considering the tradeoff between overhead and performance degradation due to hidden terminals. It is known that hidden terminals are not very common and even in the IEEE 802.11 system, RTS/CTS are commonly turned off.

Acknowledgement : This work is supported by NSF under grant CNS-0910868 and CNS-1016216.

References

[1] "IEEE standard 802.11n," Nov 2009.
[2] "Fcc 47 cfr ch. i (10105 edition)," http://www.fcc.gov.
[3] K. Sundaresan, R. Sivakumar, M. Ingram, and T.-Y. Chang, "A fair medium access control protocol for ad-hoc networks with mimo links," in *Proc. of IEEE INFOCOM*, 2004.
[4] S. Chu and X. Wang, "Opportunistic and cooperative spatial multiplexing in mimo ad hoc networks," in *Proc. of MobiHoc*, May 2008.
[5] Mundarath, J. Ramanathan, P., V. Veen, and B.D., "Nullhoc : a mac protocol for adaptive antenna array based wireless ad hoc networks in multipath environments," in *Proc. of Global Telecommunications Conference*, 2004.
[6] J.-S. Park, A. Nandan, M. Gerla, and H. Lee, "SPACE-MAC: Enabling spatial reuse using mimo channel-aware mac," in *Proc. of IEEE ICC*, 2005.
[7] P.Casari, M. Levorato, and M. Zorzi, "DSMA: an access method for mimo ad hoc networks based on distributed scheduling," in *Proc. of ACM IWCMC*, 2006.
[8] "GNURadio - the development toolkit for software-defined radio," http://gnuradio.org.
[9] G. Nychis, T. Hottelier, Z. Yang, S. Seshan, and P. Steenkiste, "Enabling mac protocol implementations on software-defined radios," in *Proc. of NSDI*, April 2009.
[10] M. Park, R. W. H. Jr., and S. M. Nettles, "Improving throughput and fairness of mimo ad hoc networks using antenna selection diversity," in *Proc. of IEEE Globecom*, 2004.
[11] T. Tang, M. Park, R. W. H. Jr., and S. M. Nettles, "A joint mimo-ofdm transceiver and mac design for mobile ad hoc networking," in *Proc. of International Workshop on Wireless Ad-Hoc Networks*, 2004.
[12] S. Haykin, *Adaptive Filter Theory, 4th Ed.* Prentice Hall, 2001.
[13] D. Halperin, T. Anderson, and D. Wetherall, "Taking the sting out of carrier sense: Interference cancellation for wireless lans," in *Proc. of ACM MobiCom*, 2008.
[14] P. Patel and J. Holtzman, "Analysis of a simple successive interference cancellation scheme in an ds/cdma system," *IEEE Journal on Selected Areas in Communications*, 1994.
[15] R. Gold, "Optimal binary sequences for spread spectrum multiplexing," *IEEE Transactions on Information Theory*, 1967.
[16] J. Wang and B. Daneshrad, "A comparative study of mimo detection algorithms for wideband spatial multiplexing systems," in *Proc. of IEEE Wireless Communications and Networking Conference*, 2005.
[17] S. Gollakota and D. Katabi, "Zigzag decoding: Combating hidden terminals in wireless networks," in *Proc. of ACM SigComm*, 2008.
[18] G. Judd, "Using physical layer emulation to understand and improve wireless networks," in *Ph.D Dissertation, Carnegie Mellon University*, 2006.
[19] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control (bic) for fast long-distance networks," in *Proc. of IEEE INFOCOM*, 2004.
[20] H. H. Hariharan Rahul and D. Katabi, "Sourcesync: A distributed wireless architecture for exploiting sender diversity," in *Proc. of ACM SigComm*, 2010.
[21] J. Heiskala and J. Terry, *OFDM Wireless LANs: A Theoretical and Practical Guide*. SAMS Publishing, 2002.
[22] K. Fazel and S. Kaiser, *Multi-Carrier and Spread Spectrum Systems*. Wiley, 2003.
[23] A. Vahdat and D. Becker, "Epidemic routing for partially-connected ad hoc networkepidemic routing for partially-connected ad hoc networks," in *Technical Report CS-2000-06 Duke University*, 2000.
[24] E. Eraslan, V. Panchagnula, B. Daneshrad, S. Yoon, I. Rhee, and J. H. Kim, "Signal processing techniques to enable concurrent communications in mimo enabled networks," in *Proc. of IEEE Asiloma Conference on Signal Systems, and Computers*, 2009.
[25] Y. Zeng, A. Leyman, and T.-S. Ng, "Joint semiblind frequency offset and channel estimation for multiuser mimo-ofdm uplink," *IEEE Transactions on Communications*, 2007.
[26] Y. Wu, S. Attallah, and J. Bergmans, "Carrier frequency offset estimation for multi-user mimo-ofdm uplink using cazac sequences," in *Proc. of IEEE WCNC2009*, 2009.
[27] "USRP - universal software radio peripheral," http://www.ettus.com/.
[28] "RFC3561 - ad hoc on-demand distance vector (aodv) routing," 2003.
[29] H. Jin, B. C. Jung, H. Y. Hwang, and D. K. Sung, "Performance comparison of uplink wlans with single-user and multi-user mimo schemes," in *Proc. of IEEE WCNC2008*, 2008.
[30] B. A. Mah, "An empirical model of http network traffic," in *Proc. of IEEE Infocom*, 1997.
[31] J. GUO, F. LIU, and Z. ZHU, "Estimate the call duration distribution parameters in gsm system based on k-l divergence method," in *Proc. of WiCom*, 2007.
[32] P. Verkaik, Y. Agarwal, R. Gupta, and A. C. Snoeren, "Softspeak: Making voip play well in existing 802.11 deployments," in *Proc. of NSDI*, 2009.