XOR-based Coding for the 3-user Broadcast Erasure Channel with Feedback

Sophia Athanasiadou, Marios Gatzianas, Leonidas Georgiadis, Leandros Tassiulas

XOR-based Coding for the 3-user Broadcast Erasure Channel with Feedback

Sophia Athanasiadou*, Marios Gatzianas†, Leonidas Georgiadis*†§ and Leandros Tassiulas‡†§
* Dept. of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Thessaloniki, 54 124, Greece.
† Centre for Research and Technology Hellas, Thermi, Thessaloniki, 60 361, Greece.
‡ Computer Engineering and Telecommunications Dept., Volos 38 221, Greece.
Emails: sathanasiadou@ee.auth.gr, mgatzia@ee.auth.gr, leonid@auth.gr, leandros@uth.gr

***Abstract*—We study the case of the three-user broadcast erasure channel, with multiple unicast traffic, where feedback from the users is fed back to the transmitter in the form of ACK/NACK messages. The capacity region of this system has been recently derived and two capacity achieving algorithms, employing inter-session linear network coding, have been proposed in [1], [2]. Since these algorithms suffer from large computational complexity and decoding delay, our aim in this paper is to design coding algorithms with reduced computational complexity and low decoding delay that achieve comparable throughput to the former algorithms. We exclusively consider algorithms that require no knowledge of channel statistics, perform only XOR operations between the packets and allow for instantaneous decoding by any receiver that successfully receives a packet. We present two algorithms with the above properties: the first one, named `XOR1`, operates on a specially constructed network of virtual queues and is seen via simulations to perform adequately with respect to a tight capacity outer bound. The second algorithm, named `XOR2`, is an enhanced version of `XOR1` that operates on the same virtual network and achieves higher throughput through more intelligent packet combining. Furthermore, we show that `XOR2` achieves capacity under a general condition, which is satisfied in the following settings: 1) spatially iid erasure channels with arbitrary values of erasure probability, and 2) spatially independent erasure channels where the maximum erasure probability does not exceed 8/9.**

I. Introduction

The broadcast erasure channel (BEC) is commonly used as an abstraction for wireless networks since it captures the two main characteristics of the wireless medium: 1) its inherently broadcast nature, and 2) its potential for packet losses (due to interference, fading, congestion, packet collision etc), which can be perceived as "erasures" by higher layers. For this reason, the channel has been studied under various assumptions and traffic scenarios (mostly, of multicast and broadcast type) and, although its capacity region remains unknown in the general case, several important special cases have been successfully solved. Specifically, it has been established that the BEC network capacity region for multicast traffic is achievable through linear coding at intermediate nodes [3], while a similar conclusion has recently been reached, concurrently in [1], [2], for the case of single-hop networks with multiple unicast traffic (which is traditionally the "difficult" regime of network coding).

The throughput-optimal coding schemes in [1], [2] exploit causal feedback to transmit specially constructed linear combinations of packets. As a result, these schemes suffer from high complexity and decoding delay since they rely on operations in a sufficiently large sized finite field (which is required by the users so that they receive innovative information with high probability) and, additionally, each destination must first receive a sufficient number of packets before it can decode a single packet. This delay may be unacceptable in certain applications such as live video streaming. Hence, there exists a definite need for an efficient algorithm that achieves the capacity of the BEC channel with multiple unicast traffic and uses simple packet operations (such as binary XORs) that lead to low decoding delay. This forms the topic of the current paper, which can be regarded as the "low complexity/delay" alternative to the algorithms in [2]. Although XOR-based opportunistic network coding schemes for unicast traffic have also been presented in [4] (as a sublayer between the IP and MAC layer of 802.11) and [5] and evaluated for realistic non-erasure networks, no explicit performance guarantees are provided. Additionally, a low complexity/delay XOR-based algorithm (similar in concept to [5]) has been presented in [6] for the 2-user unicast BEC; however, an extension of this work to $N > 2$ is not at all straightforward.

The contribution of this paper can be summarized as follows:

- we propose two online algorithms, named `XOR1` and `XOR2`, for the 3 user BEC channel with multiple unicast traffic that are specially geared towards low (computational) complexity and delay. The former is achieved by using XOR operations only, while the latter follows from the algorithm's structure and the special virtual queues that are constructed. In fact, both algorithms allow for instantaneous packet decoding upon successful reception (i.e. a packet is decoded as soon as it is received).
- we prove that `XOR2` is throughput optimal (i.e. achieves capacity) in the following settings: 1) iid erasure channels and 2) independent (but not identical) channels with a maximum erasure probability up to 8/9, provided that a general condition, called DQEF (Double-Destination Queues Empty First), is satisfied.

§ This work is supported by the European Commission through the FP7 project STAMINA 265496.

978-1-61284-824-2/2012 - Copyright is with IFIP

- finally, we evaluate, via simulations, the performance of both algorithms under a multitude of different erasure profiles (including correlated erasure channels) and show that XOR2, in particular, performs so close to the capacity bound that it can be considered to achieve capacity for most practical purposes.

A theoretical analysis of the decoding delay for unicast traffic is expected to be more complex than for the broadcast case and lies outside this paper's scope. Hence, we make no claims regarding the delay optimality of the proposed algorithms. It should also be mentioned that both algorithms require per slot ACK/NACK but no knowledge of channel statistics.

The complexity-delay-throughput tradeoff for the N-user BEC channel has been investigated in many works (albeit for broadcast traffic), where the notion of delay is usually defined as: "a user experiences a delay of one slot if it receives a packet that is not innovative or a packet that it cannot decode. The total delay for a user is the cumulative delay experienced over all time slots". Under this definition, [5] presented a simple online XOR-based scheme for 2 users that is throughput optimal and has zero delay (for both users) for arbitrary channel statistics, while [7] proposed an offline algorithm with the same properties for 3 users. For $N > 3$, no zero delay algorithm exists so that the focus is shifted towards minimizing the (worst case or median) decoding delay. To this end, online heuristics are presented in [7], [8] (the latter also proves the NP-hardness of minimizing the average delay) while [9] described a more systematic way of choosing which packets to combine that requires the solution of a set packing problem (again, NP-hard) in each slot. A stochastic shortest path formulation of the same problem is also provided in [10].

Especially for the 3-user BEC, [11] considered the case of stochastic packet arrivals and proposed an online throughput optimal algorithm that is conjectured to have asymptotically optimal delay (as the traffic load goes to 1), in the sense that the delay of the 3-user BEC scales in exactly the same manner as for the single user BEC. This claim is supported by simulation results only. All of the previous works have focused on what is referred to as "strictly instantaneous decoding", which requires each transmitted packet to be selected such that it can be instantly decoded by *all* users (provided they receive it, of course). The work in [12] shows that this requirement may be too restrictive and it is possible, by relaxing the condition of instant decodability for all users at the transmitter, to design coding schemes that actually have lower delay. However, the price to pay is that an NP-hard maximum weight clique problem must be solved in each slot.

The above works are complementary to our work, since they consider broadcast traffic whereas we exclusively study unicast traffic. Additionally, although we consider an instant decodability constraint in our model, we do not examine the problem of directly minimizing decoding delay, since the latter notion is more difficult to handle for unicast traffic compared to broadcast. Finally, the proposed algorithms do not require the solution of complex optimization problems but, instead, move packets among queues in a virtual network depending on received feedback.

The rest of the paper is organized as follows. In Section II, the system model is specified and some notation is introduced. In Section III, the two algorithms are described and the theoretical results are presented. Section IV studies overhead issues and Section V presents numerical results for the two algorithms. Finally, Section VI concludes the paper. Due to space restrictions, some proofs and intermediate results are omitted and presented in [13] instead.

II. System model and notation

We consider a time-slotted system. The time unit is considered to be the time for transmission of a single bit. Packets of length L bits are transmitted during each slot, therefore the length of the slot is L time units. Time slot $[(l-1)\,L, lL]$, $l = 1, 2, ...$ is referred to as slot l.

The communication system consists of a single transmitter T and a set $\{U_1, U_2, U_3\}$ of three receivers . Each transmission can be received by any of the receivers (i.e., the transmission is broadcasted) without error, or may be erased by a subset of these receivers. After a packet transmission each receiver U_i, $i \in \mathcal{N} = \{1, 2, 3\}$, informs through a feedback channel the transmitter whether a packet erasure occurred at U_i. This feedback is not available to the other users.

We assume that the erasure events are independent and identically distributed across time, and arbitrarily correlated at a given time. We denote by $\epsilon_{\mathcal{S}}$, $\mathcal{S} \in \mathcal{N}$, the probability that a transmitted packet is erased at all receivers in the set $\{U_i,\, i \in \mathcal{S}\}$. For simplicity, in some places we abuse somewhat the notation and write ϵ_i, ϵ_{ij} instead of $\epsilon_{\{i\}}, \epsilon_{\{i,j\}}$.

At the transmitter there are packet sets $\mathcal{K}_i$, $i = 1, 2, 3$. The packets in set $\mathcal{K}_i$ must be delivered to U_i. We denote $K_i = |\mathcal{K}_i|$, where $|\mathcal{K}|$ is the number of elements of set $\mathcal{K}$.

Let Π be the set of permutations on $\mathcal{N}$, i.e. $\pi(i) : \mathcal{N} \to \mathcal{N}$, and define $\mathcal{S}_\pi(i) = \{\pi(1), .., \pi(i)\}$. It has been shown in [1], [2] that the capacity outer bound of the communication system under consideration is described by the following Lemma.

Theorem 1. *An outer bound to the 3-user capacity region is*

$$\mathcal{C}_{out} = \left\{ \boldsymbol{R} \geq \boldsymbol{0} : \max_{\pi \in \Pi} \left[\sum_{i=1}^{3} \frac{R_{\pi(i)}}{1 - \epsilon_{\mathcal{S}_\pi(i)}} \right] \leq L \right\}$$

where $\boldsymbol{R} = \{R_1, R_2, R_3\}$ denotes the vector of transmission rates to the three receivers.

III. Network coding algorithms

We provide next some terminology and an outline of the main design ideas of the algorithms to be presented.

Packets in the sets $\mathcal{K}_i$, $i \in \mathcal{N}$ are called "native". According to the algorithms to be described below, all transmitted packets are either native, or XOR combinations of native packets. To shorten the description, in the following, we say that a packet p is an XOR combination of native packets even when p consists of a single native packet. Receiver U_i is called a *listener* of a packet p if p is an XOR combination of packets that U_i has correctly received. We also say that U_i is a *destination* of a

packet p if either $p \in \mathcal{K}_i$ and has not been received yet by (i.e., is unknown to) U_i, or if p is an XOR combination of the form $p = q \oplus c$ where $q \in \mathcal{K}_i$ is unknown to U_i, and U_i is a listener of c. As will be seen, transmitted packets may have several receivers as destinations or listeners.

The main features of the algorithms to be presented are the following.

Basic Algorithmic Features

1) Any transmitted packet is an XOR combination of native packets.
2) If the XOR combination of a transmitted packet p contains a packet $q \in \mathcal{K}_i$, which is unknown to U_i, then U_i is a destination for p. Hence, since according to the definition of "destination" it holds either $p = q$ or $p = q \oplus c$, U_i can immediately decode q.

During the algorithms' operation, packets may be placed in various virtual queues, based on the received feedback. A general queue is characterized by two index sets $\mathcal{L}, \mathcal{D} \subset \mathcal{N}$, $\mathcal{L} \cap \mathcal{D} = \emptyset$, and is denoted by $Q^{\mathcal{L}}_{\mathcal{D}}$. For simplicity, we will denote queue $Q^{\{k\}}_{\{i,j\}}$ by Q^k_{ij}, and queue $Q^{\emptyset}_{\{i\}}$ by Q_i. At the beginning of the algorithms, all packets in $\mathcal{K}_i$, $i \in \mathcal{N}$, are placed in queue Q_i and the rest of the queues are empty. The algorithms' operation ensures that the following property holds for any packet $p^{\mathcal{L}}_{\mathcal{D}} \in Q^{\mathcal{L}}_{\mathcal{D}}$:

Property of packets $p^{\mathcal{L}}_{\mathcal{D}} \in Q^{\mathcal{L}}_{\mathcal{D}}$: Any U_i, $i \in \mathcal{D}$, is a destination for $p^{\mathcal{L}}_{\mathcal{D}}$ and any U_i, $i \in \mathcal{L}$, is a listener for $p^{\mathcal{L}}_{\mathcal{D}}$.

We classify queues into levels. Level $w \in \{1, 2, 3\}$ contains all queues $Q^{\mathcal{L}}_{\mathcal{D}}$ for which the cardinality of the set $\mathcal{L} \cup \mathcal{D}$ is w. We further distinguish level 3 queues into single-destination queues, which are of the form Q^{jk}_i, and double-destination queues, which are of the form Q^k_{ij}.

In general, the algorithms transmit XOR combinations of packets, at most one from each of the queues $Q^{\mathcal{L}}_{\mathcal{D}}$. While the specific choice of packets depends on the received feedback and the specific algorithm, the following rule always holds.

Basic Coding Rule

A set $\mathcal{P}$ of $m = |\mathcal{P}|$ packets, *at most one* from each of the virtual queues, can be combined (by XORing) into a single coded packet if $\mathcal{P} = \left\{p^{\mathcal{L}_1}_{\mathcal{D}_1}, \ldots, p^{\mathcal{L}_m}_{\mathcal{D}_m}\right\}$, where $\mathcal{D}_n \subseteq \mathcal{L}_r$, for all $r \neq n$ $n, r \in \{1, ..., m\}$. In other words, the destination set of each packet is a subset of the others packets' listener sets.

Note that the Basic Coding Rule implies that $\mathcal{D}_n \cap \mathcal{D}_r = \emptyset$, for all $r \neq n$, n, $r \in \{1, ..., m\}$. Indeed, if $i \in \mathcal{D}_n$ then since by the Basic Coding Rule $i \in \mathcal{L}_r$, and by definition $\mathcal{D}_r \cap \mathcal{L}_r = \emptyset$, if follows that $i \notin \mathcal{D}_r$.

Based on the description above, we have the following important property.

Lemma 2. *Assume that Algorithmic Features [1, 2] are satisfied at the beginning of slot t. Then, if a transmitted packet is composed using the Basic Coding Rule, Algorithmic Features [1, 2] are also satisfied at slot t+1.*

Proof: Let

$$p = \bigoplus_{l=1}^{m} p^{\mathcal{L}_l}_{\mathcal{D}_l}.$$

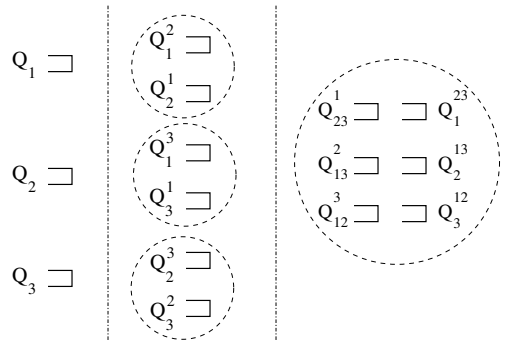

Figure 1. The group of virtual queues used by both algorithms.

Then, any receiver U_i with $i \in \cup_{l=1}^{m} \mathcal{D}_l$ is a destination of packet p. Indeed, if $i \in \mathcal{D}_n$, then by the definition of $Q^{\mathcal{L}_n}_{\mathcal{D}_n}$, packet $p^{\mathcal{L}_n}_{\mathcal{D}_n}$ has U_i as destination, i.e., $p^{\mathcal{L}_n}_{\mathcal{D}_n} = p_i \oplus c$, where $p_i \in \mathcal{K}_i$ is unknown to U_i and U_i is a listener of packet c. Also, since $\mathcal{D}_n \subseteq \mathcal{L}_r$, for all $r \neq n$, with $n, r \in \{1, ..., m\}$, U_i is a listener of all packets $p^{\mathcal{L}_r}_{\mathcal{D}_r}$, $r \neq n$. Hence $p = p_i \oplus c'$ where $c' = c \oplus_{r=1, r\neq n}^{m} p^{\mathcal{L}_r}_{\mathcal{D}_r}$ has U_i as listener. ■

According to the discussion above, if $p = \oplus_{l=1}^{m} p^{\mathcal{L}_l}_{\mathcal{D}_l}$ is transmitted, then any receiver U_i with $i \in \cup_{l=1}^{m} \mathcal{D}_l$ can correctly decode the native packet that is contained in the XOR combination and is unknown to U_i. Hence, a single transmission may potentially result in multiple receivers correctly decoding their packets. This is the main source of efficiency of the proposed coding schemes. Figure 1 shows all virtual queues which participate in the algorihms' execution.

In general, at any given time there will be several possibilities for combining packets from virtual queues according to the Basic Coding Rule. For example, Q^1_2 can be combined with either Q^2_1 or Q^{23}_1, with similar choices being available for the other queues. We will next present algorithms `XOR1`, `XOR2` that make specific choices on which queues are combined (and in which order) according to the Basic Coding Rule and analyze their performance. Both algorithms allow for certain packet transitions among the virtual queues of Figure 1 and can be regarded, in a sense, as the low complexity counterparts of algorithms `CODE1`, `CODE2` proposed in [2]. The main difference is that `XOR1`, `XOR2` are only allowed to use XOR operations which are performed in a manner that makes instantaneous decoding feasible. Although this complicates the queue structure and algorithm analysis compared to [2], it has the desirable properties of simple and instantaneous decoding at the receivers, which reduces the total delay. In addition, the algorithms have small overhead requirements and, though they cannot match the performance of `CODE1`, `CODE2` in terms of achieving capacity for a broad class of erasure profiles, they perform very well in practice.

The crux of both algorithms lies in the exact ways in which the virtual queues are combined and packets are moved among them (at the transmitter side) depending on feedback. Of course, in order for the algorithm to operate correctly, sufficient information should be contained in the packet headers so

that the receivers can decode correctly. We will postpone the discussion on the content of this information and the related overhead until Section IV.

A. Algorithm XOR1

1) Description: In the description of the two algorithms we will specify how packets move among virtual queues at the transmitter. As mentioned in the previous Section, all native packets are initially placed in queues Q_i, $i = 1, 2, 3$. The algorithm operates in phases as follows:

a) Phase 1: Phase 1 consists of subphases $1.\{i\}$, $i = 1, 2, 3$, during which packets from Q_i, $i = 1, 2, 3$ are transmitted respectively.

We describe next the operations during subphase $1.\{i\}$. Let packet $p \in Q_i$ be transmitted in a slot. If the packet is erased by all receivers, it is retransmitted. If the packet is seen by U_i, it is removed from Q_i. Otherwise, (i.e. if the packet is erased by U_i), denote with $\mathcal{L} \subseteq \mathcal{N} - \{i\}$ the set of users that receive the packet. The packet is then removed from the current queue Q_i and added to queue $Q_i^{\mathcal{L}}$ (possible choices are: Q_i^j, Q_i^k, Q_i^{jk}). This process is repeated until queue Q_i becomes empty.

b) Phase 2: In this phase, XOR combinations of packets from level 2 queues Q_i^j, $i, j \in \mathcal{N}$ are transmitted. Applying the Basic Coding Rule to this case, each transmitted packet p should either be of the form $p = p_i^j \oplus p_j^i$, where $p_i^j \in Q_i^j$ and $p_j^i \in Q_j^i$, or p should belong to one of the queues $Q_i^j, i, j \in \mathcal{N}$. This phase consists of 3 subphases, named $2.\{12\}$, $2.\{13\}$, $2.\{23\}$. We describe subphase $2.\{ij\}$. As long as both queues Q_i^j, Q_j^i are nonempty, XOR combinations of packets from Q_i^j, Q_j^i are transmitted, i.e. $p = p_i^j \oplus p_j^i$, and the following actions are taken based on the feedback received from the receivers.

1) If packet p is erased at all receivers, it is retransmitted.
2) If receiver U_i (resp. U_j) receives p, that receiver can decode its intended packet and hence packet p_i^j (p_j^i) is removed (dequeued) from Q_i^j (Q_j^i).
3) If p is erased by both U_i, U_j and U_k receives p, both packets p_i^j, p_j^i are removed from the corresponding queues and p is added to queue Q_{ij}^k. This is consistent with the queue notation, since both U_i, U_j are now destinations for p and U_k is listener of p.
4) If U_k and only one of U_i, U_j receives the packet (say p is erased by U_j and received by U_i), then packet p_j^i is removed from Q_j^i and p is added to queue Q_j^{ik} (according to step 2 packet p_i^j is also removed from Q_i^j). This is also consistent with the queue notation since both k, i received p and hence they are listeners of this packet, and j is a destination for p since $p = p_i^j \oplus p_j^i$. We summarize all these actions in Table I, where R/E denotes a reception/erasure, respectively.

If one queue of Q_j^i, Q_i^j is or becomes empty, uncoded packets from the surviving queue are transmitted until this queue also empties. We describe the actions in this case. Suppose Q_i^j is the surviving queue and packet p_i^j is transmitted

Table I
ACTIONS DURING SUBPHASE $2.\{ij\}$.

U_i	U_j	U_k	action
R	R	R	dequeue p_i^j, p_j^i , both U_i, U_j decode
R	R	E	dequeue p_i^j, p_j^i , both U_i, U_j decode
R	E	R	dequeue p_i^j, p_j^i, move p to Q_j^{ik}, only U_i decodes
R	E	E	dequeue only p_i^j, only U_i decodes
E	R	R	dequeue p_i^j, p_j^i, move p to Q_i^{jk}, only U_j decodes
E	R	E	dequeue only p_j^i, only U_j decodes
E	E	R	dequeuep_i^j, p_j^i, move p to Q_{ij}^k, no decoding
E	E	E	retransmit

1) If p_i^j is erased at both i, k it is retransmitted.
2) If p_i^j is received by U_i, it is removed from Q_i^j.
3) If p_i^j is erased at U_i and received by U_k then p_i^j is removed from Q_i^j and placed in Q_i^{jk}, which is again consistent with the queue notation.

c) Phase 3: At the beginning of phase 3, only level 3 queues, i.e. queues of the form Q_{ij}^k and Q_i^{jk}, may contain packets. According to the Basic Coding Rule, transmitted packets should have the form $p = p_{ik}^i$, $p = p_i^{ik}$, $p = p_i^{jk} \oplus p_{jk}^i$, $p = p_i^{jk} \oplus p_j^{ik}$, $p = p_1^{23} \oplus p_2^{13} \oplus p_3^{12}$.

During phase 3, XOR combinations containing at least one packet from queues Q_{jk}^i are transmitted. This phase consists of 3 subphases, denoted by $3.\{i\}, i = 1, 2, 3$. During subphase $3.\{i\}$ XOR combinations of packets from queues Q_{jk}^i, Q_i^{jk} are processed as follows. As long as both queues Q_{jk}^i, Q_i^{jk} are nonempty, XOR combinations of packets $p = p_i^{jk} \oplus p_{jk}^i$ are transmitted and the following actions are taken based on the feedback received from the receivers.

1) If no one receives the packet, it is retransmitted
2) If U_i receives p, then p_i^{jk} is removed from Q_i^{jk} and decoded.
3) If both U_j, U_k receive p, then p_{jk}^i is removed from Q_{jk}^i. In this case, both destinations of p_{jk}^i can recover their corresponding unknown packet contained in p_{jk}^i.
4) If only one, say U_j, of U_j, U_k receives p, then p_{jk}^i is removed from Q_{jk}^i and placed in Q_k^{ij}. This is consistent with the Basic Coding Rule for the following reason. By definition, p_{jk}^i has U_k as destination and U_i as listener. In addition, since U_j received $p = p_i^{jk} \oplus p_{jk}^i$ and U_j is a listener of p_i^{jk}, it can decode p_{jk}^i and hence, upon reception of p, U_j becomes a listener of p_{jk}^i.

The actions taken during subphase $3.\{i\}$ as long as both queues Q_{jk}^i, Q_i^{jk} are nonempty are summarized in Table II.

Subphase $3.\{i\}$ ends when Q_{jk}^i becomes empty. If during this subphase Q_i^{jk} empties first, then packets p_{jk}^i from Q_{jk}^i are transmittted and the following actions are taken based on the feedback received from the receivers.

1) If none of U_j, U_k receives p_{jk}^i, it is retransmitted.
2) If both U_j, U_k, receive p_{jk}^i, the packet is removed from Q_{jk}^i and both destinations of p_{jk}^i can recover the corresponding unknown packet contained in p_{jk}^i.

Table II
ACTIONS DURING SUBPHASE 3. $\{i\}$

U_i	U_j	U_k	action
R	R	R	dequeue p_i^{jk}, p_{jk}^i; all 3 users decode
R	R	E	dequeue p_i^{jk}, p_{jk}^i, move p_{jk}^i to Q_k^{ij}; U_i, U_j decode
R	E	R	dequeue p_i^{jk}, p_{jk}^i, move p_{jk}^i to Q_j^{ik}; U_i, U_k decode
R	E	E	dequeue p_i^{jk} ; U_i decodes
E	R	R	dequeue p_{jk}^i ; U_j, U_k decode
E	R	E	dequeue p_{jk}^i, move p_{jk}^i to Q_k^{ij}; U_j decodes
E	E	R	dequeue p_{jk}^i, move p_{jk}^i to Q_j^{ik}; U_k decodes
E	E	E	retransmit

3) If only one of U_j, U_k (say U_j) receives p_{jk}^i, then the packet is removed from Q_{jk}^k and is placed in Q_k^{ij}. It can again be seen that this action is consistent with the Basic Coding Rule.

d) Phase 4: At the end of phase 3, only the three single-destination queues Q_i^{jk} may contain packets. In this phase, as long as all three queues are nonempty, packets of the form $p = p_1^{23} \oplus p_2^{13} \oplus p_3^{12}$ are sent and the following actions are taken based on the feedback received from the receivers.

1) If p is erased at all receivers, it is retransmitted
2) If p is received by U_i, then p_i^{jk} is removed from Q_i^{jk}

If one of the queues (say Q_k^{ij}) empties first while the other two queues (say Q_i^{jk}, Q_j^{ik}) are nonempty, then packets of the form $p = p_i^{jk} \oplus p_j^{ik}$ are sent and the same rules as before are applied with an important addition: if p is only received by U_k, it is retransmitted. If, during this second stage, another queue (say Q_j^{ik}) becomes empty, the transmitter starts sending packets $p = p_i^{jk}$ from the remaining queue until they are received by U_i. Phase 4 ends when all single-destination level 3 queues are empty, at which point all queues are empty and XOR1 terminates.

2) Performance analysis: The determination of the throughput region of XOR1 is omitted due to lack of space and can be found in [13]. Through numerical evaluations it is observed that, on the average, XOR1 performs very well in respect to the capacity outer bound. However, in a few cases, the deviation from the capacity bound may exceed 30%. In section III-B, we present an algorithm, named XOR2, that improves performance in these cases. We theoretically prove that this algorithm achieves the capacity outer bound for the cases of i) i.i.d. channels and ii) independent channels for $\max_i \epsilon_i < 8/9$, at the cost of slightly increased complexity compared to XOR1.

B. Algorithm XOR2

Although algorithm XOR2 operates on the same virtual network of Figure 1 as XOR1, it differs from XOR1 in two distinct points:

- in phase 2, each of the subphases $2.\{ij\}$ now ends immediately when at least one of the level 2 queues Q_j^i, Q_i^j empties, i.e. there is no second part for any subphase as in XOR1. The packets of the surviving queues from phase 2 are combined with packets of level 3 single-destination queues at a subsequent phase.
- the subphases in phase 3 are now performed iteratively, possibly over non-contiguous slots, rather than being executed once. However, we still use the order $3.\{1\} \rightarrow 3.\{2\} \rightarrow 3.\{3\}$ for performing the iterations.

The above modifications are motivated by the CODE2 algorithm of [2] and aim in achieving higher throughput via more "intelligent" queue combinations. XOR2 consists of 5 phases, each of which is described in detail below (phase 1 is identical to that of algorithm XOR1 and is not repeated).

1) Description:

a) Phase 2: This phase is similar to that of XOR1 and is also performed in 3 subphases, named $2.\{12\}$, $2.\{13\}$, $2.\{23\}$, with the difference that subphase $2.\{ij\}$ ends when at least one of Q_i^j, Q_j^i empties. The actions taken when feedback is acquired from the receivers are still described by Table I. At the end of this phase, at least 3 queues of the form Q_i^j have emptied.

b) Phase 3: This phase is similar to phase 3 of XOR1 and also consists of 3 subphases, denoted by $3.\{i\}$, $i = 1, 2, 3$. During subphase $3.\{i\}$, XOR combinations of packets $p = p_i^{jk} \oplus p_{jk}^i$ are transmitted as long as both queues Q_{jk}^i, Q_i^{jk} are nonempty, and the actions described in Table II are performed. Subphase $3.\{i\}$ ends when Q_{jk}^i becomes empty, with the following important modification over XOR1. Recall that if, during this subphase, queue Q_i^{jk} empties first, XOR1 keeps transmitting packets from Q_{jk}^i. However, if the same event occurs under XOR2, phase $3.\{i\}$ temporarily pauses and the next subphase in the order $3.\{1\} \rightarrow 3.\{2\} \rightarrow 3.\{3\}$ begins. This happens because, as can be seen in Table II, packets may be moved to queues Q_j^{ik}, Q_k^{ij} during subphase $3.\{i\}$.

Therefore, even if a single-destination level 3 queue (say, Q_i^{jk}) becomes empty, it is advantageous to switch to a different subphase hoping that some new packets may be added to Q_i^{jk} so that subphase $3.\{i\}$ can resume. Therefore, after processing each pair of single/double-destination level 3 queues, subphase $3.\{i\}$ will continue if both Q_{jk}^i, Q_i^{jk} are nonempty. This iterative procedure is performed until all 3 pairs contain at least one empty queue (this implies that subphase $3.\{i\}$ may be executed in non-contiguous slots). At this point, if Q_{jk}^i has survived, the transmitter starts sending packets p_{jk}^i from Q_{jk}^i and the same actions as in the second part of phase $3.\{i\}$ of XOR1 are performed.

c) Phase 4: In this phase, XOR combinations of packets in the remaining level 2 queues and in single-destination level 3 queues are transmitted. Applying the Basic Coding Rule, transmitted packets can be of the form $p = p_i^j \oplus p_j^{ik}$, $p = p_i^j$, $p = p_j^{ik}$,$p = p_j^{ik} \oplus p_i^{jk}$ $p = p_1^{23} \oplus p_2^{13} \oplus p_3^{12}$. This phase consists of 3 subphases, named $4.\{12\}$, $4.\{13\}$, $4.\{23\}$. We describe subphase $4.\{ij\}$. As long as both queues Q_i^j, Q_j^{ik} are nonempty, XOR combinations of packets from Q_i^j, Q_j^{ik} are transmitted, i.e. $p = p_i^j \oplus p_j^{ik}$, and the following actions are taken based on received feedback.

1) If packet p is erased at all receivers, it is retransmitted.

2) If U_i receives p, then p_i^j is removed from Q_i^j.
3) If U_j receives p, then p_j^{ik} is removed from Q_j^{ik}.
4) If only U_k receives p, then p_i^j is removed from Q_i^j and moved to Q_i^{jk}, as p_i^j can be recovered by U_k, which is consistent with the queue notation.
5) If U_i and U_k receive p, then p_i^j is removed from Q_i^j.
6) If U_j and U_k receive p, then $p_j^{ik}$is removed from Q_j^{ik}, p_i^j is removed from Q_i^j and moved to Q_i^{jk}, which is again consistent with the queue notation.

The actions taken during subphase $4.\{ij\}$ as long as both queues Q_i^j, Q_j^{ik} are nonempty are summarized in Table III.

Table III
ACTIONS DURING SUBPHASE $4.\{ij\}$

U_i	U_j	U_k	action
R	R	R	dequeue p_i^j, p_j^{ik}; U_i, U_j decode
R	R	E	dequeue p_i^j, p_j^{ik}; U_i, U_j decode
R	E	R	dequeue p_i^j ; U_i decodes
R	E	E	dequeue p_i^j ; U_i decodes
E	R	R	dequeue p_i^j, p_j^{ik}, move p_j^j to Q_i^{jk}; U_j decodes
E	R	E	dequeue p_j^{ik}; U_j decodes
E	E	R	dequeue p_i^j, move p_i^j to Q_i^{jk}
E	E	E	retransmit

Subphase $4.\{ij\}$ ends when Q_i^j becomes empty. If during this subphase Q_j^{ik} empties first, then packets p_i^j from Q_i^j are transmittted and the actions taken based on the feedback received from the receivers are the same as in the second part of subphase $2.\{ij\}$ of XOR1, described in Section III-A1b.

d) Phase 5: This phase is exactly the same as Phase 4 of XOR1.

2) Performance analysis: The algorithm analysis consists of specifying the throughput region achieved by XOR2. Due to space restrictions it is presented in details in [13], while the main results of this analysis are presented below. The following definition simplifies the analysis and is necessary for Lemmas 4 and 6.

Definition 3. Node a *dominates* node b, written as $a \succeq b$, iff at the end of subphase $2.\{ab\}$ queue Q_b^a empties.

Note that this order depends implicitly on the number of packets that each user must receive (equivalently, on the achieved rates). It is shown in [13] that $a \succeq b$ and $b \succeq c$ imply $a \succeq c$. Hence we have a "preorder relation". Assume now without loss of generality that $a \succeq b$. We have two cases.

- $c \succeq b$. Then, by definition 3, there are no packets destined for receiver b in level 2 queues.
- $b \succeq c$. Then, according to the preorder relation, we must have $a \succeq c$ and conclude again that there are no packets destined for receiver c in level 2 queues.

Thus, it is certain that at the end of phase 2, there will be at least one user with no packets destined for it in level 2 queues. As can be seen, it is always possible to find $a, b, c \in \{1, 2, 3\}$ such that it holds $a \succeq b \succeq c$.

Lemma 4. *Consider the order $a \succeq b \succeq c$. If, under the application of XOR2, the queues Q_{bc}^a, Q_{ac}^b become empty at the end of the first part of phase 3 (we refer to this condition as DQEF - Double-destination Queues Empty First), i.e. when we have finished processing the pairs (Q_{23}^1, Q_1^{23}), (Q_{13}^2, Q_2^{13}), (Q_{12}^3, Q_3^{12}), then XOR2 achieves all rates that induce this order.*

Proof: [outline] The proof of Lemma 4 consists of computing the number of slots for all phases, for a certain order $a \succeq b \succeq c$ and under the assumption that queues Q_{bc}^a, Q_{ac}^b become empty at the end of the first part of phase 3. Without loss of generality we assume $(a, b, c) = (3, 2, 1)$, so the order $3 \succeq 2 \succeq 1$ holds, which means that at the end of Phase 2 queues Q_1^2, Q_1^3 and Q_2^3 are empty. We also assume that the respective queues Q_{12}^3, Q_{13}^2 become empty at phase 3. Therefore, the remaining level 2 queues are Q_2^1, Q_3^1 and Q_3^2 and the remaining level 3 queues are Q_2^{13}, Q_3^{12} and either Q_1^{23} or Q_{23}^1. Denoting with T^* the expectation of the total number of slots for Phases 1 to 5 we find [13]

$$T^* = \max\left\{ \frac{K_1}{1-\epsilon_1} + \frac{K_2}{1-\epsilon_{123}} + \frac{K_1}{1-\epsilon_{13}},\ \frac{K_1}{1-\epsilon_{123}} + \frac{K_2}{1-\epsilon_2} + \frac{K_3}{1-\epsilon_{23}},\ \frac{K_1}{1-\epsilon_{123}} + \frac{K_2}{1-\epsilon_{23}} + \frac{K_3}{1-\epsilon_3} \right\}, \tag{1}$$

so that, for the order relation $3 \succeq 2 \succeq 1$, the achievable region of XOR2 is

$$\left\{ \boldsymbol{R} :\ \max\left\{ \frac{R_1}{1-\epsilon_1} + \frac{R_2}{1-\epsilon_{123}} + \frac{R_3}{1-\epsilon_{13}},\ \frac{R_1}{1-\epsilon_{123}} + \frac{R_2}{1-\epsilon_2} + \frac{R_3}{1-\epsilon_{23}},\ \frac{R_1}{1-\epsilon_{123}} + \frac{R_2}{1-\epsilon_{23}} + \frac{R_3}{1-\epsilon_3} \right\} \le 1 \right\}, \tag{2}$$

which coincides with the set of all rate vectors in $\mathcal{C}_{out}$ (i.e. the capacity outer bound of Theorem 1) that also satisfy the specific order relation. ■

The derivation of (1) explicitly uses the law of large numbers (as $\min_i K_i \to \infty$) to replace all random variables (e.g. the number of slots required until the proposed algorithms terminate) with their expectations. A consequence of Lemma 4 is the following optimality condition.

Corollary 5. *If condition DQEF of Lemma 4 is satisfied for all $\boldsymbol{R} \ge \mathbf{0}$, algorithm XOR2 achieves capacity.*

3) Channels for which XOR2 achieves capacity: Notice that the optimality of XOR2 in the above Corollary depends upon compliance to condition DQEF. Since this condition also depends on the channel statistics, it is important to examine whether there exist channel erasure profiles that satisfy DQEF for all $\boldsymbol{R} \ge \mathbf{0}$. Based on the algorithm's analysis presented in [13] the following result is derived, which presents two such common erasure profiles:

Lemma 6. *Condition DQEF of Lemma 4 is satisfied for all $\boldsymbol{R} \ge \mathbf{0}$ (hence, XOR2 achieves capacity) in the following settings:*

- *iid channels, for all $0 \le \epsilon \le 1$.*
- *spatially independent, but not identical channels, for which it holds $\max(\epsilon_1, \epsilon_2, \epsilon_3) < 8/9$.*

In addition to the above theoretical result, numerical results presented in Section V indicate that XOR2 performs optimally in the vast majority of cases for general channel statistics.

IV. Overhead analysis

It is important to deal with overhead issues at this point. From the algorithms' construction, it follows that the receivers must always know exactly which native packets are included in each transmitted packet. With this information, according to Algorithmic Feature 2, a destination is able to immediately decode the native packet destined for it. To accomplish this, every native packet is equipped with a *Packet ID*, which consists of the packet's destination and a sequence number (according to the storing sequence in the initial queue Q_i, containing all packets with destination U_i, $i \in \{1,2,3\}$, at the transmitter). Every time two or more packets are XOR-combined, the new packet keeps in its header all Packet IDs of the native packets comprising it. The packet's destination needs 2 bits to be declared, while the sequence number needs $log_2 K_i$ bits for the packets of user i. The maximum number of native packets that can be combined with these algorithms is 6, so the overhead is $6\,(2 + log_2 K_i)$. For $K_i = 10^6$ and packet length $L = 12000$ the overhead is approximately 1.1%.

V. Numerical results

Although the theoretical results in Section III-B2 show the optimality of XOR2 under certain channel statistics, it is interesting to examine the performance of XOR1, XOR2 in more general settings than those considered in Lemma 6. Specifically, we will provide numerical answers to the following questions: how well does XOR2 perform for correlated channels or for independent channels when the erasure probabilities are higher than 8/9 (notice that the latter case is mainly of theoretical interest since a channel with such erasure probabilities would be practically useless)? Additionally, what is the margin of improvement over XOR1 when using XOR2?

To simplify the evaluation process, we compare the number of slots required by each algorithm (which we denote with T_1^*, T_2^* for XOR1, XOR2, respectively) with the optimal number of slots T_{OPT}^*. This parameter is calculated from Theorem 1 by substituting $K_{\pi(i)}$ for $R_{\pi(i)}$, that is $T_{OPT}^* = \max_{\pi\in\Pi}\left[\sum_{i=1}^{3} \frac{K_{\pi(i)}}{1-\epsilon_{\{\pi(1),\ldots,\pi(i)\}}}\right]$. As previously mentioned, T_1^*, T_2^*, T_{OPT}^* express mean values, since we assume K_i to be sufficiently large to invoke the law of large numbers. We define the relative deviation of XOR1 from optimality as $e = (T_1^* - T_{OPT}^*)/T_{OPT}^*$ (with a similar expression for XOR2). Parameters T_1^*, T_2^* depend on the ratios between the number of packets K_1, K_2, K_3 for each user as well as on the channel statistics, which are described in the most general case from the tuple $(\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_{12}, \epsilon_{13}, \epsilon_{23}, \epsilon_{123})$. Thus, we compute the deviation for a large number of different $K_1 : K_2 : K_3$ ratios, performing a sweep over all valid erasure tuples, and find its average and maximum value. For the general case of correlated channels, the results are summarized in Table IV, which indicates that, although both algorithms perform very well on the average, XOR2 has a significant worst case performance benefit over XOR1. Furthermore, XOR2 performs optimally in 95.8% of all simulated cases, with a maximum deviation of 5% for the rest.

Table IV
Performance of XOR1, XOR2 for general channel statistics.

	relative deviation d from optimal				
K_1:K_2:K_3	1:1:1	1:2:3	1:2:5	1:3:4	1:3:5
XOR1 (max)	31.6%	33.3%	32.4%	33.3%	32.9%
XOR1 (avg)	1.5%	1.4%	1.2%	1.5%	1.3%
XOR2 (max)	4.1%	3.6%	3.3%	3.3%	3.2%
XOR2 (avg)	0.004%	0.003%	0.003%	0.003%	0.003%
K_1:K_2:K_3	1:3:6	1:4:7	1:5:5	1:5:10	1:1:10
XOR1 (max)	32.7%	32.4%	32.6%	32.4%	30.8%
XOR1 (avg)	1.3%	1.3%	1.4%	1.2%	0.7%
XOR2 (max)	3.1%	2.9%	3.2%	2.4%	1.8%
XOR2 (avg)	0.003%	0.003%	0.003%	0.002%	0.003%

For the case of independent channels with erasure probabilities higher than 8/9, we calculated the maximum XOR2 deviation e from optimal, performing a sweep over all possible $K_1 : K_2 : K_3$ ratios. The maximum deviation was computed as 0.3% for $(\epsilon_1, \epsilon_2, \epsilon_3) = (0.98, 0.98, 0.93)$. Clearly, XOR2 can be considered as a practically capacity achieving algorithm for independent channels. Furthermore, in all simulated cases where XOR2 failed to perform optimally, at least one queue of the form Q_i^{jk} received no packets during Phase 1, as the probability $\epsilon_i - \epsilon_{ij} - \epsilon_{ik} + \epsilon_{123}$ of a packet moving to that queue was zero. This in turn negated the advantage of combining the single and double-destination level 3 queues in Phase 3, and eventually led to suboptimal performance.

VI. Conclusions

This paper presented two XOR-based coding algorithms, named XOR1 and XOR2, for the 3-user BEC channel with multiple unicast traffic. The algorithms are based on inter-session packet combining and careful exploitation of feedback, and operate on a specially constructed virtual network maintained by the transmitter, while allowing for instantaneous decoding. Simulations have shown that both algorithms perform very well with XOR2 achieving higher throughput at the cost of greater sophistication. XOR2, in particular, achieves capacity for the i.i.d. and independent (but not identical) channels with a maximum erasure probability of 8/9. Future work in this area could be aimed towards a generalization of the above XOR algorithms for $N > 3$ receivers. We note that the notion of virtual queues and the Basic Coding Rule described in Section III apply to general N. However, the number of possible choices for combining packets increases exponentially with N and it is not clear apriori which of these choices will lead to better performance. Also, in this work we assumed that all packets to be transmitted are present in the system at the beginning of time. In future work it would be interesting to design policies that assume that the packets arrive in a stochastic manner.

References

[1] C.-C. Wang, "Capacity of 1-to-k broadcast erasure channels with channel output feedback," in *Proc. 48th Annual Allerton Conference*,

October 2010, submitted to IEEE Trans. Inform. Theory. [Online]. Available: http://arxiv.org/abs/1010.2436v1

[2] M. Gatzianas, L. Georgiadis, and L. Tassiulas, "Multiuser broadcast erasure channel with feedback — capacity and algorithms," *submitted to IEEE Trans. Inform. Theory*. [Online]. Available: http://arxiv.org/abs/1009.1254

[3] A. Dana, R. Gowaikar, R. Palanki, B. Hassibi, and M. Effros, "Capacity of wireless erasure networks," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 789–804, March 2006.

[4] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: practical wireless network coding," vol. 16, no. 3, pp. 497–510, June 2008.

[5] M. Durvy, C. Fragouli, and P. Thiran, "Towards reliable broadcasting using acks," in *Proc. IEEE ISIT*, June 2007.

[6] L. Georgiadis and L. Tassiulas, "Broadcast erasure channel with feedback — capacity and algorithms," in *Proc. 5th Workshop on Network Coding Theory and Applications*, June 2009, pp. 54–61.

[7] L. Keller, E. Drinea, and C. Fragouli, "Online broadcasting with network coding," in *Proc. 4th Workshop on Network Coding, Theory and Applications*, June 2008.

[8] E. Drinea, C. Fragouli, and L. Keller, "Delay with network coding and feedback," in *Proc. IEEE ISIT*, July 2009.

[9] P. Sadeghi, D. Traskov, and R. Koetter, "Adaptive network coding for broadcast channels," in *Proc. 5th Workshop on Network Coding, Theory and Applications*, June 2009.

[10] S. Sorour and S. Valaee, "On minimizing broadcast completion delay for instantly decodable network coding," in *Proc. IEEE ICC*, May 2010.

[11] J. Sundararajan, D. Shah, and M. Médard, "Online network coding for optimal throughput and delay — the three-receiver case," in *Proc. International Symposium on Information Theory and its Applications*, December 2008.

[12] S. Sorour and S. Valaee, "Minimum broadcast decoding delay for generalized instantly decodable network coding," in *Proc. IEEE GLOBECOM*, December 2010.

[13] S. Athanasiadou, M. Gatzianas, L. Georgiadis, and L. Tassiulas, "Low complexity XOR-based coding algorithms for the 3-user broadcast erasure channel with feedback." [Online]. Available: http://users.auth.gr/∼leonid/Public/TechReports/tecreport_XOR.pdf