Online XOR Packet Coding: Efficient Single-Hop Wireless Multicasting with Low Decoding Delay✩,✩✩

Jalaluddin Qureshi, Chuan Heng Foh, Jianfei Cai
CeMNet, School of Computer Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798.

Abstract

In this paper we present a cross-layer solution to the problem of unreliability in IEEE 802.11 wireless multicast network, where an Access Point (AP) is multicasting a data file to a group of receivers over independent wireless erasure channels. We first present a practical scheme for collecting feedback frames from the receivers by means of simultaneous acknowledgement (ACK) frames collision. Based on these feedback frames, we design an online linear XOR coding algorithm to retransmit the lost packets. Through simulation results we first show that our proposed coding algorithm outperforms all the existing XOR coding algorithms in terms of retransmission rate. We further show that our proposed coding algorithm has the lowest average decoding delay of all the known network coding schemes. XOR coding and decoding only requires addition over GF(2), hence it enjoys lower encoding in terms of retransmission rate. We further show that our proposed coding algorithm has the lowest average decoding delay of all the known network coding schemes. XOR coding and decoding only requires addition over GF(2), hence it enjoys lower encoding and decoding computational complexities. Because of these features such an online XOR coding algorithm is also of interest for delay-sensitive applications such as multicast audio video (AV) streaming, and in battery constrained devices such as smartphones.

Keywords: Coding algorithms; IEEE 802.11 wireless multicasting; Reliability; Retransmission rate; Average decoding delay;

1. Introduction

Recent advancements in IEEE 802.11 [3] wireless local area networks (WLANs) has made ubiquitous access of multimedia content over WiFi one step closer to realization. Efficient access to the multimedia content on the wireless networks, especially real-time applications such as webcast, relies on the multicast transmission [5, 35]. Multicasting is an efficient technique to disperse common information to multiple receivers, and takes advantage of the broadcast nature of wireless transmission.

The IEEE 802.11 WLANs contention based medium access control (MAC) protocol inevitably introduces transmission collisions when simultaneous transmissions occurs, and therefore enforces an explicit positive acknowledgement (ACK) frame by the receiver to detect the successful transmission by the access point (AP) for unicast transmissions. However, this rule to achieve packet reliability fails for multicast transmission, as this will cost large control overhead [6, 10] to guarantee collision-free receipt of all the ACK frames using the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), and the optional Request to Send/ Clear to Send (RTS/CTS) control frames, implemented by the Distributed Coordination Function (DCF) at the MAC sub-layer. Legacy 802.11 multicast transmission is a “no-ACK, no-retransmission” scheme, in which the AP transmits the data packet and waits for the channel to be free, conforming only to the CSMA/CA access procedure, before transmitting the next packet. Current 802.11-2012 MAC protocol does not any provide MAC level recovery for multicast transmission by default.

The urgency to have a reliable multicast transmission scheme for 802.11 wireless networks is reflected by the recent formation of the IEEE working group, called Task Group aa (TGaa) [10]. The objective of TGaa is to make amendments to the current IEEE 802.11 standard by improving the reliability and adaptability of the multicast audio video (AV) service, while maintaining full compatibility with legacy devices. In its draft of future standard amendments, the group has clearly identified the need for policies which will improve multicast mechanism at the MAC layer and provide reliable acknowledgement of data packets for multicast transmissions.

In the recently proposed 802.11aa-2012 amendment [4], packet acknowledgement for multicast transmission is implemented using the Directed Multicast Service (DMS) and Groupcast with Retries (GCR) BlockAck procedures. For a multicast network of size N, DMS implements multicast reliability by turning a multicast transmission into N-unicast transmissions. The AP transmits the data packet as an individually addressed packet to a receiver $R_i$ and then waits for an ACK, if no ACK is received, the AP continues to retransmit the same packet to receiver $R_i$ until it receives an ACK from $R_i$. It then repeats this cycle with all the N receivers. For GCR BlockAck implementation, in each cycle the AP transmits arbitrary number of data packets and then implements BlockAck Request/BlockAck procedure with all the N receivers individually. Clearly the DMS and GCR BlockAck schemes are not scalable for large multicast networks.

✩This paper is an extension of the previous conference publication [1] of the authors, supplemented with results from [2]. The corresponding author of the paper is J. Qureshi, jalal0001@e.ntu.edu.sg.
✩✩The accepted version of the paper can be found from https://doi.org/10.1016/j.comcom.2013.09.006. © 2018. This manuscript version is made available under the CC-BY-NC-ND 4.0 license.

Preprint submitted to Computer Communications February 18, 2014
It remains important to review the existing concepts, and improve the operation by breaking the traditional rule of collision-free ACK transmissions. In this paper, we improve the reliability of wireless multicast transmission by using a mechanism whereby all receivers which have received the data packet acknowledge the multicast transmission simultaneously, causing a collision. We then show a collision coding scheme that enables the AP to decode the collided ACKs. Our collision coding scheme is based on the seminal work on physical-layer network coding (PNC) [8], in which the authors presented a study on turning a collision of two simultaneous wireless transmissions into a useful transmission. In brief, two simultaneous wireless transmissions that are added at the electromagnetic wave level can be decoded and mapped to produce an outcome such that the relationship between the transmitted and the decoded binary information follows the exclusive or (XOR) principle.

The next problem we need to consider once the AP has the packet feedback information is how to retransmit the identified lost packets efficiently. Network coding has been shown to be a throughput efficient scheme to improve the performance of wireless multicast transmission [12]. Such opportunities to code multiple packets for transmission arises due to the independent Bernoulli packet loss model inherent in wireless channels [9]. Therefore for an AP multicasting packets $c_1$ and $c_2$ to receivers $R_1$ and $R_2$ as shown in Figure 1, despite the broadcast nature of wireless transmission, $c_1$ may be received by $R_1$ only while $c_2$ may be received by $R_2$ only. However unlike the traditional Automatic Repeat reQuest (ARQ) retransmission scheme where both the packets are retransmitted separately in two different time slots, in a network coding based retransmission scheme packets $c_1$ and $c_2$ are XOR coded as $c_1 \oplus c_2$ and transmitted in one time slot, which the receivers can then use to recover their lost packets. This example illustrates the effectiveness of network coding to reduce the number of retransmissions from two time slots to one time slot.

For a given multicast network, with $N$ receivers and a packet batch size of $M$ packets being transmitted over independent and identical erasure channels, it has been shown that it is a NP-complete problem to minimize the number of transmissions when linearly coding packets over $GF(2)$ (also known as XOR coding) [16, 26]. This calls in for efficient heuristic coding algorithm to minimize the number of transmissions. To this end, there exist a rich literature [12, 24, 25, 27, 28, 29] of heuristic algorithms to minimize the number of transmissions in a multicast network using $GF(2)$ linear coding. Unfortunately the throughput performance of these proposed coding algorithms significantly falls short of the optimal performance. While it is possible to get a throughput optimal solution when coding over larger finite field size $GF(2^q)$, where $q \geq \lceil \log_2 N \rceil$ [26, 31], coding packets over large field size has a tradeoff cost of higher encoding and decoding computational complexities [32], and high encoding and decoding throughput [34], which limits its implementation for delay-sensitive applications and on battery and processor constrained devices.

Average decoding delay is another metric of interest when designing coding algorithm for delay-sensitive multicast applications. For example, for a receiver which has packet $c_1$ and wants packets $c_2$ and $c_3$, coded packets $c_2 \oplus c_3$ and $c_1 \oplus c_3$ are both linearly independent with respect to $c_1$, however only the latter coded packet can be instantly decoded by the receiver. The problem of minimizing the average decoding delay using linear network coding for a wireless multicast network is a NP-hard problem [17] even under the offline channel model. Several heuristic algorithms [19, 20, 21] have also been proposed with the objective of minimizing the average decoding delay.

For our proposed online XOR coding algorithm rather than addressing a single performance metric, we simultaneously address the problem of minimizing the number of transmissions, average decoding delay, encoding and decoding throughput, and encoding and decoding complexities, using polynomial time algorithm. We show that our proposed algorithm is the best performing XOR coding algorithm, significantly outperforming all the previously published XOR coding algorithms. We further show that our proposed coding algorithm also incurs the lowest average decoding delay. Interestingly, our proposed algorithm enjoys these advantages while preserving the benefit of low encoding and decoding computational complexities of encoding and decoding over $GF(2)$, and hence also enjoys lower encoding and decoding throughput.

We catalog the findings of our work as follow. We first propose a PNC based mechanism to simultaneously collect multiple ACK frames in Section 2. We then present the system model, related bibliography and problem formulation in Section 3. With the system modeled, we derive the expected decoding delay of RLNC, and demonstrate the encoding and decoding computational complexity in Section 4. In Section 5 we survey the various coding algorithms presented in literature for multicast transmission. We then present our proposed coding scheme, BENEFIT in Section 6 and the simulation results in Section 7. We finally conclude the main results of our work in Section 8.

1 An offline channel model is defined as a channel in which the future packet reception status of all the $N$ receivers is known to the AP a priori.
2. Reliable Wireless Multicast Using ACKs

The concept of PNC was first introduced by Zhang et al. [8], in which the authors proposed a frame-based decode-and-forward strategy for packet forwarding. In their scenario, two neighboring nodes transmit simultaneously to a common receiver. Assuming perfect transmission synchronization at the physical layer, then based on the additive nature of simultaneously arriving electromagnetic waves (EM), the receiver detects a single collided signal which is the sum of the two transmitted signals. Using a suitable mapping scheme, they show that for certain modulation schemes, there exists a mapping such that the relationship between the two transmitted binary bits and the decoded binary bit follow the XOR principle.

We here revisit the PNC operation and its mapping scheme to achieve the XOR principle. Consider two senders, \( R_1 \) and \( R_2 \), and a common receiver \( R_0 \). Let \( s_1 \) and \( s_2 \) be the binary bit transmitted by \( R_1 \) and \( R_2 \) at a particular time \( t \) respectively, and \( s_0 \) be the decoded binary bit. Based on binary phase shift keying (BPSK) modulation, the received signal \( r_0(t) \) is given as,

\[
r_0(t) = a_1 \cos(\omega t) + a_2 \cos(\omega t) = (a_1 + a_2) \cos(\omega t),
\]

where \( a_1 \) and \( a_2 \) are the transmitted amplitudes, and \( \omega \) is the carrier frequency. For BPSK, we have \( a_j = 2s_j - 1, j \in \{1, 2\} \). At \( R_0 \), a scheme (see Table 1) that maps a strong energy signal to binary 0 (i.e. \( |a_1 + a_2| = 2 \)) and a weak energy signal (i.e. \( a_1 + a_2 = 0 \)) to binary 1 can be applied which gives,

\[
s_0 = s_1 \oplus s_2.
\]

Further PNC studies on the penalty of carrier-phase synchronization errors, carrier frequency synchronization errors, and time synchronization errors are given in [8, 39], with results showing the practical potential of PNC.

2.1. Collision of Multiple BPSK Modulated Transmissions

Considering a collision consisting of an arbitrary number of simultaneous transmissions, according to Table 1, retaining the XOR principle for the decoding requires precise detection of signal energy for the PNC mapping, which reduces its robustness. Based on the additive nature of EM waves, considering the BPSK modulation scheme, it is not difficult to see that the BPSK demodulation process follows the majority principle.

To illustrate this, consider a simple example of three transmitters, \( R_i, i \in \{1, 2, 3\} \), and one common receiver, \( R_0 \). Let \( a_i \) be the amplitude of the BPSK signals corresponding to the transmitted binary information, and \( a_0 \) be the detected amplitude of the BPSK signals. Based on the additive nature of EM waves, we have \( a_0 = a_1 + a_2 + a_3 \). Considering a common design of a BPSK demodulator with a matched filter and a detection device, the demodulator produces 1 if \( a_0 > 0 \) and 0 otherwise. Table 2 exhausts all possible inputs and shows the relationship between the input and the output binary information. As can be seen, the relationship follows the majority principle.

A closer examination of BPSK shows that any pair of inputs that holds different binary information is offset when added at the EM level. As a result, the remaining input decides the binary outcome. In the case of a tie, since the detected energy fails to reach a threshold after the matched filter, a consistent conclusion will be made. Without loss of generality, we assume it to be binary 0.

2.2. Proposed Scheme for Collision Decoding

Using the majority principle, we design a coding scheme, which we call collision codes, that enables the common receiver of a collided transmission to tell the presence of individual transmission involved in the collision.

Consider that a multicast transmission is addressing to \( N \) receivers. In our design, we assume that the multicast sender embeds information in its data transmission to tell each individual receiver of its unique identifier. The identifier will be used for the acknowledgement transmission. It is possible that not all intended receivers detect the transmission due to, for example, noise or the hidden terminal problem. If an intended receiver detects the broadcast transmission it immediately replies an ACK with a predefined unique bitstream, \( s \), as part of ACK.

With the immediate replies of ACK transmissions from multiple receivers, a collision of ACK frames occurs. The sender decodes the superimposed transmission using a BPSK demodulator, and as discussed in the previous subsection the decoded bitstream, say \( v \), will follow the majority principle. Note that, to improve robustness, each receiver may use received signal strength indicator available in the IEEE 802.11 device to adjust its transmission power such that all ACK frames arrive at the multicast sender with a similar transmission power.

We now show that there exists a coding scheme such that the sender produces a unique bitstream for a particular combination of the receivers’ bitstreams. This unique bitstream enables the sender to identify whether a particular intended receiver has replied ACK indicating the success delivery of the multicast packet to that receiver.

In particular, consider the number of receivers, \( N \), to be an odd number, and a bitstream that has \( V \) bits. Let \( K = \frac{N+1}{2} \).
Table 3: The decoded bitstreams (based on the majority principle) of different receiver combinations for \( N = 3 \).

<table>
<thead>
<tr>
<th>Receiver Combination</th>
<th>Decoded Bitstream</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R_1 )</td>
<td>110</td>
</tr>
<tr>
<td>( R_2 )</td>
<td>101</td>
</tr>
<tr>
<td>( R_3 )</td>
<td>011</td>
</tr>
<tr>
<td>( (R_1,R_2) )</td>
<td>100</td>
</tr>
<tr>
<td>( (R_1,R_3) )</td>
<td>010</td>
</tr>
<tr>
<td>( (R_2,R_3) )</td>
<td>001</td>
</tr>
<tr>
<td>( (R_1,R_2,R_3) )</td>
<td>111</td>
</tr>
</tbody>
</table>

We first construct a binary matrix of size \( N \times V \) such that each column contains exactly \( K \) binary 1 and \( K-1 \) binary 0, with a unique permutation. Exhausting all permutations with \( K \) number of binary 1 and \( K-1 \) number of binary 0 produces \( \binom{N}{K} \) unique patterns, and then let \( V = \binom{N}{K} \). With this construction, the binary matrix holds \( N \) number of unique \( V \)-bit bitstreams, each of which will be assigned to a receiver. Such a construction guarantees a unique decoding output for any combination of the receivers’ bitstreams. The detailed proof of this is given in [2].

In the following, we give an example for \( N = 3 \). According to our scheme, the binary matrix for \( N = 3 \), \( A_3 \), can be constructed as,

\[
A_3 = \begin{pmatrix}
1 & 1 & 0 \\
1 & 0 & 1 \\
0 & 1 & 1
\end{pmatrix}.
\]

Each row in the binary matrix represents the unique bitstream for one receiver to transmit in its ACK. With assigning the three bitstreams to three receivers, \( R_1, R_2 \) and \( R_3 \), in order. Table 3 shows the decoded bitstreams at the sender for all possible combinations of ACK replies. It can be seen that a particular combination of receivers can always be uniquely identified by the decoded bitstream.

We would like to point out that the proposed collision codes do have some limitations. That is the length of the collision codes grows exponentially with respect to the number of receivers \( N \), which is impractical for larger \( N \). One way to resolve this is to divide stations into many small groups and let each group take turn for ACK transmission. In this way, the ACK collisions occurs only within each group, where the number of stations is small.

3. System Modelling

Similar to [12], we consider a single-hop wireless multicast network with \( N \) fixed receivers \( R_i, \ i \in \{1, \ldots, N\} \), with static membership, and an Access Point (AP) multicasting \( M \) data packets \( c_k, \ k \in \{1, \ldots, M\} \), to all the \( N \) receivers. Packet loss at each receiver is assumed to be independent, following the Bernoulli model with packet erasure probability at receiver \( R_i \) given as \( p_i, \ 0 \leq p_i < 1 \). For a packet batch, the AP builds a \( N \times M \) transmission matrix as illustrated in Table 4. Packet reception status of \( c_k \) at \( R_i \) is given by the binary bit \( r_{ik} \) in the transmission matrix, which equals ‘0’ for successful packet reception and ‘1’ for unsuccessful packet reception. Such a transmission matrix is updated after every transmission based on the packet feedback information from collision codes. The data payload of all the \( M \) packets are assumed to be equal, and is given as \( B \) bits for each packet \( c_k \).

The set of \( M \) native data packets is given by the vector \( \mathbf{M} = [c_1, \ldots, c_M] \), and the finite set of packets transmitted by the AP is given by the vector \( \mathbf{x} = [c_1, \ldots, c_M, x_1, \ldots, x_J] \), \( M \subseteq X \). The set of innovative packets which \( R_i \) has received is given as \( Y_i \), \( Y_i \subseteq X \).

We define a packet as an innovative packet for \( R_i \), if the packet is linearly independent with respect to the set of packets \( R_i \) already has. A receiver \( R_i \) is called a saturated receiver if it has received \( M \) innovative packets, otherwise it is called an unsaturated receiver. The utility of packet \( c_k \) is defined as the number of receivers which have not received \( c_k \).

3.1. Related Work

Network coding for wireless multicast transmission is a well-studied problem, and extensive literatures exist on this subject. Analytical results on throughput and latency performances of such transmission scheme are studied in [13, 14, 26, 15, 12, 36] and [17, 22, 26, 18] respectively. Experimental implementation of network coding on testbed have also been conducted [32, 24, 35, 34, 33, 27] to evaluate the practical implementation of computationally intensive packet encoding and decoding on battery and processor constrained devices such as laptop and smartphone.

Analytical results of network coding on the throughput performance has shown that for a fixed network, the throughput asymptotically increases with respect to the field size when the coding coefficient is selected randomly from the finite field [36, 15, 14], and a field size of \( GF(2^8) \) is sufficiently large enough to give such asymptotically optimal performance. Analytical results of delay show that an optimal coding scheme has the lowest decoding delay [23], and such decoding delay increases linearly with increasing packet batch size [18].

Concerns of large encoding and decoding throughput and computational complexities with increasing field size and packet batch size have been raised by Fitzek et al. research team through practical implementation of network coding on smartphones [33, 34, 36]. To reduce the high computation cost of coding over larger finite field size, the use of sparse coding coefficients have been proposed in [31, 30].

Given the low computational complexity and average decoding delay associated with XOR coding, researchers have shown...
particular interest in XOR based coding schemes. Several heuristic XOR coding algorithms have been proposed to minimize the number of transmissions [11, 12, 16, 24, 25, 29] and the average decoding delay [21, 23, 19, 20]. Since it is a NP-complete problem [16, 26] to minimize the number of transmissions using XOR coding, a small number of optimal solutions over larger finite field size have also been proposed [26, 30]. We will survey the various coding algorithms in Section 5.

### 3.2. Packet Encoding

In this paper we study the design and performance of a scalar-linear coding. Unlike the vector-linear codes, where a packet can be split in to smaller sub-packets before coding, in scalar-linear codes, the packets can not be split. A code $C$ of length $B$ bits is called binary linear code if $C$ is a subspace of the vector space $GF(2)^B$.

To code a packet, the AP chooses coefficients vectors from a finite field $GF(q)$, $q \in \mathbb{N}$, to form an encoding coefficients vector $G_j = [g_{j1}, g_{j2}, \ldots, g_{jM}]$, $g_{jk} \in GF(q)$, which is then multiplied with vector $M$ to generate a single coded packet $x_j$ given as,

$$x_j = G_j M^T.$$  

(3)

Coding over field size $q > 1$, requires multiplication and addition operations. Whereas for XOR coding, $g_{jk} \in \{0, 1\}$, only the addition operation is required.

For the receiver to be able to decode the coded packets, the AP needs to add information about the vector $G_j$ in the packet header. The number of bits required to store one such coefficient $g_{jk}$ is given as $q$ bits. Since there are $M$ such coefficients, the packet overhead for a coded packet is given as $Mq$ bits.

### 3.3. Packet Decoding

After receiver $R_i$ has received $M$ innovative packets, these $M$ packets are placed in matrix $Y_i$. The coding coefficients corresponding to each of the innovative packet is used to form a $M \times M$ coefficient matrix $H_i$. $H_i \in GF(2)^{M \times M}$. A native packet is represented by a 1-sparse row in $H_i$. The set of original packets $M$ can then be decoded by $R_i$ as,

$$M = H_i^{-1} Y_i^T.$$  

(4)

Inversion of $H_i$ is performed using Gaussian elimination. Coefficient matrix for XOR coded packets only require the “row interchange” and “row addition” operations [38] for Gaussian elimination. After the inversion of such matrix, only the addition operation needs to be performed.

However for packets coded over larger field size, Gaussian elimination would also need to perform “row scaling” operation (i.e. multiplying a row of the matrix with a non-zero scalar) in addition to the “row interchange” and “row addition” operations on $H_i$. After the inversion, the receiver will need to perform multiplication and addition to decode the native packets. Therefore even though the complexity order of matrix inversion for both $GF(2)$ matrix and $GF(q)$, $q > 1$, matrix is same, inverting a $GF(2)$ matrix requires fewer computation steps.

The use of Gaussian elimination to solve an arbitrary non-singular matrix has two major steps, triangularization with complexity $O(M^3)$, and back (forward) substitution with complexity $O(M^2)$ [38]. Triangularization converts an arbitrary matrix into an upper (lower) triangular matrix.

### 3.4. Problem Statement

The primary objective of the wireless multicast transmission problem which we study in this paper is to minimize the total number of transmissions, by using linear coding over $GF(2)$. The problem statement can be written as,

$$\text{minimize } |X|,$$  

(5)

subject to,

$$\text{rank}(H_i) = M, Y_i.$$  

(6)

$$g_k \in GF(2)$$  

(7)

The transmission minimization is subject to the constraint that all $N$ receivers should receive $M$ innovative packets, and that the packet coding is performed over $GF(2)$.

### 3.5. Complexity Class of the Problem

The complexity class of finding the minimum number of transmission problem using XOR coding has been studied in [16, 24, 26], where it has been shown that such a problem is a NP-complete problem, under the assumption that the receivers do not buffer innovative non-instantly decodable coded packets. An optimal solution for the minimum number of transmission problem using network coding can be found in polynomial time when the finite field size is bounded as $q \geq \lceil \log_2 N \rceil$ [26, 31].

### 4. Performance Bound

#### 4.1. Bandwidth Performance

Analytical results of the performance bound when the coding coefficient is selected randomly from a given finite field size is studied in [36, 15, 14]. In [12, 26] the upper bound on the retransmission bandwidth using network coding is given as a monotonic function of $p$, where $p$ is given as $p = \max(p_i)$.

**Proposition 1.** The minimum expected number of transmissions required to transmit $M$ innovative packets to $N$ memory-based receivers, using an optimal deterministic network coding algorithm such that $q \geq \lceil \log_2 N \rceil$, when packet reception is characterized by the binomial probability law, is given as,

$$G(p, k, M) = \sum_{m=0}^{\infty} \left( 1 - \sum_{i=M}^{m} \binom{m}{i} (1-p)^{m-i} p^{m-i} \right)^K,$$  

(8)

where $p = \max(p_i)$, and $K$ is the number of receivers with packet loss probability $p_i = p$.

**Proof.** The total number of transmissions will be dependent on the subset of receivers with the highest packet loss probabilities, $p$ [12, 26]. An optimal coding scheme requires that the coding field size is bounded as $q \geq \lceil \log_2 N \rceil$ [26, 31]. Under these assumptions, the proof of the derivation of Equation (8) is given in [13, 36].
Definition 2. Retransmission rate is the expected number of retransmissions for each native packet.

4.2. Latency Performance

In a network coding packet transmission scheme, even if the receiver receives an innovative coded packet, such a packet would be of no use until it can be decoded. For a receiver to be able to decode a coded packet it needs to wait for arbitrary time duration to receive more arbitrary innovative packets. Waiting to receive sufficient additional packets before being able to decode a coded packet incurs what we define as the decoding delay.

Definition 2. Decoding delay of a native packet $c_i$ at an unsaturated receiver $R_i$ is defined as the total number of time slots $R_i$ needs to wait to recover $c_i$ after its first transmission.

Decoding delay has also been defined in [20, 23], however in these works, the authors only assume that a receiver experiences a decoding delay of one unit if the successfully received packet is either non-innovative or if the packet can not be decoded instantly. Therefore such definition does not take into consideration decoding delay when the AP transmits a packet, and the receiver does not receive it. Whereas in our model, we count all the transmissions made by the AP towards the decoding delay, before $R_i$ decodes the native packet.

We compare the decoding delay of our proposed coding scheme with systematic random linear network coding (RLNC) over large field size. In RLNC, the coding coefficients are randomly selected. In a systematic RLNC transmission scheme, the AP first transmits $M$ native packets, and then transmits coded packets until all the receivers have received $M$ innovative packets. RLNC transmission over large field size gives asymptotically optimal throughput performance.

Proposition 2. The average decoding delay for $R_i$ to recover a lost packet using a systematic RLNC scheme over a large finite field size for a network with homogenous packet loss probability (i.e. $p_i = p, \forall i$) is given as,

$$D(p, M) = g(p, 1, M) - \left(\frac{M + 1}{2}\right).$$

Proof. In a systematic RLNC, the total time slots before a receiver $R_i$ is able to recover a lost packet is given as the sum of the number of transmissions $R_i$ has to wait in the native packet transmission phase and the number of transmissions during the coded packet transmission phase. For RLNC $R_i$ can only perform packet decoding after it has received $M$ innovative packets.

We first calculate the average time a receiver has to wait during the native packet transmission phase. If packet $c_k$ is lost, then the receiver needs to wait for $M - k$ transmissions in the native packet transmission phase. Packet $c_1$ needs to wait for 0 time slot, packet $c_{M-1}$ needs to wait for 1 time slot, packet $c_{M-2}$ needs to wait for 2 time slots and so on. The summation of the delay in the first phase can be simplified as the sum of first $M - 1$ natural numbers. This is given as $\frac{M(M-1)}{2}$. Therefore the average time slots a receiver needs to wait for each packet during the native packet transmission phase is given as $\frac{M^2 - 1}{2}$.

We now calculate the time slots a receiver needs to wait during the coded packet transmission phase. Each receiver performs packet decoding only after it has received $M$ innovative packets. Number of transmissions necessary before each of the receiver has $M$ innovative packets is independently and identically distributed (iid), with an expected value given as $g(p, 1, M)$. The expected number of transmissions during the coded packet transmission for each receiver is given as $g(p, 1, M) - M$. Adding the results of native packet transmissions and coded packet transmissions the equation reduces to Equation (9) in a simplified form. □

4.3. Coding and Decoding Computational Complexities

The two fundamental mathematical operations required to generate a network coded packet are multiplication and addition. Let $a$ and $b$ be two non-negative integers, then the computational complexities of multiplication\(^2\) and addition of $a$ and $b$ are given as $O([\log_2 a] \cdot [\log_2 b])$ and $O([\log_2 a] + [\log_2 b])$ respectively [37]. The complexity of the operation $g_k \cdot c_i$ (See Equation (3)) is given as $O(qB)$. Since $M$ such operations are carried out to generate a coded packet, the computation cost of multiplication to generate a coded packet is $O(MqB)$, and the computation cost of addition is given as $O(MB)$. The complexity order of generating $M$ coded packets is given as $O(M^2qB + M^3)$.

Decoding a set of $M$ packets (See Equation (4)) first requires the inversion of matrix $H_i$ using Gaussian elimination with complexity given as $O(M^3)$ [38]. Once the inversion is performed, decoding one packet is performed by multiplying the coefficients of the inverted matrix $H_i^{-1}$ with $M$ packets from $Y_i$, which has total complexity given as $O(MqB)$. Decoding $M$ coded packets therefore has total complexity given as $O(M^2qB + M^3)$.

Exception to this decoding complexity applies for decoding instantly decodable XOR coded packet. For decoding such XOR coded packets multiplication operations are not required. We consider a worst case packets transmission scenario to compute the computational complexities of encoding and decoding XOR packets. Consider an arbitrary packet reception state at $R_i$ such that $H_i$ is represented by an upper triangular $GF(2)$ matrix as follow,

$$
\begin{pmatrix}
1 & 0 & 0 & \cdots & 0 \\
1 & 0 & 0 & \cdots & 0 \\
1 & 1 & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & 1 & 1 & \cdots & 1
\end{pmatrix} =
\begin{pmatrix}
c_1 \\
c_1 \oplus c_2 \\
c_1 \oplus c_2 \oplus c_3 \\
\vdots \\
c_1 \oplus c_2 \oplus \cdots \oplus c_M
\end{pmatrix}.
$$

Solving such a matrix would require the use of back-substitution method with complexity given as $O(M^2)$ [38]. Since the length of each packet is $B$ bits, the total worst case encoding and decoding complexity for XOR packets is given as $O(M^2B)$.

\(^2\)A survey and discussion on efficient multiplication algorithms is beyond the scope of the paper.
Experimental evaluation of network coding has also been carried out to study the effect of energy cost of encoding and decoding on devices such as smartphones. Experimental evaluation of RLNC over $GF(2^8)$ on iPhone 3G, for $M = 64$ with packet length of 4096 bytes, has shown that for two devices with same configurations and running the same applications, the device running with an additional RLNC decoding application consumes approximately 20% more battery energy reserves [32]. However it only cost 191 nJ of energy to code two packets each 1000 bytes long over $GF(2)$ [33]. Transmission of a 1000 bytes long packet over IEEE 802.11 network on Nokia N95 consumes 2.31 mJ of energy [33]. This implies that the energy cost of XOR coding packets has no apparent effect on the total energy cost of coding and transmitting a XOR packet.

Encoding over $GF(2)$ is also approximately 8 times faster than encoding over $GF(2^8)$ on iPhone 3G implementation, while decoding over $GF(2)$ is approximately 6 times faster than decoding over $GF(2^8)$ on the same platform [34].

5. Packet Coding Algorithms

We survey and classify the coding algorithms proposed in literature for the wireless multicast transmission problem in this section. A summary of various packet coding algorithms is systematically presented in Figure 2.

5.1. Coding over Larger Field Size

When coding over larger field size $GF(2^q)$, $q > 1$, the coding coefficient can be either randomly selected, or deterministically selected by an algorithm. When the coding coefficients are uniformly and randomly selected, such a coding scheme is known as random linear network coding (RLNC) [15]. Throughput performance of RLNC over large field size is asymptotically optimal.

The dynamic general-coding based scheme (DGC) [26] and sparse linear network coding (SLNC) [31, 30] are deterministic coding algorithm over field size $GF(2^q)$, $q \geq \lceil \log_2 N \rceil$. The throughput performance of both DGC and SLNC are optimal. SLNC coding vector are sparser compared to the DGC coding vector. This has an advantage of smaller decoding complexity of $O(M\omega)$ using Wiedemann algorithm, where $\omega$ is the maximum number of non-zero components in any given row of $H$, [30].

5.2. Coding over $GF(2)$

Packet coding algorithms over $GF(2)$ have been traditionally designed such that if the the coded packet is innovative for any receiver, then the receiver must also be able to instantly decode the packet, otherwise the packet is discarded. This constraint is known as strictly instantly decodable network coding (SIDNC) condition [20]. When the coding algorithm is designed such that an innovative packet need not necessarily be instantly decodable by a receiver, then such coding algorithms are known as generalized instantly decodable network coding (GIDNC) condition [20]. In GIDNC if an innovative packet can not be decoded immediately, then such a packet is discarded.

In another class of XOR coding algorithm, if an innovative packet can not be instantly decoded then the receiver cache this packet in its memory buffer. We call such algorithms as memory-based XOR coding algorithms. The cache-based multicast retransmission encoding (CMRE) [28] and our proposed coding algorithm are examples of this class of algorithm.

The adaptive network coding (ANC) [19], opportunistic coding [27], sort-by-utility [24], Lu et al. proposed algorithm [25] and heuristic vertex-coloring algorithm [29] are special cases of SIDNC. Opportunistic coding uses greedy algorithm to make coding decision. In sort-by-utility and ANC, the coding decision is made based on the packets’ utilities. When making coding decision sort-by-utility and ANC gives priority to those packets with maximum utilities under the SIDNC constraint.

The maximum weight vertex search (MWVS) [20] and the systematic online network coding (SNC) [21] are heuristic coding algorithms under the GIDNC constraint. The problem of minimizing the decoding delay is formulated as a maximum clique search algorithm in [20], for which the authors propose the MWVS heuristic algorithm.

5.3. Minimum Clique Partition Problem

Coding algorithm under SIDNC constraint can be reduced to the minimum clique partition problem on graph $G = (V,E)$ [20, 24]. The set of vertices is given by $V_k \subseteq V$, $\forall i$ and $\forall k$, giving $|V| = N \times M$. There exists an edge between two vertices if coding packets corresponding to these two vertices can be decoded instantly by the receivers corresponding to these two vertices.

The minimum number of transmissions under SIDNC can be solved by solving the minimum clique partition problem on graph $G$. The minimum clique partition problem on graph $G$ is equivalent to the minimum vertex coloring problem on the complementary graph $\overline{G}$ [40]. Therefore the MWVS heuristic for the minimum clique partition problem in [20] and the largest-degree first (LDF) heuristic for the minimum vertex coloring algorithm.
problem in [29] use a similar problem formulation to minimize the number of transmissions.

In the work by Rozner et al. [24], the authors have shown the performance gain of sort-by-utility algorithm over minimum clique partition greedy algorithm, where the coding decision is made based on the degree of vertices. We therefore expect sort-by-utility to also outperform MWVS, as MWVS for homogeneous packet loss model also makes coding decision based on the degree of vertices.

5.4. Algorithm Comparison

The throughput performance of DGC and SLNC is the same as the optimal performance given in Equation (8). Similarly the upper bound for the decoding delay of DGC and SLNC is given by the average decoding delay of RLNC, which is given in Equation (9).

So far, the sort-by-utility coding algorithm (which is similar to ANC) has been shown to be the best throughput performing XOR coding algorithm under the SIDNC constraint. Simulation we carried out for CMRE under the transmission model given in Section 3 showed that sort-by-utility performs significantly better than CMRE. Similarly for the Lu et al. algorithm [25], one can easily verify from the simulation results shown in that paper\(^3\) that the performance of the Lu et al. algorithm does not show significant gain. We therefore exclude the performance of CMRE and Lu et al. algorithm [25] from our simulation in Section 7 without loss of ambiguity. Similarly we also exclude the performance of opportunistic coding from our results, as it uses a simple greedy algorithm to make coding decision.

The MWVS is the current claimant of having the lowest average decoding delay of all the minimum clique heuristic algorithms, and has been shown to outperform ANC. The SNC is a simple online greedy algorithm, where the AP transmits the coded packet using greedy search algorithm under GIDNC constraint. SNC achieves improvement in decoding delay as it transmit a coded packet once the coding conditions are satisfied rather than wait until the end of \(M\) native packets transmission before sending coded packets.

6. BENEFIT Online Coding Algorithm

We now propose our XOR coding algorithm - BENEFIT. Unlike traditional coding transmission schemes where \(M\) native packet are transmitted before the AP starts transmitting the coded packets, for our proposed algorithm the AP may transmit a coded packet in the midst of native packet transmissions if the algorithm coding conditions are satisfied. Therefore after transmitting a proper subset of packets from the set \(M\), if a subset of these transmitted packets satisfy the algorithm coding conditions (Table 6 and 7), then these packets are coded and transmitted. The fundamental essence of our coding algorithm can be summarized as follow:

A XOR coded packet is transmitted if it is an innovative packet for the maximum possible number of unsaturated receivers, while at the same time also being ‘instantly decodable’ by most of these receivers.

By instantly decodable we mean that the receiver can decode the XOR coded packet using the set of packets it already has in its buffer. If the coded packet is not instantly decoded by the receiver and it is an innovative packet, then the receiver caches this packet in its memory buffer. By transmitting a coded packet after the transmission of a proper subset of packets from \(M\) rather than waiting until the end of \(M\) packet transmissions reduces the average decoding delay, and makes our transmission scheme an online coding algorithm.

The higher retransmission bandwidth of our coding algorithm with respect to previously proposed XOR coding algorithms originates due to the fact that we deviate from the traditional SIDNC packet coding rule [27]:

For \(R_k\) to transmit \(M\) packets \(c_1, ..., c_M\) to \(M\) receivers, \(R_1, ..., R_M\) respectively, the coded packet obtained by coding \(M\) packets \(c_1, ..., c_M\) can only be decoded by \(R_i\), if \(R_i\) has \(M - 1\) of \(c_j\) packets, except \(c_i\) (\(j \neq i\)).

For our proposed coding algorithm, BENEFIT makes coding decision primarily based on the number of receivers for which the coded packet will be innovative. In addition to keeping the coded packet innovative for maximum possible number of receiver, the algorithm also gives weightage to those coded packets, which can be instantly decoded by most of the receivers. Lastly, when the coded packet is not instantly decodable by a receiver, the algorithm keeps the coding vector sparse so that the buffered coded packet is decoded at the earliest, as the receiver receives more instantly decodable packets, and also to keep the decoding complexity low.

We illustrate the characteristics of our proposed algorithm with the aid of a simple example. Consider the transmission matrix in Table 4. For the purpose of illustration and without loss of generality, for this example lets assume that the coded packets are transmitted on noiseless channels. After the AP has transmitted \(c_1\) and \(c_2\), our proposed algorithm code these two packets and transmits the coded packet. This way the receivers do not need to wait until the end of native packet transmissions before being able to recover packets \(c_1\) and \(c_2\). Packet \(c_1 \oplus c_2\) will be innovative for all the receivers, and immediately decodable by all the receivers except \(R_1\), in which case \(R_1\) cache \(c_1 \oplus c_2\) in its memory buffer. The AP then transmits packets \(c_3\) and \(c_4\) in the queue. The algorithm then code these two packets along with \(c_2\). The coded packet \(c_2 \oplus c_3 \oplus c_4\) will be innovative and instantly decodable by all the receivers. Once \(R_1\) decodes \(c_2\), it can then use \(c_2\) to decode \(c_1\) from \(c_1 \oplus c_2\) stored in its memory buffer.

6.1. Algorithm Pseudocode

The pseudocode of our algorithm is given in Table 5. The purpose of the ARQBenefit() and DecodeBenefit() coding conditions given in Table 6 is to select coding coefficients such that

\(^3\)In [25] simulation results have been shown for small networks, \(N \leq 9,\) with low loss probability of \(p \leq 0.2,\) and large packet batch size. For such parameters any efficient heuristic coding algorithm will give near-optimal throughput performance. Through simulation - results not shown in this paper - we have verified that our proposed algorithm outperforms the Lu et al. algorithm for these parameters.
The following 'while' loop searches for the coding coefficients given the current value of `expected_benefit`.

```plaintext
while continue_scan==true
    continue_scan = false;
    initialize \( G_{j} \) such that \( g_{k} = 0, \forall k \);

    for \( \forall c_{k} \)
        \( g_{k} = 1 \);
        if ARQBenefit() and DecodeBenefit() are false
            \( g_{k} = 0 \);
        else if InnovativePacket() is true
            Broadcast the coded packet;
            Superimposed acknowledgement;
            Update the transmission matrix;
            \( coefficients\_found = true; \) // All coefficients found.
        else
            // A coding coefficient has been found;
            // Continue searching for more coefficients;
            continue_scan = true;

    // If coding coefficients can not be found, and there is at least
    // one unsaturated receiver then relax the coding condition.
    if coefficients\_found==false
        decrement(expected_benefit);
```

The purpose of InnovativePacket() condition (Table 7) is to select the set of native packets for coding such that the number of retransmissions will be minimized. This is achieved by ensuring that the encoded packet is innovative for most of the receivers. If no such set of native packets can be found which satisfy these conditions, and not all the receivers have received \( M \) innovative packets, then the algorithm relaxes the coding condition. This reduces the expected number of receivers for which the coded packet should be innovative.

**Proposition 3.** For a fixed network of size \( N \), the maximum number of native packets used to generate a coded packet by the \textsc{Benefit} coding algorithm, for any arbitrary packet batch size \( M \) is bounded by \( N \).

**Proof.** The DecodeBenefit() coding condition requires that every native packet used for coding should be instantly decodable by at least one unsaturated receiver. The maximum number of receivers which can instantly decode a coded packet is \( N \) receivers. If each of the packet a receiver can decode is assumed to be unique, then the maximum number of packets used for coding is bounded as \( N \) native packets.

InnovativePacket() is implemented as follow. The AP maintains \( N \) such \( H_{i} \) matrices to store the coding vectors of the innovative packets received by the receivers. All the \( N \) matrices are arranged in echelon form. To evaluate whether a given coded packet would increase the rank of matrix \( H_{i} \) by one, the AP only need to performs the “row interchange” and “row addition” operations.

We now discuss the sparsity of such matrices \( H_{i} \). All packets transmitted during native packet transmissions are 1-sparse. During the coded packet transmissions, ARQBenefit() and DecodeBenefit() coding conditions select coded packets which are
instantly decodable by most of the receivers, and hence results in a 1-sparse rows in the matrix after being instantly decodable by the corresponding receivers. Those coded packets which can not be instantly decodable, based on Proposition 3, will result in a maximum of \( N \)-sparse row in the matrix \( \mathbf{H} \). This therefore makes \( \mathbf{H} \) a sparse matrix.

As we have shown that matrix \( \mathbf{H} \) will be sparse, InnovativePacket() for such a sparse matrix can alternatively also be implemented using the Wiedemann algorithm [41].

6.2. BENEFIT Packet Decoding

While most of the coded packets generated by the BENEFIT coding algorithm can be decoded using back-substitution, it is possible that the packet decoding may require the triangularization step on matrix \( \mathbf{H} \) to decode the native packets by the receivers. If decoding \( \mathbf{H} \), requires the triangularization step, then as shown earlier, due to the sparsity of such matrix, \( \mathbf{H} \) can be inverted using the Wiedemann algorithm with computational complexity\(^4\) bounded as \( O(M^2 \log_2 M) \) [41].

If matrix \( \mathbf{H} \) can be arranged as a triangular matrix, then the best case packet decoding complexity is bounded as \( O(M^2) \) using back-substitution, as discussed in Section 4.3. Whereas the worst case packet decoding complexity when triangularization step is required, is bounded as \( O(M^2 \log_2 M) \) using the Wiedemann algorithm.

6.3. BENEFIT Algorithm Complexity

The worst case computational complexity of ARQBenefit() and DecodeBenefit() is given as \( O(NM) \). The computational complexity of the most computationally intensive step of BENEFIT, InnovativePacket() implemented using Wiedemann algorithm assuming that the triangularization step is required, is given as \( O(NM^2 \log_2 M) \) for generating \( M \) coded packet. Therefore BENEFIT algorithm computational complexity is given as \( O(NM \log_2 M) \) for the worst-case scenario.

7. Performance Evaluation

7.1. Superimposed Acknowledgement

In this section, we show the advantage of our proposed collision codes for collecting ACKs in wireless multicast scenario. We consider the popular IEEE 802.11b standard [3] and summarize the values of protocol parameters needed for our performance evaluation in Table 8. The full data rate of 11 Mbps for data transmission is assumed. We consider data payload of 8184 bits with 272 bits of MAC header. The 802.11b long preamble physical layer convergence procedure protocol data unit (PLCP PDU) is transmitted at 1 Mbps, and has a constant transfer time of 192 \( \mu \)s.

While the IEEE 802.11 standard does not enforce explicit acknowledgement for a one-to-many transmission, for the performance comparison, we consider the use of the standardized but optional point coordination function (PCF) to achieve acknowledgement. PCF uses power-save poll (PS-Poll) frame to perform individual polling. Its protocol handshake procedure is shown in Figure 3(a), and the protocol handshake procedure for our proposed collision codes method is shown in Figure 3(b). The length of the PS-Poll and ACK frame for 802.11b is 160 bits and 112 bits respectively, both transmitted at 1 Mbps.

Based on Figure 3 and Table 8, we have,

\[
T_d = T_{PHY} + \frac{722+8184}{11}
\]

\[
T_p = SIFS + \delta + T_{POLL} + SIFS + \delta + T_{ACK}
\]

\[
T_a = SIFS + \delta + T_{CCF}
\]

\[
T_{CCF} = T_d + \frac{NT_a + DIFS + \delta}{N}
\]

where \( T_d, T_p, \) and \( T_a \) (all given in \( \mu s \)) are the duration of a data packet transmission at 11 Mbps, the duration to poll a receiver, and the duration for all receivers to simultaneously return an acknowledgement, respectively. With these timings, considering multicasting to \( N \) receivers, we can determine the duration of the protocol handshake procedure by,

\[
T_{PCF} = T_d + NT_a + DIFS + \delta
\]

\[
T_{CCF} = T_d + \left[ \frac{N}{2} \right] T_a + DIFS + \delta
\]

\(^4\)An efficient implementation and computational complexity evaluation of triangularization on \( \mathbf{H} \) is given as follow. Form a smaller matrix \( \mathbf{J} \) derived from \( \mathbf{H} \) as follow. The 1-sparse rows from \( \mathbf{H} \), can be eliminated after back-substitution of these 1-sparse rows on those rows which are not 1-sparse, such that there are no more 1-sparse rows in \( \mathbf{H} \). After such back-substitution on matrix \( \mathbf{H} \), the resulting 1-sparse columns can also be eliminated from \( \mathbf{H} \), resulting in a \( \ell \times \ell \), \( \ell \leq \kappa \leq M \), matrix \( \mathbf{J} \), where \( \kappa \) is the number of rows in \( \mathbf{H} \) which are not 1-sparse. Due to the sparsity of \( \mathbf{J} \) based on Proposition 3, the computation cost of triangularization on \( \mathbf{J} \), using Wiedemann algorithm would be given as \( O(C \log_2 \ell) \) [41], which for \( \ell \leq M^{0.7} \) is bounded as \( O(M^2) \), based on \( M^{0.7} \log_2 M \leq M^2 \), for \( M \) at least in order of 1,000’s.
where $T_{PCF}$ and $T_{CC}$ are the durations using the standard PCF mechanism and our proposed collision code mechanism respectively. For our proposed mechanism, when the number of receivers exceeds 9, we partition them into several groups for acknowledgement collisions.

Figure 4 plots and compares the two protocol handshake durations. As can be seen, our mechanism using collision codes requires significantly shorter period to complete the acknowledgement compared to IEEE 802.11b PCF.

7.2. Retransmission Rate Performance

Through simulation results, we found that the retransmission rate of CMRE, Lu et al. algorithm [25] and MWVS is significantly worse than sort-by-utility, we therefore exclude the simulation results of these coding scheme from Figure 5 without loss of ambiguity (See discussions in Sections 5.3 and 5.4).

For our simulation in Figures 5 and 6 we assume homogeneous packet loss model, i.e. $p_i = p$, $\forall i$. Such assumption is practical, because as discussed in Proposition 1, for any heterogeneous packet loss model network, the problem can be reduced to homogeneous packet loss model for those subset of receivers with highest packet loss probabilities.

The graph results in Figure 5(a)-5(c) purports that our proposed BENEFIT coding algorithm is not only the best performing XOR coding algorithm but also significantly outperforms previously known XOR coding schemes. The reason for the small difference between the performance of BENEFIT and the lower bound is explained due to the much smaller coding field size over which BENEFIT coding scheme performs packet coding. The effect of coding field size on throughput performance has been studied in [16, 31, 15], where it has been shown that the retransmission rate of a coding scheme for a given network is dependent on the coding field size over which coding is performed, and an optimal solution for network can only be found when the coding field size is bounded as $q \geq \lceil \log_2 N \rceil$. 

Figure 5: Retransmission bandwidth performance. Lower bound and RLNC results overlap.
The results in Figure 5(a) show the retransmission rate with respect to the network size. The graph also shows that the retransmission rate of RLNC over a relatively large coding field size is sufficient to achieve retransmission rate close to the optimal performance. This can be seen from the near-overlap of the RLNC and lower bound plots in the graph results. The retransmission rate increases with the number of receivers $N$, however such increase is logarithmic in nature, and is not bounded by any asymptote. Figure 5(b) shows the retransmission rate with respect to packet loss probability. The retransmission rate increases exponentially with respect to $p$. Both the results in Figure 5(a) and 5(b) shows that even for a large network with high packet loss probability, our proposed coding scheme delivers a near-optimal retransmission rate.

The retransmission rate in Figure 5(c) show logarithmic decrease with increasing packet batch size. Unlike the logarithmic graph result of Figure 5(a), the retransmission rate of Figure 5(c) is bounded by an asymptote [26, Theorem 4].

Results from Figure 5(c) demonstrate an interesting observation about the retransmission rate of a multicast transmission. The graph shows that as the packet batch size increases, the retransmission rate decreases. Similar results had also been reported in [12, 24]. However in both these works, the authors’ main reservation for using larger packet batch size is the higher packet decoding delay penalty tradeoff. As we will show in the next subsection, our proposed coding scheme has the lowest decoding delay penalty of any known network coding scheme. This therefore paves way for using a larger packet batch size. By doing so, for a given average decoding delay tolerance, our proposed coding scheme can effectively achieve lower retransmission rate by using larger packet batch size compared to other coding schemes which will be bounded to use smaller packet batch size to achieve the same average decoding delay tolerance.

7.3. Decoding Delay Performance

According to the authors of DGC, because of the difficulty to have an exact delay analysis of DGC, only an upper bound for the delay analysis of DGC has been presented in the paper, whereby the authors assume the worst-case scenario that the receiver is able to decode the coded packet after receiving $M$ innovative packets. Therefore the upper bound for delay analysis of DGC, like SLNC, equals the average decoding delay of RLNC. Figure 6 shows the latency performance of various coding schemes. Results in Figures 6(a)-6(c) show that our proposed coding scheme has the lowest average packet loss recovery delay of any known network coding retransmission scheme. The shape of these graphs is explained based on the corresponding graphs in Figure 5. For RLNC, the theoretical delay and simulation delay matches very well, and demonstrate the correctness of Proposition 2.

In Figure 6(a), the average decoding delay of BENEFIT increases in the same order as the retransmission rate increases in Figure 5(a) for BENEFIT. For smaller network size, sort-by-utility has lower decoding delay than RLNC, because every coded packet sort-by-utility transmit is instantly decodable. With increasing network size, sort-by-utility retransmission rate
increases, the increasing number of retransmissions per packet results in increasing decoding delay. Therefore for larger networks, sort-by-utility decoding delay performance gets worse than RLNC.

Figure 6(b) shows that the decoding delay increases exponentially with respect to packet loss probability, which is consistent with the graph results of Figure 5(b). Figure 6(c) shows that the decoding delay increases linearly with packet batch size. The shape of the graph is explained as follows. RLNC starts transmitting coded packet after $M$ native packets transmission, and the receiver need to have $M$ innovative packets before it can decode the set of coded packets it has received. Similarly sort-by-utility also starts transmitting coded packets after $M$ native packet transmissions. This explains the increase in average decoding delay with increasing packet batch size.

### 7.4. Computational Complexities

The computational complexities of various coding schemes is summarized in Table 9. We evaluate the computational complexities for generating $M$ coding vector, and similarly coding and decoding $M$ coded packets. The algorithm complexity is the complexity of finding coding vector. The encoding complexity is the complexity of encoding the packets which involves the multiplication and addition operations. And the decoding complexity is the complexity of decoding coded packets to retrieve the native packets. Derivation of encoding and decoding complexities is given in Section 4.3.

The complexity of running a random number generator is constant and given as $G$. Therefore the total algorithm complexity for RLNC is given as $O(M^2G)$. Sort-by-utility runs a sorting algorithm, the complexity of an efficient sorting algorithm is given as $O(M \log_2 M)$. For the worst case scenario of sorting $M$ times, sort-by-utility has worst case complexity of $O(M^2 \log_2 M)$. The total complexity of MWVS to generate $M$ coded packets is given as $O(N M^3)$.

For SLNC, the algorithm complexity of finding the sparest coding vector is given as $O(M^3 N^2)$ [30]. For SLNC encoding process we consider the multiplication cost of $c_k \cdot c_i$ only when $g_k \neq 0$. Since the coding coefficient for each coded packet has $\omega$ non-zero components, the encoding complexity of SLNC is therefore given as $O(M \omega q B)$, while inverting such $\omega$-sparse coding vector based matrix has complexity given as $O(M^2 \omega)$ [30] using the Wiedemann algorithm.$^5$

The algorithm complexity of BENEFIT is lower than SLNC, DGC and MWVS, and comparable to sort-by-utility and RLNC for a fixed network size. BENEFIT enjoys the lower encoding complexity of XOR packet encoding.

The worst case decoding complexity of BENEFIT is lower than RLNC, DGC and SLNC. While the best case decoding complexity of BENEFIT is equal to decoding complexities of SIDNC based algorithms, as discussed in Section 6.2.

### 7.5. Packet Overhead

The packet overhead of a network coded packet is derived in Section 3.2. A summary of packet overhead for various coding schemes is given in Table 9. The results highlight that BENEFIT, like other coding algorithm over $GF(2)$, has the lowest packet overhead.

### 8. Conclusion

In this paper we have demonstrated a cross-layer solution to improve the reliability of IEEE 802.11 wireless multicast transmission. The contributions of the paper are twofold. First we have shown collision codes, a practical scheme to collect ACK frames from multiple receivers simultaneously. Secondly we have shown an efficient online $GF(2)$ coding scheme to retransmit the lost packets.

Through simulation results we have shown that BENEFIT has the lowest retransmission rate of any $GF(2)$ linear coding algorithms, and the lowest average decoding delay of any linear coding scheme irrespective of the field size. The lower average decoding delay of our algorithm makes our coding scheme attractive for delay-sensitive applications such as multicasting a live video stream to a group of receivers. Whereas by using sparse $GF(2)$ coding vector, we take advantage of the lower encoding and decoding complexities, which makes BENEFIT a coding scheme of interest in battery and processor constrained devices such as smartphones.

The contribution of our algorithm is that we allow the receivers to buffer those packets which may not be immediately decodable but are innovative. By keeping the coding vector sparse, the decoding complexity of coded packets generated by BENEFIT does not suffer from the high computational complexity of $O(M^3)$ using the Gaussian elimination, as it can be solved using the Wiedemann algorithm with complexity $O(M^2 \log_2 M)$, for the worst case scenario. Similarly by transmitting a coded packet online, as soon as the coding conditions are satisfied, rather than wait until the end of native packets transmissions before starting the coded packet transmissions, keeps the decoding delay of our BENEFIT coding algorithm low.

### 9. Acknowledgement

We would like to appreciate and acknowledge the constructive feedbacks provided by the anonymous reviewers during the revision of this work. Such feedback has helped us to significantly improve the presentation and technical correctness of this paper.

### References

Table 9: Summary of the characteristics of various coding schemes.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Complexity</th>
<th>Packet Overhead (bits)</th>
<th>Encoding Complexity</th>
<th>Decoding Complexity</th>
<th>Decoding Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>RLNC</td>
<td>$O(M^2)\log(L)$</td>
<td>$M_q$</td>
<td>$O(M^2qB)$</td>
<td>$O(M^2qB + M^3)$</td>
<td></td>
</tr>
<tr>
<td>DGC</td>
<td>$O(N^2M^4)$</td>
<td>$M_q$</td>
<td>$O(M^2qB)$</td>
<td>$O(M^2qB + M^3\omega)$</td>
<td></td>
</tr>
<tr>
<td>SLNC</td>
<td>$O(M^2\log(L))$</td>
<td>$M_q$</td>
<td>$O(M^2\log(B))$</td>
<td>$O(M^2B)$</td>
<td></td>
</tr>
<tr>
<td>MWVS</td>
<td>$O(N^2\log(L))$</td>
<td>$M$</td>
<td>$O(M^2B)$</td>
<td>$O(M^2B)$</td>
<td></td>
</tr>
<tr>
<td>Sort-by-utility</td>
<td>$O(N^2\log(M))$</td>
<td>$M$</td>
<td>$O(M^2B)$</td>
<td>$O(M^2B\log(M))$</td>
<td></td>
</tr>
<tr>
<td>BENEFIT</td>
<td>$O(N^2\log(M))$</td>
<td>$M$</td>
<td>$O(M^2B)$</td>
<td>$O(M^2B\log(M))$</td>
<td></td>
</tr>
</tbody>
</table>

Remark: $q \geq 8$, $q \geq [\log_2(N)]$, $q \geq [\log_2(N)]$, - , -


