Computer Science and Artificial Intelligence Laboratory
Technical Report

MIT-CSAIL-TR-2010-036 August 2, 2010

MAC Design for Analog Network Coding

Majid Khabbazian, Fabian Kuhn, Nancy Lynch,
Muriel Medard, and Ali ParandehGheibi

massachusetts institute of technology, cambridge, ma 02139 usa — www.csail.mit.edu

MAC Design for Analog Network Coding

Majid Khabbazian[1], Fabian Kuhn[2], Nancy Lynch[1], Muriel Médard[3], and Ali ParandehGheibi[3]
mkhabbazian@gmail.com, fabian.kuhn@usi.ch, lynch@csail.mit.edu, medard@mit.edu, and parandeh@mit.edu
[1] Computer Science and Artificial Intelligence Lab, EECS, MIT, Cambridge MA, USA
[2] Faculty of Informatics, University of Lugano, 6904 Lugano, Switzerland
[3] Research Laboratory of Electronics, EECS, MIT, Cambridge MA, USA

July 30, 2010

Abstract

Most medium access control mechanisms discard collided packets and consider interference harmful. Recent work on Analog Network Coding (ANC) suggests a different approach, in which multiple interfering transmissions are strategically scheduled. The received collisions are collected and then used in a decoding process, such as the ZigZag decoding process, where the packets involved in the collisions are extracted. In this paper, we present an algebraic representation of collisions and describe a general approach to recovering collisions using ANC.

To study the effect of using ANC on the performance of MAC layers, we develop an ANC-based algorithm that implements an abstract MAC layer service, as defined in [1, 2], and analyze its performance. This study proves that ANC can significantly improve the performance of MAC layer services, in terms of probabilistic time guarantees for packet delivery. We illustrate how this improvement at the MAC layer can translate into faster higher-level algorithms, by analyzing the time complexity of a multiple-message network-wide broadcast algorithm that uses our ANC-based MAC service.

1 Introduction

The nature of wireless networks is intrinsically different from that of wired networks because the wireless medium is shared among many transmitters. The conventional approach to the Medium Access Control (MAC) problem is contention-based protocols in which multiple transmitters simultaneously attempt to access the wireless medium and operate under rules that provide enough opportunities for all nodes to transmit. Examples of such protocols in packet radio networks are ALOHA [3], MACAW [4], and CSMA/CA [5].

In contention-based protocols it is possible that two or more nodes transmit their packets simultaneously, which can result in a *collision* at the receiver. The colliding packets are generally considered to be lost. Therefore, these protocols strive to avoid simultaneous transmissions by nearby nodes. Recently, Gollakota and Katabi [6] showed how one might recover collided packets in an 802.11 system using *ZigZag decoding*, if there are relatively few colliding packets and enough transmissions involving these packets. Their scheme requires the network to operate at a sufficiently high signal-to-noise ratio (SNR) that noise can be neglected and per-symbol detection can be assumed to be error-free in the absence of a collision on that symbol. In fact, they suggest that each collision can be treated as a linear equation over the packets involved. Therefore, packets are recoverable if the resulting system of linear equations has a unique solution. This gives rise to the possibility of designing MAC protocols that exploit Analog Network Coding (ANC) [7] to increase network capacity. In such MAC protocols, unlike the conventional protocols, interference is not considered harmful. In fact, such protocols strategically schedule simultaneous transmissions in order to increase network capacity. Note that, as in digital network coding, packets are mixed together in ANC. However, in digital network

coding, the sender mixes the contents of packets before transmission whereas in ANC the wireless channel naturally mixes the packets.

In this paper, we present a new MAC protocol, *CMAC*, which exploits Analog Network Coding, and prove that it provides strong correctness and performance guarantees. Formally, we prove that *CMAC* implements the probabilistic MAC layer specification introduced in [1, 2]. This layer provides probabilistic upper bounds on the time for a packet to be delivered to any neighboring node (the *receive delay bound*), and on the total amount of time for a sender to receive an acknowledgment of successful delivery to all neighbors (the *acknowledgment delay bound*). It also provides a bound on the amount of time for a receiver to receive *some* packet from among those currently being transmitted by neighboring senders (the *progress bound*).

Showing that our *CMAC* protocol implements the probabilistic MAC layer allows us to compare the bounds achieved using ANC to those achieved using more conventional protocols. In Section 5, we show that *CMAC* implements the probabilistic MAC layer with receive delay bound, acknowledgment delay bound, and progress bound all of the form $\mathcal{O}\left(\Delta + \frac{\Delta}{c}\log(\frac{\Delta}{\epsilon})\right)$. where Δ is the maximum node degree and c is the maximum number of packets for which a collision can be decoded. In particular, if c is $\Omega(\Delta)$, then using ANC, a node can deliver a packet to all its neighboring nodes (and receive a packet from any given neighboring node) in time $\mathcal{O}(\Delta + \frac{1}{\epsilon})$. In contrast, the exponential decay protocol *DMAC* presented in [1, 2] has larger receive and acknowledgment delay bounds, of the form $\mathcal{O}(\Delta \log(\frac{1}{\epsilon}) \log(\Delta))$, but a smaller progress bound, $\mathcal{O}(\log(\Delta))$. In Section 6, we present another MAC protocol, *DCMAC*, which improves the progress bound of *CMAC* to $\mathcal{O}(\log(\Delta))$ without increasing the receive and acknowledgment delay bounds. *DCMAC* achieves these bounds by interleaving *CMAC* and *DMAC*.

An advantage of implementing the probabilistic MAC layer using ANC is that it allows us to easily analyze the effect of using ANC on the time complexity of higher-level algorithms such as network-wide broadcast algorithms. For example, in Section 7, we combine our analysis of *CMAC* with the analysis of a high-level multiple-message global broadcast protocol over the probabilistic MAC layer (from [1, 2]). This yields time bounds for multi-message broadcast over the basic network. Our results show that time complexity of multi-message broadcast can be significantly improved using ANC.

The remainder of the paper is organized as follows. In Section 2, we discuss related work. Section 3 presents our network assumptions, including the key facts about Analog Network Coding. Section 4 describes the probabilistic abstract MAC layer, as defined in [2]. Sections 5 and 6 present our two algorithms to implement the probabilistic MAC layer. Section 7 describes a simple network-wide broadcast algorithm and analyzes its time complexity. Section 8 concludes.

Notation: Let $\mathbb{C}$ denote the set of complex numbers. Throughout the paper, log denotes the base two logarithm and ln the natural logarithm. For any positive integer n, $[n]$ denotes the set $\{1, \ldots, n\}$.

2 Related Work

Analog Network Coding was first presented in [7]. For high SNR regimes, the asymptotic optimality of ANC was recently shown in [8, 9]. Its asymptotic optimality in terms of rate-diversity trade-off was established in [10, 11]. The use of of ANC, as a generalization of ZigZag, in possible combination with digital network coding, in order to increase the throughput of packetized multiple-access systems, has been considered in [12]. In that work, an algebraic model for ANC was derived, which takes into account explicitly symbols, digital network coding, modulation, and channel effects, such as attenuation, delay and additive combination of signals. We use that model in this paper to model the algebraic interaction among nodes. The use of ZigZag without additional digital network coding has recently been considered by [13] to improve congestion control and maximize aggregate utility of the users. That approach does not construct an explicit MAC mechanism such as the one we provide in this paper.

The first abstract MAC layer specification was defined by Kuhn, Lynch, and Newport [14, 15]. This basic layer provides worst-case guarantees for packet receipt and acknowledgment. These papers also present and analyze greedy global broadcast algorithms over the basic MAC layer. Khabbazian *et al.* [1, 2] continued

this work by developing a probabilistic version of the MAC layer specification (the one used throughout this paper), presenting exponential decay algorithms to implement both layers, analyzing global broadcast algorithms over both layers, and showing how to combine the high-level and low-level results automatically to obtain performance results for global broadcast over an underlying collision-prone network model. Other work using abstract MAC layers includes algorithms for Neighbor Discovery over these layers [16, 17].

3 The Network Model

Fix a static undirected graph, $G = (V, E)$. Let $n = |V|$ be the number of vertices, Δ the maximum degree, and D the diameter (i.e., the maximum hop distance between two vertices in G). We assume that n active nodes reside at the n vertices of G. Nodes have unique identifiers. We assume that the nodes have local knowledge of the graph; in particular, they know Δ and know the identifiers of their neighbors in G.

We assume a slotted system, with slots of duration $t_{slot} = 1$. When a node transmits in some slot, its message *reaches* all G-neighboring nodes, and no other nodes. Thus, each node j, in each slot, is reached by some collection of packets from its transmitting neighbors. What j actually *receives* is defined as follows: (a) If j transmits, then it receives silence, $\perp$. Thus, a node cannot receive a packet while it is transmitting. (b) If j does not transmit and is reached by no packets, then it receives silence. (c) If j does not transmit and is reached by exactly one packet from another node, then it receives that packet. (d) If j does not transmit and is reached by two or more packets, then it receives a collision. We assume that each node stores all the received packets and collisions, and uses analog network coding (such as ZigZag decoding) to decode the collided packets.

A packet is essentially a vector of N symbols over a finite field $\mathbb{F}_q$, where q is a power of two. We represent a packet as a polynomial over the delay variable D, with coefficients being the symbols of $\mathbb{F}_q$ that form the packet. The mapping from the packet to the corresponding physical signal is a result of modulation. For a system such as ZigZag, which performs per-symbol detection, no channel coding precedes the modulation. For more general ANC, however, there may also be a channel code, requiring the use of interference cancellation over the entire packet, rather than symbol-wise operations as in ZigZag. For the sake of simplicity, we discuss here the case where no channel code is added, although our discussion can be extended to the case where we have channel coding (because the effect of the noise is not entirely negligible on a symbol-by-symbol bases). We abstract the modulation to be a one-to-one map M from symbols over $\mathbb{F}_q$ to the complex number field

$$M : \mathbb{F}_q \to \mathbb{C}.$$

Using the model of [12], an equivalent representation of the collisions of w packets at receiver j in l time slots is given by

$$\begin{pmatrix} C_j^{s_1}(k) \\ C_j^{s_2}(k) \\ \vdots \\ C_j^{s_l}(k) \end{pmatrix} = \begin{pmatrix} A_{i_1}^{s_1} & \dots & A_{i_w}^{s_1} \\ A_{i_1}^{s_2} & \dots & A_{i_w}^{s_2} \\ \vdots & & \vdots \\ A_{i_1}^{s_l} & \dots & A_{i_w}^{s_l} \end{pmatrix} \begin{pmatrix} S_{i_1}(k) \\ S_{i_2}(k) \\ \vdots \\ S_{i_w}(k) \end{pmatrix}, \quad k = 1, \dots, N,$$

where $C_j^s \in \mathbb{C}^N$ represent the collision in time slot s, $S_i \in \mathbb{C}^N$ is the signal (packet) transmitted by sender i, and $A_i^s \in \mathbb{C}$ are random variables corresponding to the combination of the modulation and channel propagation effects, as well as the transmission decision of sender i at time slot s.

Note that ZigZag relies on there being non-zero time shifts among colliding packets, whereas general ANC does not. The process of decoding by inverting this matrix is more general than the ZigZag procedure of [6]. For example, consider the case where two different nodes collide twice in two different time slots. If the offsets between two packets in the two time slots are exactly the same, the ZigZag decoding process fails. However,

the transfer matrix A may still be full-rank because of the change in the channel gains over time, and hence, we may decode the packets by Proposition 1. The decoding process results in the signals corresponding to the original packets. The signals then have to be demodulated to obtain the original data. This algebraic representation formalizes the intuition introduced in [6] that every collision is like a linear equation in the original packets. Let

$$A = \begin{pmatrix} A_{i_1}^{s_1} & A_{i_2}^{s_1} & \dots & A_{i_w}^{s_1} \\ A_{i_1}^{s_2} & A_{i_2}^{s_2} & \dots & A_{i_w}^{s_2} \\ \vdots & \vdots & & \vdots \\ A_{i_1}^{s_l} & A_{i_2}^{s_l} & \dots & A_{i_w}^{s_l} \end{pmatrix}.$$

Proposition 1. *Let P, $|P| = w$, be a set of packets. Consider a node j. Let S, $|S| = l$, be a set of slots such that in every slot in S, node j receives a packet in P or a collision involving packets in P. The received packets/collisions can be represented by a system of linear equations of the form $C(k) = A \times S(k)$, where $k = 1, 2, \dots, N$ and A is an $l \times w$ transfer matrix. Given this representation, if the transfer matrix A has rank w (i.e., full rank) over the field $\mathbb{C}$, then it is possible to decode all packets in P.*

So far, we have assumed that a collision of any number of packets can be treated as a linear equation involving those packets. The largest number of packets that can be allowed to collide for collision recovery to still work depends on the range of the received SNR. In practice, as the number of packets involved in a collision increases, it becomes less likely that the collision can be used for decoding. Hence, towards a more realistic setup, we assume that any collision involving more than c (a fixed parameter) packets is not useful and will be discarded. Throughout the paper, we assume that $4 \leq c \leq \Delta$. Note that at most Δ packets are involved in a collision based on our network assumptions. Thus, a decoder with parameter $c = \Delta$ is as powerful as one with parameter $c > \Delta$.

4 The Probabilistic Abstract MAC Layer

In order to make precise claims about the correctness and performance of a MAC-layer algorithm, one requires a formal specification of what a MAC layer should do. For our definition of MAC layer behavior, we use the *probabilistic abstract MAC layer specification* defined in [1, 2]. According to this specification, a MAC layer provides an external interface by which it accepts packets from its environment (a higher-level protocol) via $bcast(m)$ input events and delivers packets to neighboring nodes via $rcv(m)$ output events. It also provides acknowledgments to senders when their packets have been successfully delivered to all neighbors, via $ack(m)$ output events. Finally, it accepts requests from the environment to abort current broadcasts, via $abort(m)$ input events.[1]

The specification is implicitly parameterized by three positive reals, f_{rcv}, f_{ack}, and f_{prog}. These bound delays for a specific packet to arrive at a particular receiver, for an acknowledgment to be returned to a sender, and for *some* packet from among many competing packets to arrive at a receiver. The specification also has corresponding parameters ϵ_{prog}, ϵ_{rcv}, and ϵ_{ack}, which represent probabilities that the delay bounds are not attained. Finally, it has a parameter t_{abort}, which bounds the amount of time after a sender aborts a sending attempt when the packet could still arrive at some receiver.

We model a MAC layer formally as a *Probabilistic Timed I/O Automaton* (PTIOA), as defined by Mitra [18]. A MAC layer PTIOA Mac can be composed with an environment PTIOA Env and a network PTIOA Net. This composition, written as $Mac \| Env \| Net$, is itself a PTIOA, and yields a unique probability distribution on executions. To satisfy our specification, a MAC layer Mac must guarantee several conditions, when composed with any Env and with Net. To define these requirements, we assume some constraints on Env: an execution

[1] In the distributed algorithms research community, where this work originated, "packets" are generally called "messages", which explains our use of the letter m.

α of $Mac\|Env\|Net$ is *well-formed* if 1. it contains at most one *bcast* event for each m (i.e., all packets are unique), 2. any $abort(m)_i$[2] event is preceded by a $bcast(m)_i$ but not by an $ack(m)_i$ or another $abort(m)_i$, and 3. any two $bcast_i$ events have an intervening ack_i or $abort_i$ (i.e., each node handles packets one at a time).

The specification says that the Mac automaton must guarantee the following conditions, for any well-formed execution α of $Mac\|Env\|Net$. There exists a *cause* function that maps every $rcv(m)_j$ event in α to a preceding $bcast(m)_i$ event, where $i \neq j$, and that maps each $ack(m)_i$ and $abort(m)_i$ to a preceding $bcast(m)_i$. The *cause* function must satisfy:

1. *Receive restrictions:* If $bcast(m)_i$ event π causes $rcv(m)_j$ event π', then (a) *Proximity:* $(i, j) \in E$. (b) *No duplicate receives:* No other $rcv(m)_j$ caused by π precedes π'. (c) *No receives after acks:* No $ack(m)_i$ caused by π precedes π'. (d) *Limited receives after aborts:* π' occurs no more than t_{abort} time after an *abort* caused by π.

2. *Acknowledgment restrictions:* If $bcast(m)_i$ event π causes $ack(m)_i$ event π', then (a) *No duplicate acks:* No other $ack(m)_i$ caused by π precedes π'. (b) *No acks after aborts:* No $abort(m)_i$ caused by π precedes π.

In addition, the Mac automaton must guarantee three probabilistic upper bounds on packet delays. If π is a *bcast* event in a closed execution β[3], then we say that π is *active at the end of* β provided that π is not terminated with an *ack* or *abort* in β. The probabilistic MAC layer guarantees the following probabilistic bounds. Here, the notation Pr_β refers to the conditional distribution on executions that extend β.

1. *Receive delay bound:* Let β be a closed execution that ends with a $bcast(m)_i$ at time t. Let j be a neighbor of i. Define the following sets of time-unbounded executions that extend β: A, the executions in which no $abort(m)_i$ occurs, and B, the executions in which $rcv(m)_j$ occurs by time $t + f_{rcv}$. If $Pr_\beta(A) > 0$, then $Pr_\beta(B|A) \geq 1 - \epsilon_{rcv}$.

2. *Acknowledgment delay bound:* Let β be a closed execution that ends with a $bcast(m)_i$ at time t. Define the following sets of time-unbounded executions that extend β: A, the executions in which no $abort(m)_i$ occurs, and B, the executions in which $ack(m)_j$ occurs by time $t + f_{ack}$ and is preceded by $rcv(m)_j$ for every neighbor j of i. If $Pr_\beta(A) > 0$, then $Pr_\beta(B|A) \geq 1 - \epsilon_{ack}$.

3. *Progress bound:* Let β be a closed execution that ends at time t. Let I be the set of neighbors of j that have active *bcast*s at the end of β, where $bcast(m_i)_i$ is the bcast at i, and suppose that I is nonempty. Suppose that no $rcv(m_i)_j$ occurs in β, for any $i \in I$.
   Define the following sets of time-unbounded executions that extend β: A, the executions in which no $abort(m_i)_i$ occurs for any $i \in I$, and B, the executions in which, by time $t + f_{prog}$, at least one of the following occurs: an $ack(m_i)_i$ for every $i \in I$, a $rcv(m_i)_j$ for some $i \in I$, or a rcv_j for some packet whose *bcast* occurs after β.
   If $Pr_\beta(A) > 0$, then $Pr_\beta(B|A) \geq 1 - \epsilon_{prog}$.

The receive bound says that, with high probability, a packet is received by a particular neighbor within time f_{rcv}. The acknowledgment bound says that, with high probability, a packet is acknowledged within time f_{ack}, and moreover, the acknowledgment is "correct" in that the packet has actually been delivered to all neighbors. The progress bound says that, if a nonempty set of j's neighbors have active *bcasts* at some point, and none of these packets has yet been received by j, then with high probability, within time f_{prog}, either j receives one of these packets or something newer, or else all of these end with *acks*. This is all conditioned on absence of aborts.

[2]Here and elsewhere, subscripts are used to identify the node at which the event occurs.

[3]An execution of a PTIOA is closed if it is a finite sequence of discrete steps and trajectories, ending with a trajectory whose domain is a right-closed time interval. Formal details of such definitions appear in [19, 18].

5 Implementing the MAC Layer Using ANC

In this section, we present a new ANC-based algorithm to implement the probabilistic MAC layer. This algorithm yields smaller receive and acknowledgment delay bounds than the *DMAC* algorithm in [1, 2]. The progress bound, on the other hand, is larger. In Section 6, we combine our new algorithm with *DMAC* to obtain a small progress bound as well.

5.1 Probability that a Matrix is Full-Rank

The heart of the analysis of the algorithm is the following lemma, which expresses a lower bound on the probability that a matrix B generated randomly from an arbitrary matrix A has full rank. In Section 5.2, we use such random matrices to model the transmission behavior of the neighbors of a particular node k, where entry $B_{i,j}$ corresponds to the transmission behavior of node j in slot i. The conclusion of the lemma is used there to show that, assuming that the algorithm executes for enough slots, with high probability, node k receives enough information to recover a set of packets.

Construction of a random matrix: Let l and w be positive integers. Let A be an arbitrary matrix of size $l \times w$, with elements in $\mathbb{C} - \{0\}$. Let p be a real number, $0 \leq p \leq 1$. We construct a random $l \times w$ matrix B from A according to the following two-step procedure: Start with $B = A$. First, for each (i, j) independently, keep $B_{i,j}$ unchanged with probability p and set it to 0 with probability $1-p$. Second, for each i independently, keep row i of B unchanged with probability p and set it to be identically 0 with probability $1 - p$.

Lemma 1. *Let l and w be positive integers. Let A be an arbitrary matrix of size $l \times w$, with elements in $\mathbb{C} - \{0\}$. Let p and ϵ be real numbers, with $0 < p \leq \frac{1}{2}$ and $0 < \epsilon < 1$. Let c be a positive integer with $c \geq 4$ and $wp \leq \frac{c}{2}$. Suppose*

$$l \geq \left\lceil \frac{\frac{14e}{e-1}}{1-p} \left(w + \frac{\ln(w+1) + 2\ln(1/\epsilon)}{p} \right) \right\rceil .$$

Let B be a random matrix constructed from A as described above. Then, with probability at least $1 - \epsilon$, B has at least w independent rows, each containing at most c non-zero elements.

Proof Sketch. Instead of fixing the number of rows of B, assume that we construct B by adding rows until there are w rows with at most c non-zero entries that span $\mathbb{R}^w$. The lemma then follows by showing that with probability at least $1 - \epsilon$, the resulting matrix has at most l rows.

For each $d \in [w]$, we define random variables X_d and Y_d. Let X_d be the smallest integer such that the submatrix spanned by the rows with at most c non-zero entries among the first X_d rows of B has rank d. Further, we define $X_0 = 0$ and $Y_d = X_d - X_{d-1}$. Note that to prove the lemma, we need to show that $\Pr(X_w > l) < \epsilon$.

In the following, we refer to non-zeroed rows of B as the rows that are not set to 0 at the end of the construction of B. The non-zeroed rows of B are random vectors in $\mathbb{C}^w$, where each coordinate is non-zero independently with probability p. Assume that we are given $d - 1$ linearly independent vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{d-1}$ for some integer $d \geq 1$. The core of the proof is to lower bound the probability that a new non-zeroed row of B has at most c non-zero entries and is linearly independent to the given $d-1$ vectors. For simplicity, assume that the vectors $\boldsymbol{x}_i$ are vectors from the standard basis of $\mathbb{C}^w$. Hence, each of them is non-zero in exactly one coordinate and an additional vector is linearly independent iff it is non-zero in at least one coordinate in which none of the $d - 1$ vectors $\boldsymbol{x}_i$ is non-zero. The probability that all these $w - d + 1$ coordinates are 0 in a random non-zeroed row is $(1 - p)^{w-d+1}$ since all coordinates are set to something non-zero with probability p. The parameter c is chosen such that the probability that a non-zeroed row has at most c non-zero entries is lower bounded by some constant q. It can be shown that the probability that a non-zeroed row has at most c non-zero entries and is linearly independent of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{d-1}$ is at least

$$p_d = \frac{1}{7} \cdot \left(1 - (1-p)^{w-d+1}\right).$$

Further it can be shown that the same bound holds if the vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{d-1}$ are arbitrary vectors in $\mathbb{C}^w$. Therefore for each $d \geq 0$ and row $i > X_{d-1}$, if row i is not set to 0 in the final step of the construction of B, the probability that row i contains at most c non-zero entries and is independent of the first X_{d-1} rows is at least p_d. Thus, the random variables Y_d are dominated by independent geometric random variables Z_d with parameter $(1-p)p_d$. Using a Chernoff bound, it can be shown that

$$\Pr(X_d \leq l) \leq \Pr\left(\sum_{i=1}^{d} Z_d \leq l\right) \leq 1 - \epsilon.$$

This completes the proof sketch. A detailed proof appears in Appendix D. □

5.2 The Coding Algorithm

In this section, we describe a simple contention-resolution protocol, which is used in our MAC algorithm. Let c be the threshold parameter defined for ANC (in Section 3). Let $\rho = \frac{c}{2\Delta}$. Note that $\rho \leq \frac{1}{2}$, as $c \leq \Delta$.

Definition 1 ($\mathcal{R}_\epsilon$, where ϵ is a real, $0 < \epsilon < 1$)**.**

$$\mathcal{R}_\epsilon = \left\lceil \frac{\frac{14e}{e-1}}{1-\rho}\left(\Delta + \frac{\ln(\Delta+1) + 2\ln(1/\epsilon)}{\rho}\right)\right\rceil$$

Lemma 2. $\mathcal{R}_\epsilon = \mathcal{O}\left(\Delta + \frac{\Delta}{c}\log(\frac{\Delta}{\epsilon})\right)$.

Proof. $1 - \rho \geq \frac{1}{2}$. Thus,

$$\begin{aligned}
\mathcal{R}_\epsilon &= \left\lceil \frac{\frac{14e}{e-1}}{1-\rho}\left(\Delta + \frac{\ln(\Delta+1) + 2\ln(1/\epsilon)}{\rho}\right)\right\rceil \\
&= \left\lceil \frac{\frac{14e}{e-1}}{1-\rho}\left(\Delta + \frac{\ln\left((\Delta+1)(1/\epsilon)^2\right)}{\frac{c}{2\Delta}}\right)\right\rceil \\
&= \mathcal{O}\left(\Delta + \frac{\Delta}{c}\log(\frac{\Delta}{\epsilon})\right).
\end{aligned}$$

□

Our contention resolution algorithm, which we call simply *Coding*, has a single explicit parameter, ϵ. It also uses c as an implicit parameter.

Coding(ϵ), where ϵ is a real, $0 < \epsilon < 1$: Assume I is a set of nodes, $1 \leq |I| \leq \Delta$, and j is a distinguished node not in I. All the nodes in I participate in the algorithm, and j may or may not participate. Each participating node i has a packet m_i, assumed fixed for the entire algorithm.

The algorithm runs for exactly $\mathcal{R}_\epsilon$ slots. Every participating node participates in all slots, with no node starting or stopping participation part-way through the algorithm. At every slot, each participating node i transmits packet m_i with probability exactly $\rho = \frac{c}{2\Delta}$.

Lemma 3. *In Coding(ϵ), with probability at least $1-\epsilon$, node j receives all packets m_i, $i \in I$, by the end of the algorithm.*

Proof. The probability that node j receives all packets if it participates in the algorithm is at most equal to the probability that j receives all packets if it does not participate. So we assume without loss of generality that j participates.

We construct a random matrix B of size $\mathcal{R}_\epsilon \times |I|$, in $\mathcal{R}_\epsilon$ steps. In step l, $1 \le l \le \mathcal{R}_\epsilon$, we define row l, as follows. If j transmits in slot l, then row l is identically 0. If j does not transmit in slot l, then write $I = \{i_w | 1 \le w \le |I|\}$. For every w, $1 \le w \le |I|$, let $B_{l,w} = 0$ if node i_w does not transmit in slot l; otherwise let $B_{l,w}$ be a non-zero complex number (which corresponds to the channel gain from node i_w to node j and the offset of i_w's packet).

Now we apply Lemma 1, with l in the statement of that lemma equal to $\mathcal{R}_\epsilon$, w in the lemma equal to $|I|$, and $p = \rho$. Since

$$|I|\rho = |I|\frac{c}{2\Delta} \le \frac{c}{2},$$

and

$$l = \mathcal{R}_\epsilon = \left\lceil \frac{\frac{14e}{e-1}}{1-\rho}\left(\Delta + \frac{\ln(\Delta+1) + 2\ln(1/\epsilon)}{\rho}\right)\right\rceil,$$

the conditions required by Lemma 1 hold. Therefore, by Lemma 1, with probability at least $1-\epsilon$, B has a set S of $|I|$ independent rows, each containing at most c non-zero elements. Any row in S corresponds to a collision that j receives during the execution of $Coding(\epsilon)$. Consequently, by Proposition 1, j can decode all the packets m_i, $i \in I$, by the end of the algorithm, with probability at least $1-\epsilon$. □

5.3 The *CMAC* Algorithm

Now we present our MAC algorithm, *CMAC*, which is based on the *Coding* algorithm.

CMAC(ϵ), where $0 < \epsilon < 1$: We group slots into *Coding* phases, each consisting of $\mathcal{R}_\epsilon$ slots. At the beginning of every *Coding* phase, each node i that has an active $bcast(m)_i$ participates in $Coding(\epsilon)$ with packet m. Node i executes exactly one *Coding* phase, and then outputs $ack(m)_i$ at the end of the phase. However, if node i receives an $abort(m)_i$ from the environment before it performs $ack(m)_i$, it continues participating in the rest of the *Coding* phase but does not perform $ack(m)_i$.

Meanwhile, node i tries to receive packets from its neighbors, in every slot. It may receive a packet directly, without any collisions, or indirectly, by decoding collisions. When it receives any packet m' from a neighbor for the first time, it delivers that to its environment with a $rcv(m')_i$ event, at a real time before the time marking the end of the slot.

Note that, in a single slot, node i may receive several packets and deliver them to its environment, by decoding a collection of received collisions. Also note that node i may continue processing a packet for some time after it is aborted; thus, *CMAC* handles aborts differently from the *DMAC* algorithm in [1, 2]. Besides increasing the t_{abort} bound, this way of handling aborts admits the possibility that the environment may submit a new packet while node i is still transmitting on behalf of the aborted one. According to the rules of *CMAC*, node i will begin handling the new packet at the start of the next *Coding* phase.

We now give several lemmas expressing properties of $CMAC(\epsilon)$. These lemmas are analogous to some in [1]. The "executions" referred to here are executions of the composition $CMAC \| Env \| Net$, for any environment Env.

Lemma 4. *In every time-unbounded execution, the* Proximity*,* No duplicate receives*,* No receives after acks*,* No duplicate acks*, and* No acks after aborts *conditions are satisfied. Also, no rcv happens more than time $\mathcal{R}_\epsilon$ after a corresponding abort.*

The next lemma provides an absolute bound on acknowledgment time.

Lemma 5. *In every time-unbounded execution α, the following holds. Consider any $bcast(m)_i$ event in α, and suppose that α contains no $abort(m)_i$. Then an $ack(m)_i$ occurs by the end of the next Coding phase that begins after the $bcast(m)_i$.*

Proof. Immediate from the definition of *CMAC* □

The remaining properties are probabilistic. For these lemmas, we fix any environment Env and consider probabilities with respect to the unique probability distribution on executions of $CMAC \| Env \| Net$. The first lemma, which is analogous to Lemma 5.5 in [1], bounds the receive delay.

Lemma 6. *Let $i, j \in V$, i a neighbor of j. Let β be a closed execution that ends with a $bcast(m)_i$ event. Let cp be the first Coding phase that starts strictly after the $bcast(m)_i$.*
Define the following sets of time-unbounded executions that extend β: A, the executions in which no $abort(m)_i$ occurs, and B, the executions in which, by the end of coding phase cp, a $rcv(m)_j$ occurs.
If $Pr_\beta(A) > 0$, then $Pr_\beta(\bar{A} \cup B) \geq Pr_\beta(B|A) \geq 1 - \epsilon$.

Proof. The first inequality is easy to see, as in [1]. For the second, assume A, that is, no $abort(m)_i$ occurs. Let I be the set of neighbors of j participating in *Coding* phase cp. Since no $abort(m)_i$ occurs, $i \in I$, and so $|I| \geq 1$. Then, Lemma 3 implies that, with probability at least $1 - \epsilon$, a rcv_j for every packet $m_{i'}$, $i' \in I$ occurs in phase cp. In particular, $rcv(m)_j$ occurs. Therefore, $Pr_\beta(B|A) \geq 1 - \epsilon$, as needed. □

The second lemma, which is analogous to Lemma 5.6 in [1], bounds the acknowledgment delay and gives a probabilistic guarantee that acknowledgments are preceded by receives.

Lemma 7. *Let $i \in V$. Let β be any closed prefix of a time-unbounded execution that ends with a $bcast(m)_i$ event. Let cp be the first Coding phase that starts strictly after the $bcast(m)_i$.*
Define the following sets of time-unbounded executions that extend β: A, the executions in which no $abort(m)_i$ occurs, and B, the executions in which, by the end of coding phase cp, $ack(m)_i$ occurs and is preceded by $rcv(m)_j$ for every neighbor j of i.
If $Pr_\beta(A) > 0$, then $Pr_\beta(\bar{A} \cup B) \geq Pr_\beta(B|A) \geq 1 - \epsilon\Delta$.

Proof. As before, the first inequality is easy. Lemma 5 implies that $ack(m)_i$ occurs by the end of phase cp. For the $rcv(m)_j$ events, by Lemma 6, the probability that each individual $rcv(m)_j$ event occurs by the end of cp is at least $1 - \epsilon$. Then, using a union bound, the probability that all the $rcv(m)_j$ events occur by the end of cp is at least $1 - \epsilon\Delta$. □

The third lemma, which is analogous to Lemma 5.4 in [1], gives a bound for progress.

Lemma 8. *Let $j \in V$ and β be a closed execution that ends at time t. Let I be the set of neighbors of j that have active bcasts at the end of β, where $bcast(m_i)_i$ is the bcast at i. Suppose that I is nonempty. Suppose that no $rcv(m_i)_j$ occurs in β, for any $i \in I$. Let cp be the first Coding phase that starts strictly after time t.*
Define the following sets of time-unbounded executions that extend β: A, the executions in which no $abort(m_i)_i$ occurs for any $i \in I$; B, the executions in which, by the end of cp, at least one of the following occurs: a $rcv(m_i)_j$ for some $i \in I$, or a rcv_j for some packet whose bcast occurs after β; and C, the executions in which, by the end of cp, $ack(m_i)_i$ occurs for every $i \in I$.
If $Pr_\beta(A) > 0$, then $Pr_\beta(\bar{A} \cup B \cup C) \geq Pr_\beta(B \cup C|A) \geq 1 - \epsilon$.

Proof. The first inequality is easy. As shown in the proof of Lemma 5.4 in [1], we have

$$Pr_\beta(B \cup C|A) \geq Pr_\beta(B|\bar{C} \cap A),$$

so, for the first conclusion, it suffices to show that $Pr_\beta(B|\bar{C} \cap A) \geq 1 - \epsilon$. Thus, assume $\bar{C} \cap A$, that is, that by the end of cp, not every $i \in I$ has an $ack(m_i)_i$, and no $abort(m_i)_i$ occurs for any $i \in I$. Then some neighbor of j in I participates in phase cp. Let I' be the set of neighbors of j participating in cp. Note that every node in I' participates in all slots of phase cp, since no node stops participating part-way through the phase. Then $|I'| \geq 1$ and thus by Lemma 3, with probability at least $1 - \epsilon$, a rcv_j for every packet $m_{i'}$, $i' \in I'$ occurs in phase $\mathcal{P}$. Therefore,

$$Pr_\beta(B|\bar{C} \cap A) \geq 1 - \epsilon,$$

as needed. □

5.4 Implementing the Probabilistic MAC Layer

In this subsection, we fix ϵ, $0 < \epsilon < 1$, and fix t_{Cphase}, the time for a *Coding* phase, to be $\mathcal{R}_\epsilon$. We define the following parameter values: $f_{rcv} = f_{ack} = f_{prog} = 2t_{Cphase}$; $\epsilon_{rcv} = \epsilon_{prog} = \epsilon$; $\epsilon_{ack} = \epsilon\Delta$; and $t_{abort} = t_{Cphase}$.

Theorem 1. *CMAC(ϵ) implements the probabilistic MAC Layer with parameters as defined above.*

Proof. Similar to the proof of Theorem 5.7 in [1], using Lemmas 4-8. □

The following corollary follows directly from Lemma 2 and Theorem 1.

Corollary 1. *CMAC(ϵ) implements the probabilistic MAC layer with f_{rcv}, f_{ack}, f_{prog}, and $t_{abort} = \mathcal{O}\left(\Delta + \frac{\Delta}{c}\log(\frac{\Delta}{\epsilon})\right)$, $\epsilon_{rcv} = \epsilon_{prog} = \epsilon$, and $\epsilon_{ack} = \epsilon\Delta$.*

In the important case where c is large compared to Δ, this bound can be specialized and simplified as follows:

Corollary 2. *If $c = \Omega(\Delta)$ then CMAC(ϵ) implements the probabilistic MAC layer with f_{rcv}, f_{ack}, f_{prog}, and $t_{abort} = \mathcal{O}(\Delta + \log(\frac{1}{\epsilon}))$, $\epsilon_{rcv} = \epsilon_{prog} = \epsilon$, and $\epsilon_{ack} = \epsilon\Delta$. If in addition $\epsilon = \Omega(\frac{1}{c^\Delta})$ for a constant c, then the time bounds are all $O(\Delta)$.*

For comparison, the *DMAC* algorithm [1, 2] yields larger bounds of $f_{rcv} = f_{ack} = \mathcal{O}(\Delta\log(\frac{1}{\epsilon})\log(\Delta))$, with $\epsilon_{rcv} = \epsilon$ and $\epsilon_{ack} = \epsilon\Delta$. However, *DMAC* yields a smaller f_{prog} bound, of $\mathcal{O}(h\log(\Delta))$, with $\epsilon_{prog} = (\frac{7}{8})^h$, for any positive integer h. In the next section, we show how to reduce the f_{prog} bound to this level.

6 An Improved MAC Layer Implementation

In this section, we describe a second MAC layer implementation that achieves both the $\mathcal{O}\left(\Delta + \frac{\Delta}{c}\log(\frac{\Delta}{\epsilon})\right)$ receive and acknowledgment bounds of *CMAC* and the $\mathcal{O}(\log(\Delta))$ progress bounds of *DMAC*. The new algorithm essentially combines *CMAC* and *DMAC* using time-division multiplexing. *CMAC* is used to guarantee the receive and acknowledgment bounds, while *DMAC* guarantees the progress bound. We call the combined algorithm *DCMAC*.

6.1 The *DCMAC* Algorithm

Formally, *DCMAC* uses the *Coding* algorithm described in Section 5.2 and the *Decay* algorithm of [1, 2]. The *Decay* algorithm operates for exactly $\sigma = \lceil\log(\Delta + 1)\rceil$ slots, in which participating nodes transmit with successively doubling probabilities, starting with $\frac{1}{2^\sigma}$ and ending with $\frac{1}{2}$.

DCMAC(ϵ), where $0 < \epsilon < 1$: We use odd-numbered slots for *Decay* and even-numbered slots for *Coding*(ϵ). We group odd slots into *Decay* phases, each consisting of σ slots, and group even slots into *Coding* phases, each consisting of $\mathcal{R}_\epsilon$ slots. The two types of phases are not synchronized with respect to each other.

At the beginning of each *Decay* phase, each node i that has an active $bcast(m)_i$ executes *Decay* with packet m. At the beginning of each *Coding* phase, each node i that has an active $bcast(m)_i$ executes *Coding*(ϵ) with packet m and outputs $ack(m)_i$ at the end of that *Coding* phase.

If node i receives an $abort(m)_i$ or performs an $ack(m)_i$, it performs no further transmission on behalf of packet m in the odd slots; that is, it stops participating in a *Decay* phase as soon as an *abort* or *ack* happens. However, if node i receives an $abort(m)_i$ before it performs $ack(m)_i$, it continues participating in the rest of the *Coding* phase and does not perform $ack(m)_i$.

Meanwhile, node i keeps trying to receive, in every slot. In even slots, it may receive a packet directly, without collisions, or indirectly, by decoding collisions. In odd slots, it does not try to decode collisions, but just looks for packets that arrive directly. When node i receives any packet m' for the first time, in either

an odd or even slot, it delivers that to its environment with a $rcv(m')_i$ event, at a real time before the time marking the end of the slot.

Thus, as in the *CMAC* algorithm, node i may receive several packets in one slot, by decoding a collection of received collisions. Also note that node i may handle two different packets in consecutive odd and even slots, because of the different handling of aborts in the odd and even slots. However, i handles at most one packet in each slot, odd or even.

We now give lemmas expressing properties of *DCMAC*. These are analogous to those for *CMAC*, in Section 5.3, but now we consider executions of the composition $DCMAC \| Env \| Net$. The first four lemmas are very similar to before, but the fifth one, which deals with the progress bound, is somewhat different because it depends on *Decay* rather than *Coding*.

Lemma 9. *In every time-unbounded execution, the* Proximity*,* No duplicate receives*,* No receives after acks*,* No duplicate acks*, and* No acks after aborts *conditions are satisfied. Also, no rcv happens more than time* $2\mathcal{R}_\epsilon$ *after a corresponding abort.*

Lemma 10. *In every time-unbounded execution* α*, the following holds. Consider any* $bcast(m)_i$ *event in* α *and suppose that* α *contains no* $abort(m)_i$*. Then an* $ack(m)_i$ *occurs at the end of the Coding phase that begins after the* $bcast(m)_i$*.*

The remaining properties are probabilistic. Fix any environment *Env* and consider the unique probability distribution on executions of $CMAC \| Env \| Net$. The first lemma bounds the receive delay, and the second bounds the acknowledgment delay and gives a probabilistic guarantee that acknowledgments are preceded by receives. The proofs are similar to those of Lemmas 6 and 7, respectively.

Lemma 11. *Let* $i, j \in V$*,* i *a neighbor of* j*. Let* β *be a closed execution that ends with a* $bcast(m)_i$ *event. Let cp be the first Coding phase that starts strictly after the* $bcast(m)_i$*.*
Define the following sets of time-unbounded executions that extend β*:* A*, the executions in which no* $abort(m)_i$ *occurs, and* B*, the executions in which, by the end of coding phase cp, a* $rcv(m)_j$ *occurs. If* $Pr_\beta(A) > 0$*, then*

$$Pr_\beta(\bar{A} \cup B) \geq Pr_\beta(B|A) \geq 1 - \epsilon.$$

Lemma 12. *Let* $i \in V$*. Let* β *be any closed prefix of a time-unbounded execution that ends with a* $bcast(m)_i$ *event. Let cp be the first Coding phase that starts strictly after* $bcast(m)_i$*.*

Define the following sets of time-unbounded executions that extend β*:* A*, the executions in which no* $abort(m)_i$ *occurs, and* B*, the executions in which, by the end of coding phase cp,* $ack(m)_i$ *occurs and is preceded by* $rcv(m)_j$ *for every neighbor* j *of* i*. If* $Pr_\beta(A) > 0$*, then*

$$Pr_\beta(\bar{A} \cup B) \geq Pr_\beta(B|A) \geq 1 - \epsilon\Delta.$$

The final lemma gives the progress bound.

Lemma 13. *Let* $j \in V$ *and* h *be a positive integer. Let* β *be a closed execution that ends at time* t*. Let* I *be the set of neighbors of* j *that have active bcasts at the end of* β*, where* $bcast(m_i)_i$ *is the bcast at* i*. Suppose that* I *is nonempty. Suppose that no* $rcv(m_i)_j$ *occurs in* β*, for any* $i \in I$*. Let dp be the* h^{th} *Coding phase that starts strictly after time* t*.*

Define the following sets of time-unbounded executions that extend β*:* A*, the executions in which no* $abort(m_i)_i$ *occurs for any* $i \in I$*;* B*, the executions in which, by the end of Decay phase dp, at least one of the following occurs: a* $rcv(m_i)_j$ *for some* $i \in I$*, or a* rcv_j *for some packet whose bcast occurs after* β*; and* C*, the executions in which, by the end of Decay phase dp,* $ack(m_i)_i$ *occurs for every* $i \in I$*. If* $Pr_\beta(A) > 0$*, then*

$$Pr_\beta(\bar{A} \cup B \cup C) \geq Pr_\beta(B \cup C|A) \geq 1 - \left(\frac{7}{8}\right)^h.$$

Proof. Analogous to that of its counterpart, Lemma 5.4 in [1]. □

6.2 Implementing the Probabilistic MAC Layer

Fix ϵ, $0 < \epsilon < 1$, and fix t_{Dphase}, the time for a *Decay* phase, to be $\sigma = \lceil \log(\Delta + 1) \rceil$. Let h be any positive integer. We define parameter values: $f_{rcv} = f_{ack} = 4t_{Cphase}$; $f_{prog} = 2(h+1)t_{Dphase}$; $\epsilon_{rcv} = \epsilon$; $\epsilon_{ack} = \epsilon\Delta$; $\epsilon_{prog} = (\frac{7}{8})^h$; and $t_{abort} = 2t_{Cphase}$.

Theorem 2. *$DCMAC(\epsilon)$ implements the probabilistic MAC Layer with parameters as defined above.*

Proof. Similar to the proof of Theorem 5.7 in [1], using Lemmas 9-13. □

The following corollary follows directly from Lemma 2 and Theorem 2.

Corollary 3. *$DCMAC(\epsilon)$ implements the probabilistic MAC layer with f_{rcv}, f_{ack}, and $t_{abort} = \mathcal{O}\left(\Delta + \frac{\Delta}{c}\log(\frac{\Delta}{\epsilon})\right)$, $f_{prog} = \mathcal{O}(h\log(\Delta))$, $\epsilon_{rcv} = \epsilon$, $\epsilon_{ack} = \epsilon\Delta$, and $\epsilon_{prog} = (\frac{7}{8})^h$.*

Again, we specialize the bound to the case where c is large:

Corollary 4. *If $c = \Omega(\Delta)$, then $DCMAC(\epsilon)$ implements the probabilistic MAC layer with f_{rcv}, f_{ack}, and $t_{abort} = \mathcal{O}(\Delta + \log(\frac{1}{\epsilon}))$, $f_{prog} = \mathcal{O}(h\log(\Delta))$, $\epsilon_{rcv} = \epsilon$, $\epsilon_{ack} = \epsilon\Delta$, and $\epsilon_{prog} = (\frac{7}{8})^h$. If in addition $\epsilon = \Omega(\frac{1}{c^\Delta})$ for some constant c, then the bounds for f_{rcv}, f_{ack}, and t_{abort} are all $O(\Delta)$.*

These bounds compare favorably in all dimensions with those of *DMAC*.

7 Multi-Message Broadcast

The Multi-Message Broadcast (MMB) problem is widely studied in the distributed algorithms research community. In this problem, an arbitrary number of uniquely-identified messages originate at arbitrary nodes in the network, at arbitrary times; the problem is to deliver all messages to all nodes. We assume that each message fits in a single MAC-layer packet. An MMB protocol has an external interface by which it receives messages from its environment via $arrive(m)$ input events and delivers messages to the environment via $deliver(m)$ output events. We describe the Basic Multi-Message Broadcast ($BMMB$) protocol from [14, 15].

Basic Multi-Message Broadcast Protocol ($BMMB$): Every node i maintains a FIFO queue named *bcastq* and a set named *rcvd*. Both are initially empty. If node i does not have a pending MAC-layer transmission and *bcastq* is not empty, node i composes a packet containing the message m at the head of *bcastq*, removes m from *bcastq*, and passes the packet to the MAC layer for transmission, using a *bcast* output event. If node i receives an $arrive(m)$ input event, it immediately performs a $deliver(m)$ output and adds m to the back of *bcastq* and to the *rcvd* set. If node i receives a message m from the MAC layer, it first checks *rcvd*. If $m \in rcvd$ it discards the message. Otherwise, node i immediately performs a $deliver(m)$ output, and adds m to the back of *bcastq* and to the *rcvd* set.

We now consider executions of $BMMB$ composed with *DCMAC* and some environment that generates the messages. The following definitions are summarized from [1].

Definition 2 (Nice executions)**.** *A $bcast(m)$ event that occurs at node i at time t_0 in an execution is* nice *if the corresponding $ack(m)$ event occurs by time $t_0 + f_{ack}$ and is preceded by a corresponding $rcv(m)$ event at every neighbor j of i. An execution is* nice *if all bcast events in the execution are nice.*

Definition 3 (The set $overlap(m)$)**.** *Let α be a nice execution and m be a message such that $arrive(m)$ occurs in α. Then we define $overlap(m)$ to be the set of messages m' whose processing overlaps the interval between the $arrive(m)$ and the final $ack(m)$ event for m anywhere in the network. Formally, this means that an $arrive(m')$ event precedes the final $ack(m)$ event and the final $ack(m')$ event follows the $arrive(m)$ event.*

The following theorem follows from Theorem 7.20 of [1] and Corollary 3. It assumes an upper bound k on the total number of messages that arrive from the environment in any execution.

Theorem 3. *Let m be a message and ϵ be a real, $0 < \epsilon < 1$. Then $BMMB$ composed with $DCMAC(\frac{\epsilon}{2nk\Delta})$ guarantees that, with probability at least $1 - \epsilon$, the following hold: 1. α is a nice execution. 2. Suppose an $arrive(m)$ event occurs in α at node i at time t_0. Let $k' = |overlap(m)|$. Then $deliver(m)$ events occur at all other nodes in α by time $t_0 + \mathcal{O}\left(\left(D + \log(\frac{nk}{\epsilon})k'\right)\log(\Delta)\right) + (k'-1)\left(\Delta + \frac{\Delta}{c}\log(\frac{\Delta nk}{\epsilon})\right)$.*

8 Conclusions

We have presented a MAC layer design, *CMAC*, based on Analog Network Coding. We have proved its basic correctness and performance properties by showing that *CMAC* implements a formal probabilistic abstract MAC layer specification. This analysis shows that the new design improves on a traditional probabilistic retransmission algorithm *DMAC* in two of three performance metrics (the receive and acknowledgement delay bounds), while doing worse on one (the progress bound). However, a hybrid design combining *CMAC* and *DMAC* achieves the best of both algorithms.

In addition to providing an objective basis for comparing MAC layer designs, the abstract MAC layer allows us to combine complexity bounds for MAC layer designs with complexity bounds for higher-level protocols that run over MAC layers. To illustrate this, we showed how to combine a high-level global broadcast protocol with an ANC-based MAC layer, and easily obtain complexity bounds for the combination.

Future work will include explicit consideration of other metrics, such as capacity, and comparison of ANC-based strategies with more different kinds of MAC-layer designs. It would also be interesting to study how the results can be extended to accommodate packet erasures. The development of the MAC we have provided here has considered physical-layer ANC and not higher layer network coding. The use of our MAC and related approaches could be considered in the presence of transport-layer network coding, since the two network coding approaches are compatible [12]. Moreover, our work here has considered coding only for local node interactions but invites questions of considering it for larger, multihop networks, particularly when transport-layer coding is integrated. The interaction between the broadcast MAC and network coding at the transport layer has been shown to provide, in a simple, opportunistic, fashion, considerable gains in a multihop setting [20]. Moreover, a MAC-aware coding approach in multihop networks can lead to even more considerable gains [21].

References

[1] M. Khabbazian, D. Kowalski, F. Kuhn, and N. Lynch. The cost of global broadcast using abstract MAC layers. *MIT Technical Report (MIT-CSAIL-TR-2010-005)*, February 2010.

[2] M. Khabbazian, D. Kowalski, F. Kuhn, and N. Lynch. The cost of global broadcast using abstract MAC layers. *Submitted to the International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 2010.

[3] N. Abramson. ALOHA System-Another alternative for computer communications. In *Proceedings of Fall Joint Computer Conference*, 1970.

[4] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang. MACAW: A medium access protocol for wireless lans. In *The Proceedings of the ACM SIGCOMM Conference*, 1994.

[5] D. Bertsekas and R. Gallager. *Data networks (2nd ed.)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1992.

[6] S. Gollakota and D. Katabi. ZigZag decoding: Combating hidden terminals in wireless networks. In *The Proceedings of the ACM SIGCOMM Conference*, volume 38, pages 159–170, 2008.

[7] S. Katti, S. Gollakota, and D. Katabi. Embracing wireless interference: Analog network coding. In *The Proceedings of the ACM SIGCOMM Conference*, pages 397–408, 2007.

[8] I. Maric, A. Goldsmith, and M. Médard. Analog network coding in the high SNR regime. *invited paper, ITA Workshop*, 2010.

[9] I. Maric, A. Goldsmith, and M. Médard. Analog network coding in the high SNR regime. In *The Proceedings of Wireless Network Coding Workshop*, 2010.

[10] S. Borade, L. Zheng, and R. Gallager. Amplify-and-forward in wireless relay networks: Rate, diversity and network size. *IEEE Transactions on Information Theory*, 53(10):3302–3318, 2007.

[11] H. Bolcskei, R. U. Nabar, O. Oyman, , and A. J. Paulraj. Capacity scaling laws in MIMO relay networks. *IEEE Transactions on Wireless Communications*, 5(6):1433–1443, 2006.

[12] A. ParandehGheibi, J. Sundararajan, and M. Médard. Collision helps - algebraic collision recovery for wireless erasure networks. In *The Proceedings of Wireless Network Coding Workshop*, 2010.

[13] D. Wischik and D. Shah. MAC3: Medium access coding & congestion control. *Presentation at the Newton Institute for Mathematics*, 2010.

[14] F. Kuhn, N. Lynch, and C. Newport. The abstract MAC layer. *MIT Technical Report (MIT-CSAIL-TR-2009-021)*, May 2009.

[15] F. Kuhn, N. Lynch, and C. Newport. The abstract MAC layer. In *The Proceedings of the International Symposium on Distributed Computing*, pages 48–62, 2009. To appear in a special issue of Distributed Computing.

[16] A. Cornejo, N. Lynch, S. Viqar, and L. Welch. A neighbor discovery service using an abstract MAC layer. In *The Proceedings of the Annual Allerton Conference on Communication, Control, and Computing*, pages 1460–1467, 2009.

[17] A. Cornejo, S. Viqar, and J. Welch. Reliable neighbor discovery for mobile ad hoc networks. *Submitted to the International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 2010.

[18] Sayan Mitra. *A Verification Framework for Hybrid Systems*. PhD thesis, Massachusetts Institute of Technology, 2007.

[19] Dilsun K. Kaynar, Nancy Lynch, Roberto Segala, and Frits Vaandrager. *Theory of Timed I/O Automata*. Synthesis Lectures on Computer Science. Morgan-Claypool Publishers, May 2006. Also, revised and shortened version of Technical Report MIT-LCS-TR-917a (from 2004), MIT Laboratory for Computer Science, Cambridge, MA.

[20] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft. XORs in the air: Practical wireless network coding. *IEEE/ACM Transactions on Networking*, 16(3):497–510, 2008.

[21] F. Zhao and M. Médard. On analyzing and improving COPE performance. In *ITA Workshop*, January 2010.

A Binomial Random Variables

Lemma 14. *Let X be a binomial random variable. Then* $\Pr(X \leq \lceil \mathbb{E}[X] \rceil) \geq \frac{1}{2}$ *and* $\Pr(X > \lceil \mathbb{E}[X] \rceil) \leq \frac{1}{2}$.

Proof. This follows from the fact that the median of X is either $\lfloor \mathbb{E}[X] \rfloor$ or $\lceil \mathbb{E}[X] \rceil$. □

Lemma 15. *Let X be a random variable with binomial distribution $B(n,p)$, where n is a positive integer and p is a real, $0 < p < 1$. Then* $\Pr(X = 1) \geq \Pr(X = 0)(1 - \Pr(X = 0))$.

Proof. Bernoulli's inequality implies that $\Pr(X = 0) = (1-p)^n \geq 1 - np$, so $1 - \Pr(X = 0) \leq np$. Therefore,

$$\begin{aligned}
\Pr(X = 1) &= np(1-p)^{n-1} \\
&\geq np(1-p)^n \\
&= np \Pr(X = 0) \\
&\geq (1 - \Pr(X = 0)) \Pr(X = 0).
\end{aligned}$$

□

B Sequences of Geometric Random Variables

Lemma 16. *Let k be a positive integer. Let X_i, $i \in [k]$, be independent geometric random variables, where X_i has parameter p_i. Define $X = \sum_{i=1}^{k} X_i$, $\mu = \mathbb{E}[X] = \sum_{i=1}^{k} 1/p_i$, and $p_{\min} = \min_{i \in [k]} p_i$.*
Then for every $\epsilon > 0$,

$$\Pr\left(X \geq 2\mu + \frac{4\ln(1/\epsilon)}{p_{\min}}\right) \leq \epsilon.$$

Proof. We prove this using a Chernoff-type argument. For every $\gamma > 0$ and every $t \geq 0$, we have

$$\begin{aligned}
\Pr(X \geq t) &= \Pr\left(e^{\gamma X} \geq e^{\gamma t}\right) \\
&\leq \frac{\mathbb{E}\left[e^{\gamma X}\right]}{e^{\gamma t}} \\
&= \frac{\prod_{i=1}^{k} \mathbb{E}\left[e^{\gamma X_i}\right]}{e^{\gamma t}},
\end{aligned}$$

where the inequality follows by applying Markov's inequality. We first derive a bound on $\mathbb{E}[e^{\gamma X_i}]$. Assume that $\gamma = p_{\min}/4 \leq 1/4$, yielding $e^{\gamma} - 1 \leq p_{\min}/3$. Then for every $i \in [k]$, we get

$$\begin{aligned}
\mathbb{E}\left[e^{\gamma X_i}\right] &= \sum_{s=1}^{\infty} p_i (1-p_i)^{s-1} e^{\gamma s} \\
&= \frac{p_i e^{\gamma}}{1 - (1-p_i)e^{\gamma}} \\
&\overset{(1+x \leq e^x)}{\leq} e^{\frac{e^{\gamma}-1}{p_i e^{\gamma} - (e^{\gamma}-1)}} \\
&\overset{(e^{\gamma}-1 \leq \frac{p_{\min}}{3})}{\leq} e^{\frac{p_{\min}}{3p_i e^{\gamma} - p_{\min}}} \\
&\overset{(e^{\gamma}>1)}{\leq} e^{\frac{p_{\min}}{2p_i}}.
\end{aligned}$$

Plugging this into the previous inequality, we obtain

$$\Pr(X \geq t) \leq e^{\frac{p_{\min}}{2} \cdot \sum_{i=1}^{k} \frac{1}{p_i} - \gamma t} = e^{\frac{p_{\min}}{2} \cdot \left(\mu - \frac{t}{2}\right)}.$$

The lemma now follows by using $t = 2\mu + 4\ln(1/\epsilon)/p_{\min}$. □

Lemma 17. *Let k be a positive integer and q, p, and ϵ be positive reals less than one. Let X_i, $i \in [k]$, be independent geometric random variables, where X_i has parameter $p_i = q(1-(1-p)^{k+1-i})$. Define $X = \sum_{i=1}^{k} X_i$. Then*

$$\Pr\left(X \leq \frac{2e}{(e-1)q}\left(k + \frac{\ln(k+1) + 2\ln(1/\epsilon)}{p}\right)\right) \geq 1 - \epsilon.$$

Proof. We show that

$$\mathbb{E}[X] \leq \frac{e}{(e-1)q} \cdot \left(k + \frac{\ln(k+1)}{p}\right). \tag{1}$$

The lemma then follows directly from Lemma 16 because the smallest probability p_i is $p_k = pq$. In order to prove (1), we partition the variable X_i into two parts. For every x, we have $1 - x \leq e^{-x}$ and therefore we get

$$p_i \geq q\left(1 - e^{-p(k+1-i)}\right) \tag{2}$$

for every $i \in [k]$. Let $I_1 \subset [k]$ be the set of $i \in [k]$ for which $p(k+1-i) \geq 1$, and let $I_2 = [k] \setminus I_1$ be the set of $i \in [k]$ for which $p(k+1-i) < 1$. For every $i \in I_1$, (2) yields $p_i \geq q(1-1/e)$. For $x < 1$, we have $e^{-x} \leq 1 - \frac{e-1}{e} \cdot x$ and therefore we obtain $p_i \geq \frac{e-1}{e} \cdot qp(k+1-i)$ for every $i \in I_2$. Together, these yield

$$\begin{aligned}
\mathbb{E}[X] &= \sum_{i=1}^{k} \frac{1}{p_i} \\
&= \sum_{i \in I_1} \frac{1}{p_i} + \sum_{i \in I_2} \frac{1}{p_i} \\
&\leq \frac{e}{(e-1)q} \cdot \left(k + \frac{1}{p} \cdot \sum_{j=1}^{k} \frac{1}{j}\right) \\
&\leq \frac{e}{(e-1)q} \cdot \left(k + \frac{H(k)}{p}\right),
\end{aligned}$$

where $H(k)$ denotes the harmonic sum $\sum_{j=1}^{k} 1/j$. Inequality (2) and thus the lemma now follow from the fact that $H(k) \leq \ln(k+1)$. □

C Linear Independence of Random Vectors

Lemma 18. *Let d and w be positive integers, $1 \leq k \leq w$. Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{d-1} \in \mathbb{C}^w$ be $d-1$ linearly independent vectors. Let p be a positive real, $p \leq 1/2$, c be an integer, where $c \geq 4$ and $wp \leq c/2$, and $\boldsymbol{z} = (z_1, \ldots, z_w) \in \left\{\mathbb{C} \setminus \{0\}\right\}^w$ be a vector.*
Construct a random vector $\boldsymbol{y} = (y_1, \ldots, y_w) \in \mathbb{C}^w$ as follows. For every $i \in [w]$, independently, set $y_i = z_i$ with probability p and $y_i = 0$ with probability $1-p$.
Then the probability that $\boldsymbol{y}$ has at most c non-zero coordinates and that $\boldsymbol{y}$ is linearly independent of all the vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{d-1}$ is at least

$$p_d := \frac{1}{7} \cdot \left(1 - (1-p)^{w-d+1}\right). \tag{3}$$

Proof. Let $\mathcal{C}$ be the event that $\boldsymbol{y}$ has at most c non-zero coordinates and let $\mathcal{I}$ be the event that $\boldsymbol{y}$ is linearly independent of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{d-1}$. We aim to show that $\Pr(\mathcal{C} \cap \mathcal{I}) \geq p_d$.

Let $W \subset [w]$ be a set of size $|W| = d-1$ such that the vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{d-1}$ restricted to only the coordinates in W are still linearly independent. Note that such a set W always exists and that W can be obtained, e.g., by Gaussian elimination. By the choice of W, there exist $\alpha_1, \ldots, \alpha_{d-1} \in \mathbb{C}$ such that for the linear combination

$$\boldsymbol{v} = (v_1, \ldots, v_{d-1}) = \sum_{i=1}^{d-1} \alpha_i \cdot \boldsymbol{x}_i, \tag{4}$$

we have $v_i = -y_i$ for all $i \in W$. The vector $\boldsymbol{y}$ is linearly independent of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{d-1}$ if and only if $\boldsymbol{y} + \boldsymbol{v} \neq \boldsymbol{0}$, i.e., there is a coordinate $j \notin W$ such that $y_j + v_j \neq 0$. We define three random variables $N_1, N_2, N_2' \subseteq [w]$ as follows:

$$\begin{aligned} N_1 &= |\{i | i \in W, y_i \neq 0\}, \\ N_2 &= |\{i | i \notin W, y_i \neq 0\}|, \\ N_2' &= |\{i | i \notin W, y_i + v_i \neq 0\}|. \end{aligned}$$

Notice that $N_1 + N_2$ is the number of non-zero elements of $\boldsymbol{y}$, i.e., event $\mathcal{C}$ occurs if and only if $N_1 + N_2 \leq c$. Furthermore, event $\mathcal{I}$ occurs if and only if $N_2' > 0$. Thus, $\Pr(\mathcal{C} \cap \mathcal{I}) = \Pr(N_2' > 0 \wedge N_1 + N_2 \leq c)$. Every element y_i of $\boldsymbol{y}$ is zero with probability $(1-p)$, and is non-zero with probability p. Hence, the random variables N_1 and N_2 have binomial distributions with expected values $\mu_1 = (d-1)p$ and $\mu_2 = (w-d+1)p$, respectively. Note that

$$\begin{aligned} \lceil \mu_1 \rceil + \lceil \mu_2 \rceil &\leq (d-1)p + (w-d+1)p + 2 \\ &= wp + 2 \\ &\leq \frac{c}{2} + 2 \\ &\leq c, \end{aligned}$$

because, by the lemma's statement, $wp \leq \frac{c}{2}$ and $c \geq 4$. Thus, we have:

$$\begin{aligned} & Pr\left(N_2' > 0 \wedge N_1 + N_2 \leq c\right) \\ & \qquad \geq Pr\left(N_2' > 0 \wedge N_2 \leq \lceil \mu_2 \rceil \wedge N_1 \leq \lceil \mu_1 \rceil\right) \\ & \qquad = Pr\left(N_2' > 0 \wedge N_2 \leq \lceil \mu_2 \rceil | (N_1 \leq \lceil \mu_1 \rceil)\right) \cdot Pr\left(N_1 \leq \lceil \mu_1 \rceil\right) \\ & \qquad \geq \frac{1}{2} Pr\left(N_2' > 0 \wedge N_2 \leq \lceil \mu_2 \rceil | (N_1 \leq \lceil \mu_1 \rceil)\right), \end{aligned} \tag{5}$$

where the first inequality is by the fact that $\lceil \mu_1 \rceil + \lceil \mu_2 \rceil \leq c$ (see (5)) and the last inequality is by Lemma 14.

Claim 1: $\Pr\left(N_2' = 0 | N_1 \leq \lceil \mu_1 \rceil\right) \leq \Pr\left(N_2 = 0\right).$

Proof of Claim 1:
Recall that $N_2' = 0$ iff $y_j + v_j = 0$ for all $j \in [w] \setminus W$. The values v_j for $j \notin W$ are determined by the vectors $\boldsymbol{x}_i$ and y_j for $j \in W$. Hence, the values y_j for $j \notin W$ are independent of the values v_j, $j \notin W$. We define $F = \{j \in [w] \setminus W | v_j \neq 0\}$. For $j \notin F$, $v_j = 0$ and therefore $y_j + v_j = 0$ iff $y_j = 0$, which happens with probability $1-p$. For $j \in F$, we can only have $y_j + v_j = 0$ if $y_j \neq 0$, which happens with probability p. We

therefore obtain

$$\begin{aligned}
&\Pr\left(N_2' = 0 \middle| N_1 \le \lceil \mu_1 \rceil\right) \\
&= \sum_{F' \subseteq [w] \setminus W} \Pr\left(N_2' = 0 | N_1 \le \lceil \mu_1 \rceil \wedge |F| = f\right) \Pr(|F| = f | N_1 \le \lceil \mu_1 \rceil) \\
&\le \sum_{F' \subseteq [w] \setminus W} \Pr\left(\bigwedge_{j \notin F'} (y_i = 0) \wedge \bigwedge_{j \in F'} (y_j \ne 0) | N_1 \le \lceil \mu_1 \rceil \wedge F = F'\right) \Pr(F = F' | N_1 \le \lceil \mu_1 \rceil) \\
&= \sum_{F' \subseteq [w] \setminus W} \Pr\left(\bigwedge_{j \notin F'} (y_i = 0) \wedge \bigwedge_{j \in F'} (y_j \ne 0) | F = F'\right) \Pr(F = F' | N_1 \le \lceil \mu_1 \rceil) \\
&= \sum_{F' \subseteq [w] \setminus W} p^{|F'|} (1-p)^{w-d+1-|F'|} \cdot \Pr(F = F' | N_1 \le \lceil \mu_1 \rceil) \\
&\le \sum_{F' \subseteq [w] \setminus W} (1-p)^{w-d+1} \cdot \Pr(F = F' | N_1 \le \lceil \mu_1 \rceil) \\
&= (1-p)^{w-d+1} \\
&= \Pr\left(N_2 = 0\right),
\end{aligned}$$

where the second inequality follows from $p \le \frac{1}{2}$.

End of Claim 1.

We have:

$$\begin{aligned}
&\Pr\left(N_2' > 0 \wedge N_2 \le \lceil \mu_2 \rceil | N_1 \le \lceil \mu_1 \rceil\right) \\
&= 1 - \Pr\left(N_2' = 0 \vee N_2 > \lceil \mu_2 \rceil | N_1 \le \lceil \mu_1 \rceil\right) \\
&\ge 1 - \Pr\left(N_2' = 0 | N_1 \le \lceil \mu_1 \rceil\right) - \Pr\left(N_2 > \lceil \mu_2 \rceil | N_1 \le \lceil \mu_1 \rceil\right) \\
&= 1 - \Pr\left(N_2' = 0 | N_1 \le \lceil \mu_1 \rceil\right) - \Pr\left(N_2 > \lceil \mu_2 \rceil\right) \\
&\ge 1 - \Pr\left(N_2 = 0\right) - \Pr\left(N_2 > \lceil \mu_2 \rceil\right),
\end{aligned} \tag{6}$$

where the last inequality is by Claim 1. Using Lemma 14, we have $\Pr\left(N_2 > \lceil \mu_2 \rceil\right) \le \frac{1}{2}$. Thus, by (6), we get

$$\Pr\left(N_2' > 0 \wedge N_2 \le \lceil \mu_2 \rceil | N_1 \le \lceil \mu_1 \rceil\right) \ge \frac{1}{2} - \Pr\left(N_2 = 0\right). \tag{7}$$

Also, since $\lceil \mu_2 \rceil \ge 1$, by (6), we get

$$\Pr\left(N_2' > 0 \wedge N_2 \le \lceil \mu_2 \rceil | N_1 \le \lceil \mu_1 \rceil\right) \ge \Pr\left(N_2 = 1\right). \tag{8}$$

If $\Pr(N_2 = 0) \le \frac{2}{7}$, we have

$$\frac{1}{2} - \Pr(N_2 = 0) \ge \frac{2}{7}\left(1 - \Pr(N_2 = 0)\right).$$

Otherwise, $\Pr(N_2 = 0) > \frac{2}{7}$ and hence, by Lemma 15, we get

$$\Pr(N_2 = 1) \ge \frac{2}{7}(1 - \Pr(N_2 = 0)).$$

Therefore, using (7), (8), we have

$$\Pr\left(N_2' > 0 \wedge N_2 \le \lceil \mu_2 \rceil | N_1 \le \lceil \mu_1 \rceil\right) \ge \frac{2}{7}(1 - \Pr(N_2 = 0)). \tag{9}$$

Combining (5) and (9), we get

$$\begin{aligned}\Pr(\mathcal{C} \cap \mathcal{I}) &= \Pr\left(N_2' > 0 \wedge N_1 + N_2 \leq c\right) \\ &\geq \frac{1}{2} \Pr\left(N_2' > 0 \wedge N_2 \leq \lceil \mu_2 \rceil | N_1 \leq \lceil \mu_1 \rceil\right) \\ &\geq \frac{1}{2} \cdot \frac{2}{7}(1 - \Pr(N_2 = 0)) \\ &= \frac{1}{7}(1 - \Pr(N_2 = 0)) \\ &= \frac{1}{7}(1 - (1-p)^{w-d+1}),\end{aligned}$$

which concludes the proof. □

D Proof of Lemma 1

Proof. Instead of fixing the number of rows of B, assume that we construct B by adding rows until there are w rows with at most c non-zero entries that span all of $\mathbb{R}^w$. The lemma then follows by showing that with probability at least $1 - \epsilon$, the resulting matrix has at most l rows.

For each $d \in [w]$, we define random variables X_d and Y_d. Let X_d be the smallest integer such that the sub-matrix spanned by the rows with at most c non-zero entries among the first X_d rows of B has rank d. Further, we define $X_0 = 0$ and $Y_d = X_d - X_{d-1}$.

The non-zero rows of B are chosen in the same way as vector $\boldsymbol{y}$ in the statement of Lemma 18. Therefore by Lemma 18, for each $d \geq 0$ and row $i > X_{d-1}$, if row i is not set to 0 in the final step of the construction of B, the probability that row i contains at most c non-zero entries and is independent of the first X_{d-1} rows is at least $p_d = \frac{1}{7}\left(1 - (1-p)^{w-d+1}\right)$. Note that different rows are independent and that this is also independent of X_{d-1} and of the content of the first X_{d-1} rows. Thus, the random variables Y_d are dominated by independent geometric random variables Z_d with parameter $(1-p)p_d$. By applying Lemma 17 (using $q = (1-p)/7$) and the assumption on l, we then get

$$\Pr(X_d \leq l) = \Pr\left(\sum_{i=1}^{d} Y_d \leq l\right) \leq \Pr\left(\sum_{i=1}^{d} Z_d \leq l\right) \leq 1 - \epsilon.$$

This completes the proof. □