802.11ec: Collision Avoidance without Control Messages*

Eugenio Magistretti
ECE Dept.
Rice University
Houston, TX
emagistretti@rice.edu

Omer Gurewitz
CSE Dept.
Ben Gurion University
Beer Sheva, Israel
gurewitz@cse.bgu.ac.il

Edward W. Knightly
ECE Dept.
Rice University
Houston, TX
knightly@rice.edu

ABSTRACT

In this paper, we design, implement and evaluate 802.11ec (Encoded Control), an 802.11-based protocol *without* control messages: instead, 802.11ec employs correlatable symbol sequences, which together with the timing the codes are transmitted, encode all control information and change the fundamental design properties of the MAC. The use of correlatable symbol sequences provides two key advantages: *(i)* efficiency, as it permits a near order of magnitude reduction of the control time; *(ii)* robustness, because codes are short and easily detectable even at low SINR and even while a neighbor is transmitting data. We implement 802.11ec on an FPGA-based software defined radio. We perform a large number of experiments and show that, compared to 802.11 (with and without RTS/CTS), 802.11ec achieves a vast efficiency gain in conveying control information and resolves key throughput and fairness problems in the presence of hidden terminals, asymmetric topologies, and general multi-hop topologies.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design- Wireless Communication

General Terms

Design, Experimentation, Measurement, Performance

Keywords

WLANs, 802.11, Channel Access, Collision Avoidance, Control Messages, RTS/CTS, Signal Correlation

*While we employ the name 802.11ec, our work does not represent an IEEE standard. We use this name for reasons that will be apparent in the body of the paper.

MobiCom'12, August 22–26, 2012, Istanbul, Turkey.

1. INTRODUCTION

MAC control messages are essential: for example, ACKs convey correctly received data and RTS/CTS exchange can significantly mitigate hidden terminal collisions. However, even though the information conveyed in MAC control messages is small, their duration can be quite long, as in addition to the control information, they also need to include source/destination address, message type, etc., all of which are transmitted at base rate to improve the likelihood that they can be successfully decoded. For example, an ACK message is 14 bytes plus physical layer encapsulation, but contains only one bit of relevant information (that DATA was successfully received). Likewise, RTS/CTS is rarely used in practice precisely due to excessive overhead despite its important role in mitigating collisions.

In this paper, we design, implement, and evaluate 802.11ec (Encoded Control) as a control message free MAC. Instead of control messages, 11ec employs correlatable symbol sequences (CSS's), which together with their transmission timing, convey all control information, and change the fundamental design properties of the MAC. For example, 11ec replaces an 802.11 ACK *message* with a predefined ACK CSS that can be correlated instead of decoded, thereby vastly reducing its duration and dramatically improving its robustness by enabling its reception at low SINR.

Control information can be classified along two dimensions: first, as to whether or not the information in the message can be represented from a small dictionary or codebook. For example, a small dictionary can encode the three different control messages used in 802.11 for data exchange (RTS, CTS, and ACK). Likewise, while the space of all MAC addresses is large (seemingly precluding a small dictionary), each node communicates with only a limited number of addresses at a time. Thus, both MAC addresses and control message type can be encoded from a small dictionary. Second, control information can further be classified according to whether they are necessarily public or can be private. For example, for correctness of the protocol, all nodes must know that a CTS should cause them to defer, i.e., this message must be public; on the other hand, only a data sender need know that its data was correctly received, i.e., its ACK may be private.

802.11ec's key techniques are two fold: first, we use a dictionary of correlatable symbol sequences to convey control information that can be represented by a limited dictionary. For example, instead of CTS that contains physical layer preamble, frame control sequence, type field, frame checksum, destination address, duration field (as well as it incurs

a T_{SIFS} delay), we transmit a short (e.g., 127 symbols) CSS from a small dictionary to convey that it is a CTS. For an 802.11a physical layer, we show that this reduces the time to convey the control information by nearly an order of magnitude, from 60 μs to 6.35 μs. Second, we show that the information that cannot be represented by a limited dictionary can be conveyed via CSS timing. For example, nodes overhearing the 802.11 CTS message need to defer for an amount of time as specified by the CTS duration field contained in the message. We show that 11ec nodes can instead simply defer until a channel-clear CSS is transmitted by the receiver (or until a timeout).

802.11ec's second technique is to distinguish between public and private information. Namely, 11ec only uses public CSS's for information that is required to be public, such as conveying channel reservation and channel clear. On the other hand, address fields need not be public, as the identity of the sender and receiver need not be known by other nodes. 11ec ensures that private control information, including addresses and ACKs, is not correlated by other nodes. This has the potential to thwart eavesdroppers not only from decoding data (as data can be encrypted), but even from knowing which nodes are communicating with each other; we show how all private control information, including addresses can only be correlated by the intended receiver.[1]

802.11ec enhances robustness in two ways. First, control information is more likely to be received in 11ec because control information is conveyed in short CSS's that are correlatable even at low SINR. For example, because 802.11ec replaces an ACK message with a CSS, 11ec ACKs are more robust and can be received even in the presence of transmitting interferers. Second, 802.11 is "fragile" to topological factors in that while 802.11 DCF without RTS/CTS yields high performance in fully connected wireless LANs [5], hidden terminals, asymmetric topologies, and general multi-hop topologies can yield severe throughput degradation and unfairness [6]. These latter topologies are becoming increasingly common because of device power asymmetries, e.g., between APs, laptops, and popular smart-phones, and of the wider coverage achievable with the adoption of sub-GHz frequencies, including TV white spaces [16, 19]. While use of RTS/CTS can significantly improve throughput in such challenged topologies, the additional overhead of RTS/CTS can sometimes overwhelm this improvement. Moreover, in fully connected topologies RTS/CTS degrades throughput due to its unnecessary overhead. In contrast, 11ec overcomes these limitations through robust and short-duration control signals, i.e., 11ec minimally penalizes station throughput thus allowing to enable channel reservation independently of the network topology. Consequently, 11ec stations have vastly increased opportunities to obtain channel access thereby dramatically improving the network's fairness in throughput distribution.

We implement correlatable symbol sequences in a software defined radio, perform a large set of experiments and study issues that have not been experimentally investigated previously. We find a correlatable symbol sequence *length* that simultaneously: *(i)* provides sufficient physical-layer robustness, *(ii)* limits communication overhead, and *(iii)* supports large networks. Specifically, we first investigate the trade-offs between sequence length and physical-layer robustness and show that even short sequences, e.g., 127-symbol or 6.35 μs long, can be detected at -6 dB SINR with only 5% false negatives. We demonstrate that our encoded sequences can be detected at an SINR 10 dB lower than 802.11 control messages. Second, we show that 127-symbol code lengths can support more than 50 co-located nodes, with minimal penalty on detection errors.

Finally, we implement 11ec in a measurement-driven emulator, whose inputs are channel measurements collected in a real deployment and real card performance parameters (e.g., BER and multiple supported modulations). We compare 11ec's performance to 802.11 with and without RTS/CTS. We examine a wide set of basic topologies that are at the origin of throughput losses and/or imbalances in 802.11-based networks in order to provide an insight in understanding the performance of larger networks. Our finding is that 11ec can dramatically reduce throughput imbalances by improving the Jain index [12] by up to 88%. Moreover, while such a fairness improvement can often decrease total utilization, 11ec increases channel utilization by more than 10% via the use of short encoded control that simultaneously decreases vulnerability intervals and control overhead. We also study a larger topology and show that 802.11ec can improve the throughput of an under-served flow by a factor of 1255%. Over all flows, we improve Jain index by up to 217% while also improving the channel utilization by up to 44%.

The remainder of the paper is organized as follows. Section 2 discusses coded control CSS's and the design of 802.11ec. Section 3 includes a thorough experimental evaluation of CSS's using a software defined radio platform. Section 4 investigates the benefits of 802.11ec in a measurement-driven emulator. Finally, Sections 5 and Section 6 overview related works and conclude the paper.

2. MAC PROTOCOL DESIGN

11ec collision avoidance realizes and improves on the collision avoidance mechanism of IEEE 802.11 DCF with RTS/CTS, reduces the overhead by nearly an order of magnitude, and practically eliminates collisions, even in hidden terminal topologies. Specifically, 11ec retains the four-way handshake suggested by 802.11, where control messages are replaced by very short correlatable symbol sequences (see Figure 1). It is important to note that: *(i)* the duration of each correlatable symbol sequence is nearly zero; *(ii)* the duration between the start of the reservation signal until the data transmission is negligible, hence practically invulnerable to collisions.

In this section, we define correlatable symbol sequences (CSS's) and explain how control messages can be turned into CSS's. Furthermore, we show that CSS's are a key element for the realization of 11ec efficiency and robustness. The second part of the section provides a detailed description of 11ec including protocol primitives, and an analysis of hidden terminal vulnerability leading to novel collision reduction opportunities.

2.1 Coded Control Information versus Message Control Information

CSS's. Correlatable symbol sequences are predefined pseudo-noise binary codewords; namely, while codewords are deterministically generated, they retain the statistical properties of a sampled white noise. For this reason, the cross-correlation of any such sequence with a matching copy ob-

[1]While development of a complete privacy protocol is beyond the scope of this work, 11ec provides important mechanisms to design such a protocol.

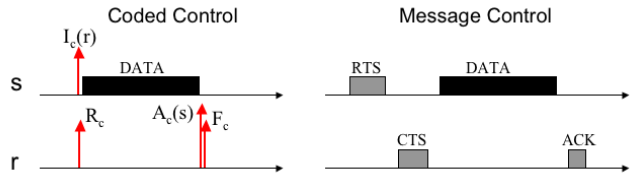


Figure 1: Timeline of a packet exchange with Coded Control versus Message Control.

tains spike values, while it appears random to a listener without prior knowledge of the codeword. An example of a CSS is the 802.11 preamble used for packet detection, symbol synchronization, and radio parameter tuning.[2]

The CSS detection process via cross-correlation enjoys three key advantages over data decoding. First, cross-correlation obtains a large processing gain even for small codewords (e.g., the 802.11 preamble used for detection is 64 symbols), which permits reliable detection even at low SINR. Second, differently from decoding, detection is highly robust to imperfect radio parameter tuning and thus a codeword does not need to be preceded by a preamble. For these reasons, a CSS can be short; for example, in our implementation, 11ec utilizes 127-symbol codewords that can be transmitted in 6.35 μs. Third, detection is almost instantaneous as no decoding is needed. For example, 802.11 inter-control message time is at least $T_{SIFS} = 16$ μs, including about 14 μs of data processing, while in 11ec no control message processing is required; hence, 11ec reduces substantially the short inter-CSS time.

Encoded Control. While consuming a significant amount of airtime, 802.11 control messages usually convey little information. For example, an ACK occupies the medium for up to 60 μs, i.e., an airtime sufficient to transmit 3240 bits at 54 Mbps, while it contains a single bit of relevant information. 11ec replaces control messages with CSS's, which permit to shorten the transmission duration of nearly an order of magnitude to 6.35 μs, while retaining the information content.

According to the 802.11 standard [1], RTS, CTS, and ACK control messages may include up to four information fields: destination address, sender address, duration, and frame control (a fifth field is the frame check-sum that protects the other four). In particular, the frame control field is a 2-byte long sequence of bits representing specific control parameters. The values of most control bits are fixed for control messages; only frame subtype (4 bits) and station power management flag (1 bit) can assume different values. However, the latter does not convey novel information when used in control messages.

In order to represent the information content of the control messages as described above, 11ec considers the size of the dictionary needed to represent such information. Information that can be expressed by a small dictionary is conveyed using CSS's, while information that needs large dictionaries is conveyed with timing codes. First, in 802.11 data exchange, the type field that distinguishes the control messages may assume only three values, i.e., RTS, CTS, and ACK; 11ec conveys the type by associating each message with distinct CSS's. Second, 802.11 control messages include addresses (sender and/or receiver). Since the number of nodes a station communicates with at a time is generally small, i.e., can be represented by a small dictionary, 11ec integrates the addresses in CSS's, i.e., a single CSS may represent the combination of a control type and a specific address. For example, in 802.11 the RTS includes the address of the intended receiver; accordingly, in 11ec, RTS addressed to different receivers are represented by distinct CSS's. Third, some control messages include a duration field that cannot be represented by a small dictionary. For this information, 11ec utilizes a combination of time codes and new control types. For example, 11ec nodes reserve the channel for the duration of a data reception, by transmitting two CSS's corresponding to channel reservation (immediately before the reception) and release (immediately after). The potential interferers do not access the channel during the interval between the two CSS's (or before a timeout expires), i.e., effectively implement a form of virtual carrier sensing.[3]

Public CSS's versus Private CSS's. A second dimension of control messages, and of the corresponding CSS's, is whether they can be private, i.e., carry information relevant only to a specific destination (e.g., acknowledgements), or are necessarily public, i.e., meant to be heard by all neighbors of a node (e.g., channel reservation/release). Accordingly, in the case of a private CSS, only the intended receiver possesses a copy of the correlatable symbol sequence, and thus can correctly detect it; conversely, all nodes possess copies of the public CSS's and can detect them. For example, only the data sender needs to cross-correlate its private acknowledgement CSS, while all nodes must cross-correlate public channel reservation/release CSS's.

Control during Data. Because correlatable symbol sequences can be detected even at sub-noise SINR (e.g., 11ec 127-symbol CSS's can be detected at -6 dB with high reliability), 11ec nodes attempt detection even while receiving data. This technique provides a signaling mechanism effective even in cases in which the receiver is subject to long periods of noisy channel, or undesired data overhearing. In 802.11, if a node receives an RTS while also overhearing data, the node cannot decode the RTS and therefore cannot respond. In contrast, 11ec uniquely enables a node to receive a CSS signaling that another node is requesting to communicate. Therefore, the receiver can send a Request for RTS (RRTS) CSS to reserve the medium when it becomes free and initiate a data exchange [4]. Likewise, because ACKs are also encoded, they can be correctly received even if an interfering terminal is simultaneously transmitting data.

2.2 11ec Channel Reservation Primitives

Wireless MAC protocols perform collision avoidance by silencing the medium in the vicinity of a transmitting link via channel reservation. Channel reservation fundamentally hinges on three key mechanisms: *(i) initiation*, performed by the node that initiates the exchange to request the cooperation of the other endpoint to reserve the channel; *(ii) reservation*, performed to inform nodes potentially hindering the exchange; and *(iii) deferral*, performed by the surrounding terminals in order to avoid disturbing ongoing transmissions. 802.11 implements the three mechanisms via *(i)* RTS; *(ii)*

[2]A detailed discussion of signal correlation can be found in literature [10, 14, 22]. A short introduction is in Appendix A.

[3]Another alternative is to discretize the duration and associate different CSS's to each discrete value.

CTS and data packet - the latter realizes channel reservation in the vicinity of the sender; and *(iii)* NAV and carrier sensing. When RTS/CTS is disabled, during the data transmission the medium is reserved exclusively in the vicinity of the sender. In the following, we show how 11ec implements these three mechanisms via CSS's and timing codes.

A key concept of 11ec channel reservation is very short channel reservation negotiation for near immunity to interruptions, e.g., collisions and capturing by other nodes. Specifically, 11ec channel reservation is based on three basic primitives (the subscript c indicates CSS's).

Initiation: $\mathbf{I_c(r)}$. In 11ec, a sender wishing to start a data exchange performs virtual, and optionally physical, carrier sensing. If the medium is free, the sender waits for a backoff interval similar to 802.11 and then transmits a sender side channel request primitive, in short $I_c(r)$, to request the receiver r to reserve the channel. $I_c(r)$ need only be detected by r and not necessarily by the neighboring nodes; thus, we implement it as a private CSS. In order to convey the identity of the receiver r (as 11ec does not transmit the traditional MAC address), 11ec implements $I_c(r)$ via several CSS's, and associates a distinct CSS with each receiver; i.e., when a sender needs to contact a receiver, it uses the receiver's $I_c(r)$. Nodes in the vicinity of the sender do not detect the initiation $I_c(r)$.

Reservation: $\mathbf{R_c}$. A node r receiving an $I_c(r)$ checks if other nodes are communicating in its vicinity and may hinder its reception. If that is not the case, r immediately transmits a channel reservation primitive R_c to notify potential interferers. In order to realize the reservation, R_c should be detected by all nodes in the vicinity of the receiver r and it is therefore transmitted via a public CSS.

In addition to channel reservation that forces neighbors to defer, R_c implicitly communicates to the transmitter that the channel is available and that data can be transmitted. Instead of providing a distinct CSS to convey the sender address, as in the previous case of the $I_c(r)$, we employ a simple temporal code: Since a receiver r transmits R_c immediately after $I_c(r)$, the sender, in contrast to the other neighboring nodes, interprets the reception of a R_c as an authorization to begin a data transmission.

Deferral: $\mathbf{R_c \rightarrow F_c}$. 11ec implements the deferral and conveys its duration via a combination of CSS's and a simple time code. Specifically, after data reception and acknowledgement, the receiver explicitly releases the channel with a channel free primitive F_c. Thus, nodes receiving R_c need to wait to receive an F_c (or wait a predefined timeout) before accessing the channel; practically, this procedure represents a form of virtual carrier sensing. Because all neighboring nodes need to receive F_c, 11ec implements F_c as a public CSS.

11ec further increases the robustness of R_c/F_c messages by pairing them. Our technique is based on associating a small number of CSS pairs to distinct R_c^i/F_c^i pairs; a receiver randomly picks and transmits any such pair to reserve and free the channel. This feature is particularly useful for a node located in the neighborhood of several receivers that may be active simultaneously, i.e., the receivers may send R_c and F_c that denote overlapping intervals. In that case, the node can still correctly determine the state of the medium in each moment, by associating each overheard reservation R_c^i to its corresponding F_c^i.

Finally, we define an acknowledgement primitive, $A_c(s)$ (see Figure 1), and we implement it as a private CSS associated to each sender s. Few recently proposed packet forwarding schemes leverage the ACK exchange for additional purposes other than data acknowledgement, e.g., network coding [7] and routing [8]. In such cases, 11ec can revert to 802.11 ACKs with small overhead penalty (the major gain of 11ec both in throughput and robustness derives from the novel channel reservation scheme). Note that CSMA/CN [22] uses signatures for acknowledgement, which are similar to our CSS's.

CSS Association. The Initiation and Acknowledgement primitives $I_c(r)$ and $A_c(s)$ require association of unique CSS's to specific nodes. While a detailed investigation of the issue is beyond the scope of this paper, we suggest two simple mechanisms that can be used for this purpose. The first leverages the solution already existing in the 802.11 standard, in which the AP assigns an identifier (AID) upon station association [1]; AIDs can be easily mapped to CSS's. In order to initiate the association, the stations may use a reserved CSS. The second mechanism utilizes multiple hash functions based on the station MAC address. In case of assignment conflicts, which can be easily detected using the address fields in the header of the data frames, the stations can switch the hash function utilized.

Primitive Extensions. As briefly mentioned above, 11ec can support a signaling mechanism to alleviate starvation similar to RRTS [4] via control during data, and can highly reduce the occurrence of exposed terminals via robust CSS acknowledgement. For reason of space, these cases are not covered in this paper.

2.3 Contending Flows and Vulnerability Interval

The shortness and robustness of correlatable symbol sequences dramatically reduces vulnerability to collisions. The vulnerability interval of a packet exchange is twice the time delay from the beginning of a transmission until all potential interferers are prevented from corrupting the exchange, i.e., it includes the whole interval, before and after the beginning of the intended exchange, during which interfering transmissions may start and corrupt the exchange. In the case of hidden terminals in **802.11** with RTS/CTS, the vulnerability interval is twice the delay from the moment a node sends an RTS to the detection of CTS by the hidden terminals. Considering the case of 802.11a/g and 6 Mbps control packets, the total duration of the vulnerability interval can be computed as follows. First, node s transmits the RTS, for a total duration of 52 μs including preambles; second, the RTS propagates to the receiver for up to 1 μs; third, the receiving node r waits $T_{SIFS} = 16$ μs before sending the CTS, due to practical communication issues such as RTS decoding; fourth, the CTS propagates to hidden terminals for up to 1 μs; fifth, the hidden terminals need up to 4 μs to detect the packet; last, additional 2 μs account for potential radio turn-around (all temporal indications are taken from section 17 of the 2007 version of the standard, for 20 MHz bandwidth [1]). The total amounts to 152 μs, i.e., twice 76 μs. In 802.11 without RTS/CTS, the vulnerability interval can be considerably larger, spanning twice the data transmission duration and as large as 4 ms.

802.11ec's CSS's shorten the vulnerability interval and practically reduce the set of potential hidden terminals. The vulnerability interval of 11ec can be computed as follows (see

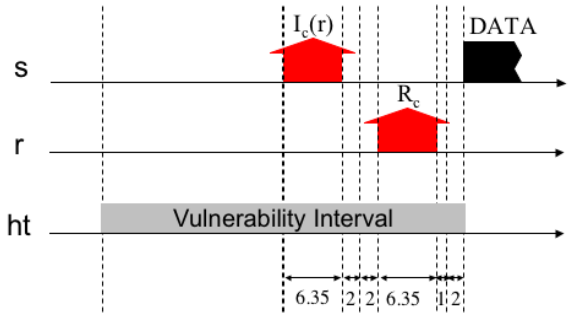


Figure 2: Timeline of the vulnerability interval of 802.11ec (time indications are in μs).

Figure 2, where the intervals below are denoted by numerical time indications). First, node s transmits $I_c(r)$ for 6.35 μs; second, the receiving node r waits for up to 2 μs, in order to detect potential overlapping hidden terminal transmissions as explained below (this interval is a design choice and includes the propagation delay from the transmitter); third, r needs a turn-around time to prepare its radio for transmission, i.e., 2 μs (according to 802.11 standard); fourth, r transmits R_c for 6.35 μs; finally, the hidden terminals need a propagation delay of up to 1 μs to receive the R_c and an additional 2 μs to account for potential radio turn-around. 11ec vulnerable interval is twice 19.7 μs, i.e., 39.4 μs or about 25.9% of the vulnerability interval of 802.11.

CSS's also nearly eliminate the collisions of nodes starting in the same slot when the SINR allows it.[4] In fact, the receiver r can detect simultaneous and overlapping transmissions of multiple $I_c(r)$ because of the CSS processing gain, and request retransmission. Specifically, r waits for a round-trip propagation time in order to detect potentially overlapping transmissions and, in that case, sends a negative acknowledgement which prompts the contending nodes to undergo a quick backoff repetition. However, in order to take advantage of this technique, we need to enlarge the slot size to encompass the half vulnerability duration, i.e., 20 μs (note that this is sufficient, as the F_c of the receiver synchronizes all of its transmitters). Also in cases of no hidden terminals, this choice induces only minor throughput penalties at high data rates, as we show in Section 4.2.

Co-existence with legacy 802.11. For backward compatibility with 802.11, 11ec exactly follows the standard [1] except for the techniques described in this section. For example, a key element for co-existence is the arbitration of the medium, which leverages carrier sensing based on the correlation of the data preambles and backoff mechanism. Accordingly, 11ec uses the same data preamble format as 802.11, and sets the contention window size following the same binary exponential backoff scheme. A more complete discussion of co-existence is beyond the scope of this paper.

[4]Nodes may use power adaptation techniques exclusively to transmit control CSS's while transmitting data at full power, i.e., without modulation rate adaptation requirement or throughput penalty. While this may improve the performance of 11ec, we defer its investigation to future work.

3. EXPERIMENTAL EVALUATION OF CSS

In this section we present an experimental evaluation of correlatable symbol sequences using software defined radios. Specifically, our evaluation covers the following issues, *none of which has been previously experimentally studied in the literature.*

(i) We explore the trade-off between length of the sequences, i.e., overhead, and processing gain, i.e., robustness. **Our finding is that 127-symbol sequences provide a good trade-off between overhead (6.35 μs) and robustness (5% false negatives at -6 dB SINR).**

(ii) We contrast the performance of CSS detection with control message decoding. **Our finding is that 127-symbol CSS's can be reliably detected at about 10 dB lower SINR than 802.11 6 Mbps OFDM control packets.**

(iii) We determine the codebook size that 11ec can support, i.e., the number of distinct CSS's that can be practically used, by studying the cross-correlation between different CSS's and its effect on the probability of false positives. **Our finding is that the design of 127-symbol CSS's via Gold codes can support more than 50 co-located nodes (a total of 127 CSS's), without any penalty on false positives.**

3.1 Experimental Setup

3.1.1 Tools

WARP and WARPLab. Our reference software defined radio is the WARP platform [3]. WARP is an FPGA-based platform, including custom designed radios based on the MAX2829 chipset. WARPLab is a programming environment that permits to drive WARP from a host computer. Relevantly to our experiments, WARPLab supports the execution of micro experiments, each one of approximately 400 μs duration (2^{14} samples at the 40 MHz frequency of DAC/ADC), and access analog sample send/receive buffers and RSSI recordings collected during each experiment. RSSI is measured by the MAX2829 circuit, and digitized by a dedicated 10-bit ADC.

Azimuth Channel Emulator. In order to perform experiments under controllable and repeatable conditions, we used an Azimuth ACE MX channel emulator.[5] The channel emulator permits creation of different network topologies, by tuning the attenuation along each path independently and predictably.

3.1.2 Implementation

We implement CSS transmission/detection and OFDM packet transmission/decoding on WARPLab. Specifically, CSS's are BPSK sequences filtered, upsampled, and transmitted via standard wideband methods. This solution enjoys a practical advantage over alternative solutions, e.g., OFDM-modulated BPSK sequences, due to the lower peak to average power ratio [26]. Finally, in order to reproduce 802.11 as closely as possible, we implement all types of OFDM 802.11a/g modulation and convolutional code pairs in WARPLab.

3.2 Channel Emulator Validation

The results in this section are obtained using a channel emulator. In order to validate the emulator setting, our

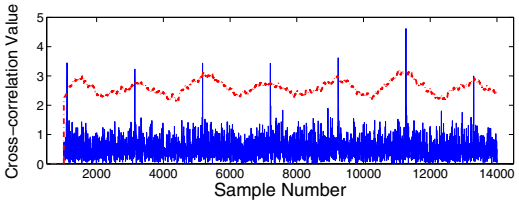
[5]Azimuth Systems Inc., `http://www.azimuthsystems.com/`

methodology includes a preliminary validation contrasting results of a cross-correlation experiment performed over the air, with an identical experiment conducted with the emulator. Specifically, our experiment consists of exchanging CSS's between two WARP nodes a and b, under the interference generated by random OFDM transmissions of a third interfering node c. In the first part of the experiment, we deploy the three nodes inside an office building. In an over the air setting, it is difficult to control SINR given interference from 802.11 networks operating in the building. For this reason, we perform the experiment late at night and measure the SINR on links $a-b$ and $c-b$ a few seconds before and after the experiment. Then, we repeat the experiment under controllable and repeatable conditions using the channel emulator, where we can control the SINR with high accuracy. We repeat both experiments several times and for different SINR, and show a representative sample result.

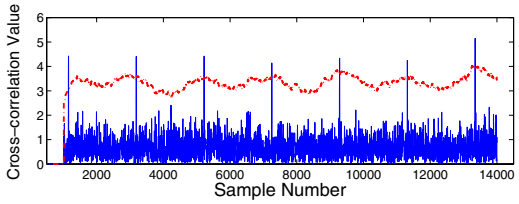
In both experiments, node a transmits 7 repetitions of a 127-symbol CSS. For each newly acquired sample (i.e., at 40 MHz frequency of the WARP platform ADC), node b computes the signal correlation with a local copy of the transmitted CSS. Figures 3(a) and 3(b) show a representative outcome of a realization in which the SINR on link $a-b$ carrying the CSS is -6 dB. The x-axis is the temporal progression of collected samples, while the y-axis is the correlated value. The thick crossing line in the plots represents a possible choice of the detection threshold; such a threshold is strategic in determining the robustness of CSS's, by balancing false positives and false negatives. In the experiments we conduct in this section, the threshold is chosen in order to obtain a false positive probability of 10^{-8}. While we defer more details on how to tune the threshold to Section 3.6, here we observe that because of the threshold design the correlation value on the y-axis is normalized according to the magnitude of correlated I/Q samples. In the figures, correlation spikes are clearly identifiable in coincidence with the reception of each single CSS, as all and only marks exceeding the threshold. Thus, the detection of 127-symbol CSS's is possible at -6 dB with few errors. *By comparing the two plots, we conclude that controllable emulator experiments and over the air experiments provide similar results for identical SINR values.* However, because of the difficulty to constantly control the SINR in over the air settings, we perform the remaining experiments in this section using the channel emulator, thereby also ensuring their repeatability.

3.3 CSS Length versus Robustness Trade-off

The first issue is the trade-off between the CSS length L and robustness under different SINR; we quantify robustness in terms of the probability of false positives and false negatives. The outcome of this assessment is important, as robustness to SINR is one of the two key elements in the choice of CSS length (the other element guiding this choice is the number of CSS's, discussed in Section 3.5). In this subsection, we determine a length that can tolerate significant interference without high communication overhead. In this experiment, we deploy the three node topology above and use the channel emulator to vary the SINR on link $a-b$. Specifically, the link between node a transmitting the CSS and the receiving node b is maintained fixed to -82 dBm, while the attenuation on the interfering link $c-b$ is set in order to obtain the desired SINR. We iterate the experiment



(a) Over the air



(b) Channel emulation

Figure 3: Example of 127-symbol CSS correlation at -6 dB.

for different combinations of SINR and CSS lengths. Each experiment consists in the detection of at least 100 CSS's of lengths L ranging from 63 to 511 symbols. We vary the SINR between 0 dB and -10 dB.

Figure 4 shows the probability of false negatives as a function of SINR and CSS length. Specifically, the x-axis denotes the SINR on link $a-b$, while the y-axis denotes the probability of false negatives. The different curves correspond to different CSS lengths. The figure shows that longer CSS's are more robust due to the processing gain, e.g., at -8 dB, CSS's of length 63 can be detected only 4% of the time (96% of false negatives in the figure), while CSS's of lengths 127, 255, 511 can be detected approximately 30%, 99%, and 100% of the time respectively. However, increasing the CSS length involves an overhead penalty; in fact, while a 63-symbol CSS can be delivered in about 3.15 μs, 127, 255, 511-symbol CSS's require 6.35, 12.75, 25.55 μs respectively. With regard to the probability of false positives, we never obtained more than a single occurrence (out of hundreds of thousands of tests performed) for all the experiments related to a fixed SINR and length combination. Finally, it is relevant to notice that our results show only minor degradation with respect to theoretical performance in AWGN channel. For instance, considering the probability of false positives and false negatives at -8 dB, the length of sequences with similar performance in AWGN would be 47, 81, 198 for the cases of 63, 127, 255 actual lengths. *We conclude that 127-symbol CSS's provide a good compromise between overhead (6.35 μs), and resilience as they can be detected at -6 dB with 5.7% false negatives and no false positives.*

3.4 CSS Detection versus Control Message Decoding

Our second experiment aims to show that control CSS's are more robust to noise than 802.11 control messages, i.e., that CSS's can be reliably detected at considerably lower SINR than control messages. The metrics we use are false positives for the case of CSS detection, and packet error rate for control message decoding. We consider 127-symbol CSS's vs. 160-bit messages transmitted via BPSK modula-

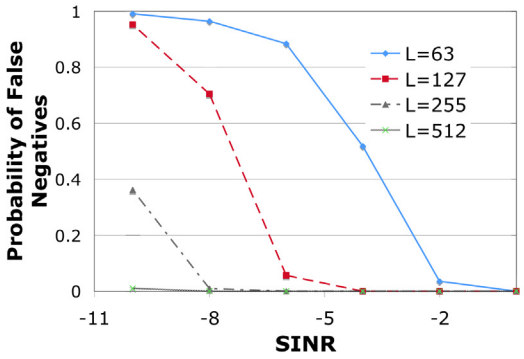


Figure 4: Robustness vs length tradeoff for different CSS lengths.

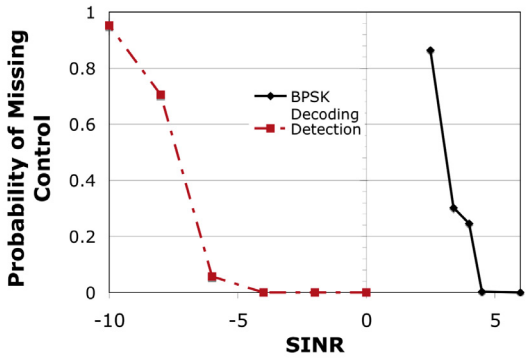


Figure 5: Probability of missing CSS detection vs missing message decoding.

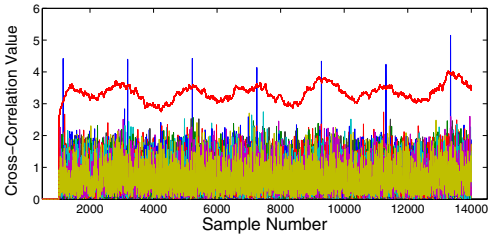


Figure 6: Low cross-correlation of CSS's different from the one transmitted.

tion, with 1/2 rate convolutional coding, corresponding to an RTS packet transmitted at base rate of 6 Mbps in 802.11 OFDM. In order to create the interference scenarios, we follow a methodology identical to the previous experiment. For the case of BPSK modulation, our experiment directly measures the BER out of at least 100000 bit transmissions. Then, we convert the obtained value to packet error rate, by considering a random and independent distribution of the bit errors among the packets (i.e., $1-(1-BER)^{PL}$, where PL is the packet length in bits). Note that the adoption of burst error models, such as Gilbert-Elliot [9], with expected burst length of 6 bits [27] may vary the results by about 1 to 1.5 dB.

Figure 5 shows two curves corresponding to CSS detection and control messages decoding. The x-axis denotes the SINR, while the y-axis denotes the *probability of missing control*, i.e., the probability of false negatives (resp. of packet decoding error) for CSS's (resp. for control messages). The plot shows that control CSS's are substantially more robust than control messages, since their probability of false negatives is much less than the error probability of control packets for any SINR. Furthermore, similar *probabilities of missing control* are obtained for the two control mechanisms, for SINR values separated by about 10 dB. For example, CSS's obtain probability of false detection of 5.7% at -6 dB, while control messages achieve 24% packet error rate at +4 dB, and 0.2% at +4.5 dB. *We conclude that due to the improved robustness of CSS detection with respect to packet decoding, control CSS's are about 10 dB more resilient to noise than 802.11 control messages.*

3.5 11ec Codebook Size

In 11ec, nodes use multiple CSS's and need to be able to reliably detect and discern all of them. In this experiment, we investigate whether cross-correlation between CSS's affects detection accuracy, and we explore the number of distinct CSS's that can be practically used. Specifically, we assess the probability of falsely detecting CSS A when CSS B is transmitted instead. For a given CSS length, a trade-off exists between the number of CSS's that 11ec uses, and the magnitude of the cross-correlation between any CSS pair, which in turn influences the probability of false positives. For this error probability to be small, we use well known sparse binary sequences, with optimal cross-correlation properties. Instances of such sequences have been studied for lengths corresponding to powers of 2, e.g., in cellular communications [21]. Different families of sequences provide larger (resp. smaller) sets of codewords, with larger (resp. smaller) cross-correlation between any codeword pair. Our design implements Gold codes, which provide 127 CSS's for our 127-symbol length, with a theoretical cross-correlation on the order of 12%. The choice of Gold codes permits us to support more than 50 co-located nodes by assigning distinct CSS's pairs to each node representing $I_c(r)$ and $A_c(s)$, while saving several CSS's for F_c^i/R_c^i pairs. In case a larger number of nodes needs to be supported, 11ec can switch to 255-symbol Kasami large codes for example, which allow more than 2000 nodes with 4011 CSS's.

To verify our choice, we emulated a situation in which a CSS A is sent, and 10 nodes try to detect CSS's different from A within the same samples. We repeated the experiment for 100 detection attempts, for 127-symbol Gold codes and SINR from 0 dB to -10 dB. The goal of this experiment is to assess the probability that the other nodes obtain false positives of their own CSS when the signature A is sent. The number of false negatives is immaterial in this experiment. For each SINR experiment, we obtained at most one false positive more than the case of a single CSS detection. Figure 6 shows an example outcome for -6 dB SINR (where the axes have the same meaning as in the experiment in Figure 3). The overlapping plots show the cross-correlation values obtained by 11 different nodes (including the one that expects to detect the transmitted CSS). While the number of spikes is unchanged, the noise looks visually denser due to overlapping plots. *We conclude that by using Gold codes, 11ec can support more than 50 co-located nodes without significant incidence of false positives.*

3.6 Discussion on Signal Correlation

Practical Detection Threshold Selection. The choice of the detection threshold is strategic in balancing the trade-off between false positives and false negatives. For example, as mentioned above, in Figures 3(a) and 3(b) the threshold is denoted by the thick crossing line; corresponding to the chosen threshold, the figures show 0 false positives and 0 false negatives. In general, a higher value of the threshold decreases the probabilty of false positives at the expense of a large probability of false negatives; viceversa, a lower threshold increases the occurrence of false positives. The theory of correlation in Gaussian noise provides an optimal threshold for a target detection SINR [14]. Specifically,

$$T = \sqrt{\frac{L \cdot \mathcal{E} \cdot \mathcal{N}}{2}} * Q^{-1}(P_{FA}) \quad (1)$$

where Q is the tail probability of the standard normal function. The formula shows that the optimal T depends on noise power $\mathcal{N}$, CSS power $\mathcal{E}$, CSS length L, and on the target probability of false positives P_{FA} that we fix to 10^{-8}. We remark that: *(i)* Generally, the power of the noise (which may be due to interfering transmissions) varies in time, thus the detection threshold should also change; *(ii)* Practically, it is difficult to estimate in advance the power of the noise and the power of the signal. In order to address the latter concern, we establish a lower bound on the SINR of the signal that we aim to detect (in our experiments -6 dB), and we tune T correspondingly (i.e., $T = \sqrt{\frac{L \cdot SINR \cdot \mathcal{N}^2}{2}} * Q^{-1}(P_{FA})$). Unfortunately, this solution is not sufficient because of the difficulty to estimate $\mathcal{N}$. In fact, the receiver can only measure the total power of the incoming signal, which may or may not contain the target CSS. Thus, we conservatively choose to replace $\mathcal{N}$ with the total signal power received, as if the incoming signal did not contain the CSS; practically, when the CSS is actually present, this choice has the effect of tuning T to a higher value than desired, i.e., it increases the occurrence of false negatives. Figures 3 and 6 show that the value of the threshold increases when the signal is present, and decreases otherwise.

Wireless Communications Issues. It is important to note that two issues may affect the performance of correlation, both due to the fact that the transmitter and receiver radio generate independent clocks [10, 22]. First, the clock phases at the transmitter and receiver are in general not aligned; this produces a *phase offset* between the two radios, which causes a fixed rotation of the received symbols of an angle γ. In order to compensate for this effect, we compute the magnitude of the correlation, with a theoretical penalty on the processing gain of about 0.5 dB [20]. Second, while the nominal frequencies of transmitter and receiver clocks are identical, they practically differ by a small Δf; this problem is known as *carrier frequency offset*. Carrier frequency offset produces a continuous rotation of the received symbols. Practically, Δf is sufficiently small (e.g., $\sim$ 1-4 KHz [22]), so that its effect is negligible over the CSS lengths/durations considered in this paper.

Hardware Implementation. The hardware implementation of CSS transmission and detection only requires the replication of components which are already present in off-the-shelf 802.11 chipsets, and specifically of filters and correlators. The basic implementation of 11ec needs four correlators (additional correlators may help increase channel reservations robustness as per Section 2.2). Because the correlated BPSK sequences at most require a sign flip on the received I/Q samples and several summations (with respect to expensive multiplications required to implement 802.11 floating-point correlators), CSS correlators occupy a very limited amount of resources.

4. EXPERIMENTAL EVALUATION OF 11EC

In this section, we present an experimental validation of 11ec using a measurement-based emulator that we design and implement. We perform the following experiments.

(i) We investigate the performance of 11ec in basic topologies that typically incur loss or imbalance for 802.11. **Our finding is that 11ec increases the throughput of underserved flows compared to 802.11 with or without RTS/CTS up to 10-fold (resp. 200-fold), with a benefit of almost 60% (resp. 55.6%) in fairness according to the Jain index.**

(ii) We investigate the performance of 11ec in a larger 5-flow network topology. **Our finding is that compared to 802.11 with or without RTS/CTS, 11ec achieves a gain of 30% (resp. 44%) in airtime utilization, and improves a severely underserved flow's throughput from 160 kbps (resp. 0 kbps) to 2.168 Mbps, for a gain of 1255%.**

The results in this section show that 11ec dramatically *improves network fairness*; furthermore, while such improvement can often decrease total utilization, 11ec remarkably *increases channel utilization*. Unlike 802.11, 11ec gives equal opportunities to weak links characterized by low data rates and strong high data rate links; for this reason, 11ec may sometimes achieve lower cumulative network throughput.

4.1 Measurement-driven Network Emulator

Our measurement-driven emulator is based on the GloMoSim simulator [28].[6] For the sake of realism, we modify GloMoSim in two ways: *(i)* we implement the support for multiple 802.11a/g modulations, i.e., BPSK 1/2, QPSK 1/2, 16QAM 1/2 and 64QAM 3/4 (corresponding to 6, 12, 24, 54 Mbps respectively); *(ii)* we implement a new propagation model that calculates links attenuation using our channel measurements. Specifically, with regard to the former, we perform a set of measurements at the channel emulator using the same transmitter/receiver/interferer setup described in the previous section, and we measure the BER as a function of the SINR. With regard to the second issue, we deploy up to 8 WARP nodes simultaneously in an office building (see Figure 7) and measure the signal strength between any pair, i.e., for any run of the experiment a single node transmits 400 μs packets and all others record the received power. As a result of these two measurements, we manually select for each link in the network emulator the highest data rate that its channel SINR can support with negligible packet error probability. Finally, we integrate all results into our measurement-driven emulator.

CSS Implementation. We implement CSS reception and detection as an autonomous physical layer component,

[6]Despite that the last version of GloMoSim dates to late 2001, the basic operations of the 802.11 MAC layer are consistent with the latest standard. The physical layer includes features such as noise accumulation, which make it preferable to alternative simulators.

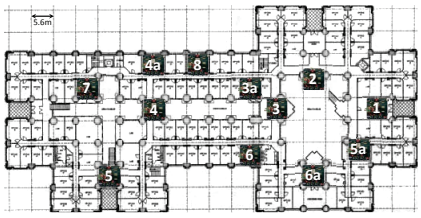


Figure 7: Layout of our office building deployment.

independent of the packet detection architecture of GloMoSim, i.e., CSS's are not simulated via small packets. Specifically, nodes store incoming CSS's, and schedule their evaluation after a delay corresponding to CSS's length, i.e., 6.35 μs for a 127-symbol CSS. For any stored CSS, the emulator keeps track of the variation of the background interference. At the moment of the evaluation the average SINR of the CSS is computed, and CSS detection is triggered if the SINR exceeds a threshold tuned to -6 dB for 127-symbol CSS (see Section 3.3). Our implementation permits each node to simultaneously store, evaluate and potentially detect multiple CSS's overlapping with other CSS's or incoming packets. Note that the original GloMoSim implementation of packet decoding does not support any of the features above, i.e., delayed evaluation and simultaneous multi-signal reception.

Finally, we implemented 11ec's MAC layer state machine by building on GloMoSim's 802.11. In particular, the design integrates the novel procedures corresponding, e.g., to deferral and timeout management.

4.2 Basic Topologies

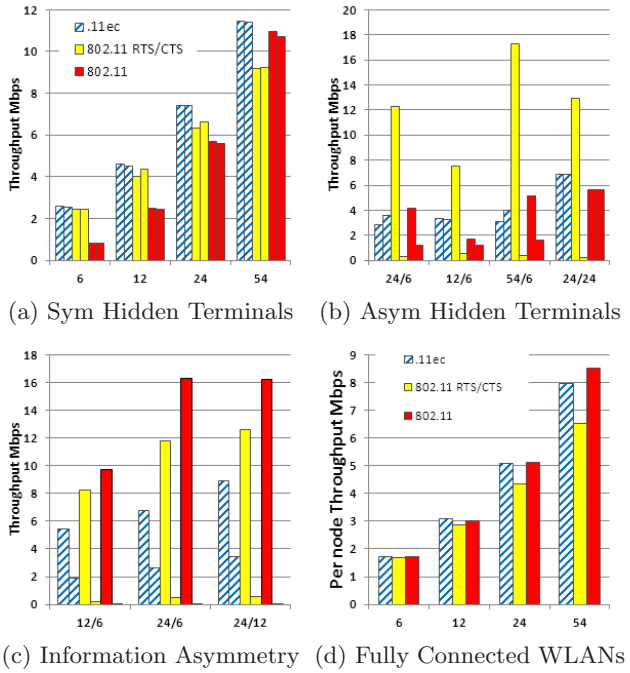
In this set of experiments, we evaluate the performance of 11ec in a few basic topologies (mostly including two flows) that are characterized by symmetries or asymmetries in link signal strength differences and carrier sensing relationships. This study is important because these topologies are at the origin of throughput losses and/or imbalances in 802.11-based networks [6]. We show that 11ec largely overcomes the problems of 802.11 with and without RTS/CTS.

In our study, we classify the basic topologies into 4 main groups according to the prevalence of one of the two links with respect to the other (e.g., due to higher SINR), and to the carrier sensing relationships between the transmitters. Specifically, the 4 topologies are the following: *(i) Symmetric Hidden Terminals* include the case where two links are formed by transmitters that do not carrier sense each other, and share a common receiver; moreover, the links have similar reception power at the receiver. Specifically, these topologies include links whose signal strengths at the receiver are not different by more than 4 dB; we choose this threshold to exclude capture at any 802.11 modulation. *(ii) Asymmetric Hidden Terminals* include topologies where two non carrier sensing transmitters share a common receiver, but one of the formed links has a significant power advantage. Specifically, these topologies includes links whose signal strengths at the receiver differ by more than 5 dB; the choice of the threshold is to permit to one of the two links to capture over the other at BPSK modulation. *(iii) Information Asymmetries* include link pairs *a-b* and *c-d* whose transmitters *a* and *c* do not carrier sense each other, and whose receivers differ; moreover, one of the two links *c-d* interferes with the other link *a-b*, but not viceversa. *(iv) Fully Connected WLANs* include topologies where all nodes carrier sense each other and transmit to a common receiver.

Most of the experiments in this section are performed by reproducing the topologies in our measurement-driven network emulator, with fully backlogged UDP/CBR traffic formed by 1024-byte packets. Each figure includes the bar graph of the throughputs of the flows for the three protocols we compare, namely 11ec, 802.11 with RTS/CTS, and 802.11 without RTS/CTS. Where utilized, RTS/CTS are transmitted at the OFDM 6 Mbps base rate. 11ec implementation includes CSS acknowledgements, but does not support RRTS mechanisms. The experiments in Figures 8(a) to 8(c) involve flow pairs; accordingly, the figures contain groups of six bars that correspond to the throughput of each flow, as achieved by the three protocols. The x-axes of the graphs indicate the data rates of the flows involved in the denoted experiment (when two different values are used, they orderly match the bar pairs), while the y-axes are in Mbps. In Figure 9 we introduce a metric termed *total airtime utilization*, which denotes the time share during which successful data packets are transmitted. Finally, we evaluate fairness according to two well known indicators, namely the Jain index [12], and proportional fairness [15]. The Jain index assumes values in the interval $[0, 1]$; for both indicators, higher values correspond to higher fairness.

Symmetric Hidden Terminals. We consider four instances of symmetric hidden terminals: three of them are selected from our deployed network, and use modulation rates corresponding to 6/12/24 Mbps, respectively. The last case is artificially generated in order to explore the effect of the use of higher modulation schemes. Figure 8(a) shows that all solutions assign similar throughputs to both flows. However, 11ec and 802.11 with RTS/CTS achieve considerably higher total throughput than 802.11 for most rates. Furthermore, 11ec outperforms 802.11 with RTS/CTS due to the smaller duration of CSS's with respect to control messages (11ec control CSS exchange lasts 19.7 μs, with respect to the 128 μs of 802.11 control messages). This entails a reduction of the probability of collisions (see Section 2.3), and lower control overhead. Both effects become more and more evident as the packet data-rate increases; at 54 Mbps the total throughput gain of 11ec over 802.11 with RTS/CTS is about 25%. Finally, at 54 Mbps data packets are sufficiently short to permit low collision probability to 802.11 without RTS/CTS; nonetheless, 11ec still shows 5% throughput gain.

Asymmetric Hidden Terminals. We consider four instances of asymmetric hidden terminals, all based on actual link power measurements. In these topologies, packets sent at base rate (e.g., control packets) by the sender with high SNR capture over packets sent by the sender with low SNR. However, because of our choice of the data modulation rate as the highest that can be supported by the link in the absence of interference, data packets always collide for both senders. Figure 8(b) shows that the capture effect has disastrous consequences for the flow with low SNR in 802.11 with and without RTS/CTS, while it has no effect on 11ec. For example, in the first instance represented (i.e., the left-most six bars in the figure) 11ec improves the throughput of the under-served flow by 11-fold (resp. by 2-fold) with respect to 802.11 with (resp. without) RTS/CTS. In 802.11 with RTS/CTS the imbalance is due to the fact that in case of overlapping RTS, the RTS of the stronger link is correctly decoded, while the other is ignored. Since 802.11



(a) Sym Hidden Terminals (b) Asym Hidden Terminals

(c) Information Asymmetry (d) Fully Connected WLANs

Figure 8: Throughput of 11ec, 802.11 with/without RTS/CTS in basic topologies

without RTS/CTS does not use the base rate, but transmits all packets at data rate, the throughput imbalance is originated only by the shorter duration of the data packet of the dominant link, which permits it to enjoy higher success probability. The result for the last instance (i.e., last two bars in the graph) supports this claim: when both links transmit at 24 Mbps, their throughputs do not depend on any SNR imbalance. Finally, in general, because of the larger number of collisions, 802.11 without RTS/CTS has a lower total throughput. In contrast to 802.11, 11ec reduces the imbalance by reducing the number of collisions, and actually reverses the imbalance, i.e., the link with the lower SNR obtains higher throughput. In brief, this is due to the fact that the link with higher SNR has the possibility to collide multiple times with a single packet sent at lower data rate by the link with lower SNR. This deduction is corroborated by the last instance (i.e., the right-most group of six bars) in Figure 8(b); for two links with different SNR, but both using 16QAM 1/2 modulation, the throughputs are identical.

In terms of fairness, considering for example the first instance, 11ec improves the Jain index from approximately 0.52 and 0.76 in 802.11 with and without RTS/CTS to 0.98; similarly, in terms of proportional fairness, 11ec improves the sum of the logs of the rates from 4.69 and 4.82 of 802.11 with and without RTS/CTS to 5.13. The total throughput of 11ec is lower than the competitors; however, this is misleading, and is due to the fact that 11ec improves the throughput of flows with lower data rate. Figure 9 clarifies this aspect, by showing the total airtime utilization for all instances represented in Figure 8(b). 11ec obtains up to 30% higher airtime utilization than the other 802.11 versions.

Information Asymmetry. The three instances we consider are again based on our channel measurements. In these

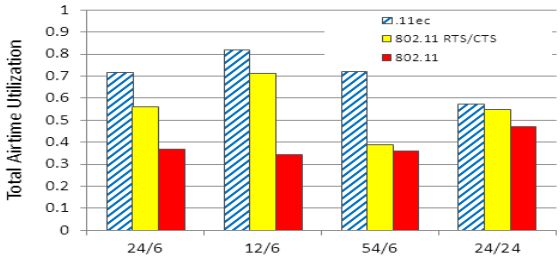


Figure 9: Total airtime utilization in the case of Asymmetric Hidden Terminals.

topologies, the link interfering always succeeds, while the interfered may become severely under-served due to high number of collisions. In fact, the sender of the under-served link cannot perceive when the channel is free at its receiver, and randomly selects transmission instants; in the likely case of collision, the sender of the under-served link decreases its sending rate due to backoff. Figure 8(c) shows that the information asymmetry completely starves the under-served link in 802.11. 802.11 with RTS/CTS slightly outperforms the version without, due to the larger probability of the under-served flow to correctly transmit an entire RTS without being interrupted by the interferer. It is important to notice that several of the bars corresponding to 802.11 flows in this and the next figures are almost or completely invisible. In contrast, even without the feature of control during data, 11ec manages to assign a significant throughput (about 40% of the interfering link) to the under-served link. For example, in the first instance represented (i.e., the left-most six bars in the figure) 11ec improves the throughput of the under-served link by 20-fold (resp. by 200-fold) with respect to 802.11 with (resp. without) RTS/CTS. In terms of fairness, 11ec improves the Jain index from approximately 0.5 in 802.11 with and without RTS/CTS to 0.8, with a 60% gain, and proportional fairness from 4.30 and 3.03 of 802.11 with and without RTS/CTS to 5.14. Similarly to the case of asymmetric hidden terminals, 11ec achieves a cumulative throughput lower than 802.11 but gains up to 26% and 5% in airtime utilization with respect to 802.11 with and without RTS/CTS. These results are due to the small size of control CSS's that have a high probability of being received during free channel intervals.

Fully Connected WLANs. Figure 8(d) shows the average throughput obtained for three carrier-sensing links, and groups of three bars correspond to the three protocols compared. In this scenario, 802.11 without RTS/CTS performs the best due to low overhead and rare collisions. In the worst case of 54 Mbps, 11ec obtains 6% less throughput than 802.11 without RTS/CTS, while 802.11 with RTS/CTS achieves 18% less throughput than 11ec due the control message overhead. At low data rates all protocols perform similarly due to the long packet durations. This result clearly shows that even in the absence of hidden terminals, control CSS's and larger slot size of 11ec do not involve significant throughput penalties.

Final Remarks. First, in the experiments above, we consider fixed (large) packet size and UDP traffic. We notice that: *(i)* as the packet size decreases, the overhead of RTS/CTS becomes more and more relevant, i.e., 11ec is expected to have larger and larger relative gain; *(ii)* through-

put imbalances highly affect TCP in most 802.11 cases. By inspecting the traces, we also notice that even in cases of balanced throughput (such as *symmetric hidden terminals*), 802.11 (with and without RTS/CTS) alternately serves for long periods of time one of the two links, by almost starving the other [4]; this is extremely detrimental for TCP performance. Second, our experiments do not implement rate-adaptation, but manually select the best rate achievable based on the links SNR. We observe that rate adaptation does not produce any benefit to 802.11 with RTS/CTS, since control messages need still be transmitted at base rate, and data packets rarely collide. On the other hand, 802.11 without RTS/CTS may benefit in case of hidden terminals, but typically at the price of higher unfairness even in fully connected WLANs due to capture effect [18]. Third, we notice that in the asymmetric topologies, the feature of receiver-side signaling, briefly mentioned at the end of Section 2 and which we do not implement in the experiments in this paper, improves the throughput of links with low SNR due to the capability of a receiver to contend for channel access. Finally, it is remarkable to notice that in contrast to common tenets of related literature, RTS/CTS at 6 Mbps does produce a large performance improvement vs. without RTS/CTS, and only slightly penalizes the throughput in the absence of hidden terminals.

4.3 Network Wide Experiment

Here we investigate larger topologies in order to demonstrate the fairness gains of 11ec in case of multiple flow interactions. We consider a 5-flow topology based on the channel measurements we performed; the flows operate at 24, 24, 24, 54, and 6 Mbps, respectively. Figure 10 shows the detailed bar graph of the throughput of all flows for 11ec and 802.11 with/without RTS/CTS; the flows referred on the x-axis match the node positions in Figure 7. Notice that for need of representation, we do not entirely reproduce the throughput of the flow $5a \rightarrow 1$ in the case of 802.11 without RTS/CTS; that flow's throughput is 21.23 Mbps because of high data rate and strong capture effect. The figure shows that 802.11 with RTS/CTS and 11ec achieve higher fairness than 802.11 without RTS/CTS. In addition, while 802.11 with RTS/CTS almost starves flow $4a \rightarrow 3a$ by assigning 160 Kbps, 11ec manages to assign it 2.168 Mbps for a gain of 1255%. Because the flow operates at 6 Mbps, this has a large effect on the airtime utilization, which increases from 0.5 and 0.45 of 802.11 with and without RTS/CTS to 0.65 of 11ec, for a gain of 30% and 44% respectively. 802.11 without RTS/CTS completely starves that flow. Finally, 11ec significantly increases throughput fairness; specifically, the Jain index is 0.65, 0.23, 0.73 for 802.11 with and without RTS/CTS, and 11ec respectively. 11ec shows about 10% higher proportional fairness with respect to 802.11 with RTS/CTS.

5. RELATED WORK

Many random access MAC protocols have been proposed to mitigate collisions and address hidden terminals, e.g., [4, 13, 25]. In contrast to tones and messages used by such protocols, we design a new primitive of correlatable symbol sequences which, by virtue of being short and robust, increases throughput and fairness with minimal overhead. More recently, while several papers have addressed 802.11 throughput overhead reduction [18, 23], they completely neglect the case of hidden terminals and, because of their more

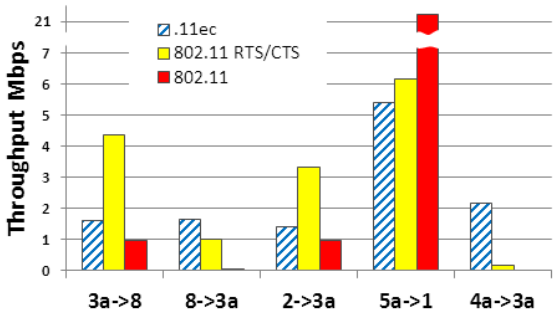


Figure 10: Throughput distribution for a 5-flow topology.

aggressive contention mechanisms, suffer severe throughput penalties in their presence. Ongoing standardization efforts, namely 802.11ah [2], target overhead reduction for sub-GHz communications, with application to smart grids, surveillance systems, etc. The strategy adopted in [2] consists in removing or compressing some information fields of the control messages, for a total of few bytes; compared to 11ec, this has a minor effect on the control message overhead, e.g., due to the preservation of the cumbersome preamble structure. Other techniques have been proposed to improve 802.11 throughput, e.g., [10, 17, 22], but address collision resolution rather than collision avoidance. For this reason, they are complementary to (and can be used in combination with) 11ec. Furthermore, our physical layer model is significantly more simple than [10, 18, 22, 23], since it relies only on the replication of components (correlators) that are already present in common 802.11 cards. Finally, other applications of signal correlation have been recently shown in [11, 24, 29].

6. CONCLUSIONS

In this paper we introduce 802.11ec, an 802.11-based protocol without control messages. 802.11ec introduces control correlatable symbol sequences which provide robustness and efficiency. Through a wide set of experiments on a software defined radio we show that 6.35 μs correlatable symbol sequences can be detected at an SINR of -6 dB, i.e., 10 dB lower than 802.11 control messages. Finally, we implement 802.11ec on a measurement-based network emulator and show that it improves network fairness by up to 217%, channel utilization by up to 44%, and throughput of underserved flows by 1255%, with respect to 802.11.

7. ACKNOWLEDGEMENTS

This research was supported by NSF grants CNS-1126478 and CNS-1012831 and by a grant from Cisco Systems Inc. O. Gurewitz is in part supported by EU FP7-FLAVIA project, contract number 257263.

8. REFERENCES

[1] IEEE Std 802.11-2007 - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE*, 2007. `http://standards.ieee.org/about/get/802/802.11.html`

[2] IEEE P802.11 Sub 1GHz Study Group. http://www.ieee802.org/11/Reports/tgah_update.htm
[3] Rice University WARP project. Available at: http://warp.rice.edu.
[4] V. Barghavan, A. Demers, S. Shenker, L. Zhang. MACAW: a Media Access Protocol for Wireless LAN's. In *Proc. of ACM SIGCOMM*, 1994.
[5] G. Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, 18(3): 535–547, March 2000.
[6] J. Camp, E. Aryafar, E. Knightly. Coupled 802.11 Flows in Urban Channels: Model and Experimental Evaluation. In *Proc. of IEEE INFOCOM*, 2010.
[7] T. Cui, L. Chen, T. Ho. Energy Efficient Opportunistic Network Coding for Wireless Networks. In *Proc. of IEEE INFOCOM*, 2008.
[8] D.S.J. De Couto, D. Aguayo, J.C. Bicket, R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proc. of ACM MobiCom*, 2003.
[9] E.O. Elliot. Estimates of Error Rates for Codes on Burst-Noise Channels. *Bell Syst. Tech. J.*, 42(5): 1977–1997, September 1963.
[10] S. Gollakota, D. Katabi. Zig-zag Decoding: Combating Hidden Terminal in Wireless Networks. In *Proc. of ACM SIGCOMM*, 2008.
[11] S. Hong, S. Katti. DOF: A Local Wireless Information Plane. In *Proc. of ACM SIGCOMM*, 2011.
[12] R.K. Jain, D.M.W. Chiu, W. Hawe. A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems. *Technical Report TR-301,* DEC Research, 1984.
[13] P. Karn. MACA- A New Channel Access Method for Packet Radio. In *Proc. of ARRL Computer Networking Conference*, 1990.
[14] S.M. Kay. *Fundamentals of Statistical Signal Processing - Vol. 2.* Prentice Hall, 1998.
[15] F.P. Kelly, A.K. Maulloo, D.K.H. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49(3): 237–252, March 1998.
[16] K. LaCurts, H. Balakrishnan. Measurement and Analysis of Real-World 802.11 Mesh Networks. In *Proc. of ACM IMC*, 2010.
[17] T. Li, M.K. Han, A. Bhartia, L. Qiu, E. Rozner, Y. Zhang. CRMA: Collision-Resistant Multiple Access. In *Proc. of ACM MobiCom*, 2011.
[18] E. Magistretti, K.K. Chintalapudi, B. Radunovic, R. Ramjee. WiFi-Nano: Reclaiming WiFi Efficiency via 800 ns Slots. In *Proc. of ACM MobiCom*, 2011.
[19] B. Radunovic, D. Gunawardena, P. Key, A. Proutiere, N. Singh, V. Balan, G. Dejean. Rethinking Indoor Wireless Mesh Design: Low Power, Low Frequency, Full-duplex. In *Proc. of IEEE WiMesh*, 2010.
[20] M. Richards. *Fundamentals of Radar Signal Processing.* McGraw-Hill, 2005.
[21] D.V. Sarwate, M.B. Pursley. Crosscorrelation properties of pseudorandom and related sequences. *Proceedings of the IEEE*, 68(5): 593–619, May 1980.
[22] S. Sen, R.R. Choudury, S. Nelakuditi. CSMA/CN: Carrier Sense Multiple Access with Collision Notification. *Proc. of ACM MobiCom*, 2010.
[23] S. Sen, R.R. Choudury, S. Nelakuditi. No Time to Countdown: Moving Backoff to the Frequency Domain. In *Proc. of ACM MobiCom*, 2011.
[24] K. Tan, H. Liu, J. Fang, W. Wang, J. Zhang, M. Chen, G. Voelker. SAM: Enabling Practical Spatial Multiple Access in Wireless LAN. In *Proc. of ACM MobiCom*, 2009.
[25] F. Tobagi, L. Kleinrock. Packet Switching in Radio Channels: Part II–The Hidden Terminal Problem in Carrier Sense Multiple-Access and the Busy-Tone Solution. *IEEE Transactions on Communications*, 23(12): 1417–1433, December 1975.
[26] R. Van Nee, R. Prasad. *OFDM for Wireless Multimedia Communications.* Artech House, 2000.
[27] A. Willig, M. Kubish, C. Hoene, A. Wolisz. Measurements of a Wireless Link in an Industrial Environment Using an IEEE 802.11-Compliant Physical Layer. *IEEE Trans. on Industrial Electronics*, 49(6): 1265–1282, December 2002.
[28] X. Zeng, R. Bagrodia, M. Gerla. GloMoSim: a library for parallel simulation of large-scale wireless networks. In *Proc. of IEEE PADS*, 1998.
[29] X. Zhang, K. Shin. E-MiLi: Energy-Minimizing Idle Listening in Wireless Networks. In *Proc. of ACM MobiCom*, 2011.

APPENDIX

A. SIGNAL CORRELATION

A CSS s is detected via cross-correlation with a local copy, i.e., at the reception of a complex signal y that may contain s, y is cross-correlated with the complex conjugate of the target CSS $s*$. Formally, for a CSS of length L,

$$\mathcal{C}(\Delta) = \sum_{0}^{L-1} s^*(k)y(k+\Delta) \quad (2)$$

where Δ represents the position of the correlation with respect to the input signal, i.e., the sample for which we perform the correlation. Note that: *(i)* if $[y(\Delta)\dots y(\Delta+L-1)]$ does not contain exactly s, the value of $\mathcal{C}(\Delta)$ is nearly 0; *(ii)* if $[y(\Delta)\dots y(\Delta+L-1)]$ contains a copy of s with sufficient SINR, the $\mathcal{C}(\Delta)$ obtains a large value proportional to the energy of the signal.

We use cross-correlation as a test statistic for the detection of a target CSS, and repeat its computation at each new sample of the incoming signal. Specifically, detection is performed by setting a threshold T (see Section 3.6); if $\mathcal{C}(\Delta) \geq T$ (resp. $\mathcal{C}(\Delta) < T$), the presence (resp. absence) of the CSS in $y(k+\Delta)$ is declared. The performance of cross-correlation can be quantified in terms of false positives and false negatives. Specifically, after deciding on a threshold T, a false positive is declared when $C(\Delta) < T$ even though the CSS is present within $[y(\Delta)\dots y(\Delta+L-1)]$, and a false negative is declared when $C(\Delta) \geq T$ even though the CSS is absent. Cross-correlation detection provides a processing gain, which is linear in the length of the correlated sequence [14]. This means that a sequence of length $2L$ obtains a value of $C(\Delta)$ twice as large a sequence of length L, i.e., it is considerably more likely to exceed T.