Multiuser Broadcast Erasure Channel with Feedback — Capacity and Algorithms

Marios Gatzianas*, Leonidas Georgiadis*§ and Leandros Tassiulas†§
*Department of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Thessaloniki, 54 124, Greece.
†Computer Engineering and Telecommunications Department, University of Thessaly, Volos, 38 221, Greece.
§Center for Research and Technology Hellas, Thermi, Thessaloniki, 60361, Greece.
Emails: {mgkatzia, leonid}@auth.gr, leandros@uth.gr

***Abstract*—We consider the N-user broadcast erasure channel where feedback from the users is fed back to the transmitter in the form of ACK messages. We provide a generic outer bound to the capacity of this system and propose a coding algorithm that achieves this bound for an arbitrary number of users and symmetric channel conditions, assuming that instantaneous feedback is known to all users. Removing this assumption results in a rate region which differs from the outer bound by a factor $O(N^2/L)$, where L is packet length. For the case of non-symmetric channels, we present a modification of the previous algorithm whose achievable region is identical to the outer bound for $N \leq 3$, when instant feedback is known to all users, and differs from the bound by $O(N^2/L)$ when each user knows only its own ACK. The proposed algorithms do not require any prior knowledge of channel statistics.**

***Index Terms*—Broadcast erasure channels, feedback-based coding, capacity achieving algorithms.**

I. Introduction

Broadcast channels have been extensively studied by the information theory community since their introduction in [1]. Although their capacity remains unknown in the general case, special cases have been solved, including the important category of “degraded” channels [2]. Another class of channels that has received significant attention is erasure channels, where either the receiver receives the input symbol unaltered or the input symbol is erased (equivalently, dropped) at the receiver. The latter class is usually employed as a model for lossy packet networks.

Combining the above classes, a broadcast erasure channel (BEC) is a suitable abstraction for wireless communications modeling since it captures the essentially broadcast nature of the medium as well as the potential for packet loss (due to fading, packet collision etc). Since this channel is not necessarily degraded, the computation of its feedback capacity region is an open problem. Numerous variations of this channel, under different assumptions, have been studied, a brief summary of which follows.

For multicast traffic, an outer bound to the capacity region of erasure channels is derived in [3], in the form of a suitably defined minimum cut, and it is proved that the bound can be achieved by linear coding at intermediate nodes. The broadcast nature is captured by requiring each node to transmit the same signal on all its outgoing links, while it is assumed that the destinations have complete knowledge of any erasures that occurred on *all* source-destination paths. In a sense, [3] is the “wireless” counterpart to the classical network coding paradigm of [4], since it carries all results of [4] (which were based on the assumption of error-free channels) into the wireless regime.

The concept of combining packets for efficient transmission based on receiver feedback is also used in [5], where broadcast traffic is assumed and a rate-optimal, zero-delay, offline algorithm is presented for $N = 3$. Online heuristics that attempt to minimize the decoding delay are also presented. Reference [6] expands on this work by presenting an online algorithm that solves at each slot a (NP-hard) set packing problem in order to decide which packets to combine. This algorithm also aims in minimizing delay.

Multiple unicast flows, which are traditionally difficult to handle within the network coding paradigm, are studied in [7] for a network where each source is connected to a relay as well as to all destinations, other than its own, and all connections are modeled as BECs. A capacity outer bound is presented for arbitrary N and is shown to be achievable for $N = 3$ and almost achievable for $N = 4, 5$. The capacity-achieving algorithm operates in two stages with the relay having knowledge of the destination message side information at the end of the first stage but not afterward (i.e. once the second stage starts, the relay does not receive feedback from the destinations).

A similar setting is studied in [8], where ACK-based packet combining is proposed and emphasis is placed on the overhead and complexity requirements of the proposed scheme. An actual implementation of the use of packet XORing in an intermediate layer between the IP and 802.11 MAC layers is presented and evaluated in [9], while [10] proposes a replacement for the 802.11 retransmission scheme based on exploiting knowledge of previously received packets.

This paper expands upon earlier work in [11] (which studied the case $N = 2$) and is sufficiently different from the aforementioned work in that, although it also uses the idea of packet mixing (similar to the network coding sense), it provides explicit performance guarantees. Specifically, an outer bound to the feedback capacity region for multiple unicast flows (one

‡This work was supported by the EU project N-CRAVE, FP7-215252.

for each user) is computed and two online algorithms are presented that achieve this bound for the following settings, respectively: an arbitrary number of users N with symmetric channels (this concept will be defined later), and 3 users with arbitrary channel statistics.

The algorithms do not require any knowledge of channel parameters (such as erasure probabilities) or future events so that they can be applied to any BEC. They use receiver feedback to combine packets intended for different users into a single packet which is then transmitted. The combining scheme (i.e. choosing which packets to combine and how) relies on a set of virtual queues, maintained in the transmitter, which are updated based on per-slot available receiver ACK/NACKs. This queue-based coding concept has also been used in [12], albeit for broadcast traffic with stochastic arrivals where the stability region of the proposed algorithm becomes asymptotically optimal as the erasure probability goes to 0, whereas we consider systems with an arbitrarily fixed number of packets per unicast stream where the capacity is achieved for arbitrary values of erasure probability.

The paper is structured as follows. Section II describes the exact model under investigation and provides the necessary definitions in order to derive the capacity outer bound in Section III. The first coding algorithm is presented in Section IV, which also contains a discussion of the intuition behind the algorithm, its correctness and optimal performance for symmetric channels. The incorporation of overhead and the corresponding reduction in the achievable region are also examined. A modification of the algorithm that achieves capacity for 3 users under arbitrary channel conditions is presented in Section V, while Section VI concludes the paper. Due to space restrictions, the proofs of all stated results are omitted and presented in [13] instead.

II. System model and definitions

The system model is a direct extension to N users of the corresponding model in [11] but is nonetheless repeated for completeness. Consider a time slotted system where messages (packets) of length L bits are transmitted in each slot. We normalize to unity the actual time required to transmit a single bit so that the time interval $[(l-1)L\ lL)$, for $l = 1, 2, \ldots$, corresponds to slot l. The system consists of a single transmitter and a set $\mathcal{N} \triangleq \{1, 2, \ldots, N\}$ of receivers, while there exists at the transmitter a distinct stream of unicast packets for each receiver, with the packets destined for receiver i comprising set $\mathcal{K}_i$. The channel is modeled as memoryless broadcast erasure so that each broadcast packet is either received unaltered by a user or is dropped (i.e. the user does not receive it), in which case an erasure occurs for the user. This is equivalent to considering that the user receives the special symbol E, which is distinct from any transmitted symbol. Hence, each user knows whether an erasure has occurred or not by examining its received symbol.

Define $Z_{i,l} \triangleq \mathbb{I}[\text{user } i \text{ receives } E \text{ in slot } l]$, where $\mathbb{I}[\cdot]$ denotes an indicator function, and consider the random vector $\boldsymbol{Z}_l = (Z_{1,l}, Z_{2,l}, \ldots, Z_{N,l})$. The sequence $\{\boldsymbol{Z}_l\}_{l=1}^{\infty}$ is assumed to consist of iid vectors (we denote with $\boldsymbol{Z} = (Z_1, \ldots, Z_N)$ the random vector with distribution equal to that of $\boldsymbol{Z}_l$), although, for a fixed slot, arbitrary correlation between erasures in different users is allowed. For any index set $\mathcal{I} \subseteq \mathcal{N}$, we define the probability that an erasure occurs to all users in $\mathcal{I}$ as

$$\Pr(Z_i = 1,\ \forall\, i \in \mathcal{I}) \triangleq \varepsilon_{\mathcal{I}}, \tag{1}$$

where, by convention, it holds $\varepsilon_{\varnothing} = 1$. For simplicity, we write ε_i instead of $\varepsilon_{\{i\}}$ and assume $\varepsilon_i < 1$ to avoid trivial cases.

According to the introduced notation, when the transmitter, at the beginning of slot l, broadcasts symbol X_l, each user i receives symbol $Y_{i,l}$ given by

$$Y_{i,l} = Z_{i,l}E + (1 - Z_{i,l})X_l, \tag{2}$$

where we denote $\boldsymbol{Y}_l \triangleq (Y_{i,l})_{i\in\mathcal{N}}$. At the end of each slot l, all users inform the transmitter whether the symbol was received or not, which is equivalent to each user i sending the value of $Z_{i,l}$ through an error-free control channel. In information-theoretic terms [14], the broadcast channel is described by the input alphabet $\mathcal{X}$, the output alphabets $\mathcal{Y}_1, \mathcal{Y}_2, \ldots, \mathcal{Y}_N$ for users $1, 2 \ldots, N$, respectively, and the probability transition function $p(\boldsymbol{Y}_l|X_l)$. Due to the memoryless property, the transition probability function is independent of l, so that it can be written as $p(\boldsymbol{Y}|X)$. In the rest of the paper, we set $\mathcal{X} = \mathbb{F}_q$, with $\mathbb{F}_q$ a suitable field of size q, so that, by definition of erasure channel, it holds $\mathcal{Y}_i = \mathcal{X} \cup \{E\}$ for all $i \in \mathcal{N}$.

A channel code $(2^{nR_1}, \ldots, 2^{nR_N}, n)$ for the broadcast channel with feedback consists of the following components:

- message sets $\mathcal{W}_i$ of size 2^{nR_i} for each user $i \in \mathcal{N}$. Denote $\boldsymbol{W} = (W_1, \ldots, W_N) \in \mathcal{W}_1 \times \ldots \times \mathcal{W}_N$.
- an encoder that at slot l transmits symbol X_l based on the value of $\boldsymbol{W}$ and all previously gathered feedback $\boldsymbol{Y}^{l-1} \triangleq (\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_{l-1})$. X_1 is a function of $\boldsymbol{W}$ only.
- N decoders, one for each user $i \in \mathcal{N}$, represented by the functions $g_i : \mathcal{Y}_i^n \to \mathcal{W}_i$.

A decoding error occurs with probability $P_e = \Pr(\cup_{i\in\mathcal{N}}\{g_i(Y_i^n) \neq W_i\})$, where $Y_i^n \triangleq (Y_{i,1}, \ldots, Y_{i,n})$. A rate $\boldsymbol{R} = (R_1, \ldots, R_N)$ is achievable if there exists a sequence of channel codes $(2^{nR_1}, \ldots, 2^{nR_N}, n)$ such that $P_e \to 0$ as $n \to \infty$. Finally, the capacity region of this channel is defined as the closure of the set of achievable rates.

The following definition, introduced in [2], will be useful in deriving the outer bound for the capacity of the broadcast erasure channel.

Definition 1: A broadcast channel $(\mathcal{X}, (\mathcal{Y}_i)_{i\in\mathcal{N}}, p(\boldsymbol{Y}|X))$ with receiver set $\mathcal{N}$ is physically degraded if there exists a permutation π on $\mathcal{N}$ such that the sequence $X \to Y_{\pi(1)} \to \ldots \to Y_{\pi(N)}$ forms a Markov chain.

A generalization to N users of the 2-user proof in [15] provides the following remarkable result.

Lemma 1: Feedback does not increase the capacity region of a physically degraded broadcast channel.

We now have all necessary tools to compute the actual capacity outer bound.

III. Capacity outer bound

The derivation of the capacity outer bound is based on a method similar to the approaches in [16]–[18]. We initially state a general result on the capacity of broadcast erasure channels *without feedback* [19].

Lemma 2: The capacity region (measured in information bits per transmitted symbol) of a broadcast erasure channel with receiver set $\mathcal{N}$ and no feedback is

$$\mathcal{C}_{nf} = \left\{ \boldsymbol{R} \geq \boldsymbol{0} : \sum_{i\in\mathcal{N}} \frac{R_i}{1-\varepsilon_i} \leq L \right\}, \tag{3}$$

which implies that any achievable rate is achieved through simple timesharing between the users.

We denote with C the channel under consideration and, for an arbitrary permutation π on $\mathcal{N}$, introduce a new, hypothetical, broadcast channel $\hat{C}_\pi$ with the same input/output alphabets as C and an erasure indicator function of

$$\hat{Z}_{\pi(i),l} = \prod_{j=1}^{i} Z_{\pi(j),l}. \tag{4}$$

In other words, a user $\pi(i)$ in $\hat{C}_\pi$ erases a symbol if and only if all users $\pi(j)$, with $j \leq i$, erase the symbol in channel C. This occurs with probability $\hat{\varepsilon}_{\pi(i)} \triangleq \varepsilon_{\cup_{j=1}^{i}\{\pi(j)\}}$. The following two results are proved in [13].

Lemma 3: Channel $\hat{C}_\pi$ is physically degraded.

Lemma 4: Denote with $\mathcal{C}_f$, $\hat{\mathcal{C}}_{\pi,f}$ the feedback capacity regions of channels C, $\hat{C}_\pi$, respectively. It holds $\mathcal{C}_f \subseteq \hat{\mathcal{C}}_{\pi,f}$.

Notice that Lemma 4 already provides an outer bound to $\mathcal{C}_f$. In order to derive this bound, we note that the previous results imply that the feedback capacity region of the physically degraded channel $\hat{C}_\pi$ is identical, due to Lemma 1, to the capacity region of $\hat{C}_\pi$ without feedback. The latter is described, in general form, in Lemma 2 whence we deduce the following result.

Lemma 5: The feedback capacity region of $\hat{C}_\pi$ is given by

$$\hat{\mathcal{C}}_{\pi,f} = \left\{ \boldsymbol{R} \geq \boldsymbol{0} : \sum_{i\in\mathcal{N}} \frac{R_{\pi(i)}}{1-\hat{\varepsilon}_{\pi(i)}} \leq L \right\}. \tag{5}$$

The above analysis was based on a particular permutation π. Considering all $N!$ permutations on $\mathcal{N}$ provides a tighter general outer bound.

Theorem 1: The following set inclusion is true

$$\mathcal{C}_f \subseteq \mathcal{C}_{out} \triangleq \cap_{\pi\in\mathcal{P}} \hat{\mathcal{C}}_{\pi,f}, \tag{6}$$

where $\mathcal{P}$ is the set of all possible permutations on $\mathcal{N}$.

IV. Coding algorithm

In this section, we present a coding algorithm named `CODE1`, show its correctness, and analyze its performance for symmetric channels, i.e. channels which satisfy the condition $\varepsilon_\mathcal{I} = \varepsilon_\mathcal{J}$ whenever $|\mathcal{I}| = |\mathcal{J}|$, for any $\mathcal{I}, \mathcal{J} \subseteq \mathcal{N}$. To indicate this special setting, we introduce the notation $\epsilon_{|\mathcal{I}|} \triangleq \varepsilon_\mathcal{I}$ (i.e. the subscript of ϵ indicates the cardinality of the erasure set). In the following, we assume that each user knows the size $|\mathcal{K}_i|$ of all streams and instant feedback is available to all users. The first assumption can be easily satisfied in practice while the second one will be removed in a later section.

Before the algorithm's description, a brief discussion of the underlying rationale will be useful. Since each user i must decode exactly $|\mathcal{K}_i|$ packets, one way of achieving this is by sending linear combinations, over the field $\mathbb{F}_q$, of appropriate packets so that user i eventually receives $|\mathcal{K}_i|$ linearly independent combinations of the packets in $\mathcal{K}_i$. Specifically, all stream packets are viewed as elements of $\mathbb{F}_q$, while each transmitted symbol (or packet) s has the form

$$s = \sum_{p\in\cup_{i\in\mathcal{N}}\mathcal{K}_i} a_s(p)p, \tag{7}$$

where $a_s(p)$ are suitable coefficients in $\mathbb{F}_q$. If the symbol s can also be written as

$$s = \sum_{p\in\mathcal{K}_i} b_s(p)p + c_s, \tag{8}$$

where $\boldsymbol{b}_s \triangleq (b_s(p),\ p \in \mathcal{K}_i)$, c_s are known to user i, then s is considered to be a "token" for i. Additionally, if s is received by i and the $\boldsymbol{b}_s$ coefficients of s, along with the $\boldsymbol{b}_{s'}$ coefficients of all previously received (by i) tokens s', form a linearly independent set of vectors over $\mathbb{F}_q$, then s is considered to be an "innovative token" for i. In words, an innovative token for i is any packet s that allows i to effectively construct a new equation (with the packets in $\mathcal{K}_i$ as unknowns, since $\boldsymbol{b}_s$, c_s are known), that is linearly independent w.r.t. all previously constructed equations by i. Hence, each user i must receive $|\mathcal{K}_i|$ innovative tokens in order to decode its packets. Note that it is quite possible, and actually very desirable, for the same packet to be a token (better yet, an innovative token) for multiple users.

In order to avoid inefficiency and, hopefully, achieve the outer bound of Section III, it is crucial that, under certain circumstances, a symbol (i.e. a linear combination of packets) that is erased by some users, but is received by at least one other user, is stored in an appropriate queue so that it can be combined in the future with other erased symbols to provide tokens for multiple users (and thus compensate for the loss). The crux of the algorithm is in the careful bookkeeping required to handle these cases.

A. Description of algorithm `CODE1`

The transmitter maintains a virtual network of queues $Q_\mathcal{S}$, indexed by the non-empty subsets $\mathcal{S}$ of $\mathcal{N}$ (see Fig. 1 for an illustration for 4 users). The queues are initialized with the stream packets as follows

$$Q_\mathcal{S} = \begin{cases} \mathcal{K}_i & \text{if } \mathcal{S} = \{i\}, \\ \varnothing & \text{otherwise.} \end{cases}$$

Additionally, with each queue $Q_{\mathcal{S}}$, indices $T^i_{\mathcal{S}}$ are maintained for all $i \in \mathcal{S}$ and are initialized as

$$T^i_{\mathcal{S}} = \begin{cases} |\mathcal{K}_i| & \text{if } \mathcal{S} = \{i\}, \\ 0 & \text{otherwise.} \end{cases}$$

It will become apparent from the algorithm's description that index $T^i_{\mathcal{S}}$ represents the number of innovative tokens (i.e. packets of the form in (8)) that user i must receive successfully from $Q_{\mathcal{S}}$ in order to decode its packets[1] (due to the performed initialization, this statement is trivially true for all $\mathcal{S}$ with $|\mathcal{S}| = 1$). These indices are dynamically updated during the algorithm's execution based on the received feedback, as will be explained soon. Finally, each receiver $i \in \mathcal{N}$ maintains its own set of queues $R^i_{\mathcal{S}}$, for all non-empty $\mathcal{S} \subseteq \mathcal{N}$ with $i \in \mathcal{S}$, where it stores the innovative tokens it receives from $Q_{\mathcal{S}}$.[2] We assume for now that all users know which queue the packet they receive comes from. All queues $R^i_{\mathcal{S}}$ are initially empty.

Denote with $\mathcal{Q}_n$ the set of all queues $Q_{\mathcal{S}}$ with $|\mathcal{S}| = n$. The algorithm operates in N phases so that in phase n, with $1 \leq n \leq N$, only transmissions of linear combinations of packets in one of the queues in $\mathcal{Q}_n$ occur. Specifically, at phase n, the transmitter orders the set $\mathcal{Q}_n$ according to a predetermined rule, known to all users (say, according to lexicographic order, which corresponds to the top-to-bottom ordering shown in Fig. 1). The transmitter then examines the first (according to this order) queue $Q_{\mathcal{S}}$ and transmits a symbol (or packet) s that is a linear combination of all packets in $Q_{\mathcal{S}}$, i.e. $s = \sum_{p \in Q_{\mathcal{S}}} a_s(p)p$. We slightly abuse parlance and say that "s is transmitted from $Q_{\mathcal{S}}$", although it is clear that s is not actually stored in $Q_{\mathcal{S}}$. The coefficients $a_s(p) \in \mathbb{F}_q$ can be produced either via a pseudo-random number generator or through structured codes. The exact generation method for $a_s(p)$ is unimportant as long as the following requirements are met:

- the generation procedure is known to all users, so that they can always reproduce the values of $a_s(p)$ even whey they don't receive the packet s. This implies that the receivers must also know the size of all queues $Q_{\mathcal{S}}$, $\mathcal{S} \subseteq \mathcal{N}$, at all times.
- the set of coefficient vectors $(a_s(p) : p \in Q_{\mathcal{S}})$, for all packets (i.e. linear combinations) s transmitted from $Q_{\mathcal{S}}$, is a linearly independent set of vectors over $\mathbb{F}_q$.

If the coefficients $a_s(p)$ are randomly generated, the second requirement need only be satisfied with probability arbitrarily close to 1 for sufficiently large field size q.

[1]it will be seen that the transmitted combination of packets from $Q_{\mathcal{S}}$ can never become a token for any user $i \in \mathcal{N} - \mathcal{S}$, so that the transmitter does not need to maintain indices for them.

[2]it will be seen in a later Section that, if instant feedback is not available to all users, the feedback information is sent to the users after all information packets have been sent. In this case, any information packets received by user i are initially placed in a single queue. Once the complete feedback is known, the packets of this queue are moved to the appropriate queues $R^i_{\mathcal{S}}$ so that the decoding procedure (i.e. the construction of the $|\mathcal{K}_i|$ linearly independent equations) can begin.

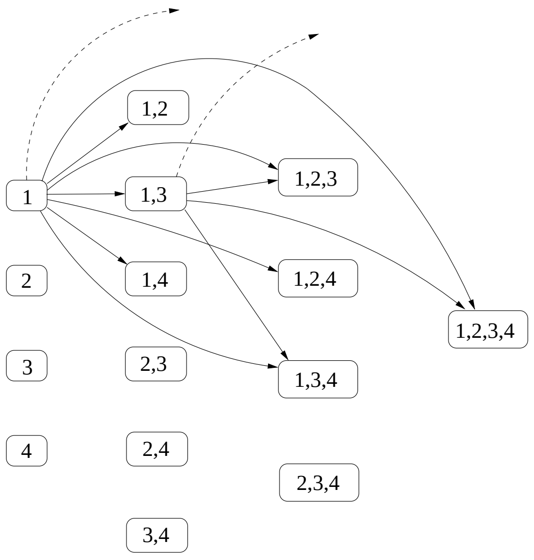

Fig. 1. Transmitter virtual queues required for 4 users and some possible index transitions.

Depending on the received feedback for the packet s transmitted from queue $Q_{\mathcal{S}}$, the following actions, collectively referred to as `ACTFB1`, are taken (all 4 cases must be examined)

1) if no user in $\mathcal{N}$ receives s, it is retransmitted.
2) for each user $i \in \mathcal{S}$ that receives s *and* satisfies $T^i_{\mathcal{S}} > 0$, s is added to queue $R^i_{\mathcal{S}}$ and $T^i_{\mathcal{S}}$ is decreased by 1.
3) if s has been erased by at least one user $i \in \mathcal{S}$ and has been received by *exactly* the users in some set $\mathcal{G}$, with $\varnothing \neq \mathcal{G} \subseteq \mathcal{N} - \mathcal{S}$, the following 2 steps are performed
   - packet s is added to queue $Q_{\mathcal{S} \cup \mathcal{G}}$ (no packets are removed from $Q_{\mathcal{S}}$).
   - for each user $i \in \mathcal{S}$ that erased s *and* satisfies $T^i_{\mathcal{S}} > 0$, $T^i_{\mathcal{S}}$ is reduced by 1 and $T^i_{\mathcal{S} \cup \mathcal{G}}$ is increased by 1.
4) if the set $\mathcal{G}$ of users that receive s is a subset of $\mathcal{S}$ such that $T^i_{\mathcal{G}} = 0$ for all $i \in \mathcal{G}$, s is retransmitted.

Fig. 1 presents the allowable index transitions from queues $Q_{\{1\}}$, $Q_{\{1,3\}}$ that occur in step 3 of `ACTFB1` (the other transitions are not shown to avoid graphical clutter; dashed lines correspond to step 2 of `ACTFB1`). Transmission of linear combinations of packets from $Q_{\mathcal{S}}$ continues for as long as there exists at least one $i \in \mathcal{S}$ with $T^i_{\mathcal{S}} > 0$. When it holds $T^i_{\mathcal{S}} = 0$ for all $i \in \mathcal{S}$, the transmitter moves to the next queue $Q_{\mathcal{S}'}$ in the ordering of $\mathcal{Q}_n$ and repeats the above procedure until it has visited all queues in $\mathcal{Q}_n$. When this occurs, phase n is complete and the algorithm moves to phase $n+1$. `CODE1` terminates at the end of phase N.

B. *Properties and correctness of `CODE1`*

The second statement in the following Lemma, which is proved rigorously in [13] although it can be intuitively ascertained through induction on $|\mathcal{S}|$, is the crucial property of `CODE1` and follows from its construction.

Lemma 6: Any packet s that is stored in queue $Q_{\mathcal{S}}$ with $|\mathcal{S}| \geq 2$ is a linear combination of all packets in queue $Q_{\mathcal{I}_s}$, for some non-empty $\mathcal{I}_s \subset \mathcal{S}$, that has been received by *exactly* all users in $\mathcal{S} - \mathcal{I}_s$. Hence, any packet in queue $Q_\mathcal{S}$ is a token for all $i \in \mathcal{S}$ (and only these $i \in \mathcal{S}$), and any linear combination of all packets in $Q_\mathcal{S}$ is an innovative token for all $i \in \mathcal{S}$ with $T^i_\mathcal{S} > 0$.

The above Lemma gives a very intuitive explanation to the algorithm's operation. Specifically, step 2 of CODE1 is equivalent to saying that whenever user i receives a useful token (meaning that $T^i_\mathcal{S} > 0$ so that there remain innovative tokens to receive) from $Q_\mathcal{S}$, this (innovative) token should be added to $R^i_\mathcal{S}$. If this is not the case and there exist users, comprising set $\mathcal{G} \subseteq \mathcal{N} - \mathcal{S}$, who receive this packet (step 3), then the packet has become a token for users in $\mathcal{S} \cup \mathcal{G}$ and should be placed in queue $Q_{\mathcal{S}\cup\mathcal{G}}$. This allows the token to be simultaneously received by multiple users in the future and thus compensate for the current loss. Additionally, since user i can now recover this token more efficiently from $Q_{\mathcal{S}\cup\mathcal{G}}$ instead of $Q_\mathcal{S}$, the indices $T^i_\mathcal{S}$, $T^i_{\mathcal{S}\cup\mathcal{G}}$ should be modified accordingly to account for the token transition. Step 4 merely states that the packet is retransmitted when it is only received by users who have already recovered all tokens intended for them.

Finally, since for any slot t that some $T^i_\mathcal{S}$ is reduced by 1, either some other $T^i_{\mathcal{S}\cup\mathcal{G}}$ is increased by 1 or (exclusive or) some packet is added to queue $R^i_\mathcal{S}$ in the same slot, it follows that the following quantity is constant during the execution of CODE1.

$$\sum_{\mathcal{S}:i\in\mathcal{S}} |R^i_\mathcal{S}(t)| + \sum_{\mathcal{S}:i\in\mathcal{S}} T^i_\mathcal{S}(t) = const = |\mathcal{K}_i|, \quad \forall\, i \in \mathcal{N}, \tag{9}$$

where the last equality follows from the initialization of CODE1. Since the algorithm terminates when it holds $T^i_\mathcal{S} = 0$ for all non-empty $\mathcal{S} \subseteq \mathcal{N}$ and all $i \in \mathcal{S}$, we conclude that at the end of the terminating slot t_f it holds $\sum_{\mathcal{S}:i\in\mathcal{S}} |R^i_\mathcal{S}(t_f)| = |\mathcal{K}_i|$ for all $i \in \mathcal{N}$. Hence, each user has recovered $|\mathcal{K}_i|$ tokens which, by choosing a sufficiently large field size q (which also implies a sufficiently large L), can be made linearly independent with probability arbitrarily close to 1. Thus, all users can decode their packets with a vanishing probability of error and CODE1 operates correctly. Notice that this result holds for arbitrary channels, so that, in principle, CODE1 is universally applicable. In addition, no prior knowledge of channel parameters is required for its execution.

C. *Performance of CODE1 for symmetric channels*

The complete analysis of the performance of CODE1 is quite lengthy with full details being given in [13]. We present here the starting point of the analysis along with the main results. We assume without loss of generality that $|\mathcal{K}_1| \geq \ldots \geq |\mathcal{K}_N|$ and $|\mathcal{K}_N|$ is sufficiently large to invoke the strong law of large numbers. We denote the events $E_\mathcal{S} \triangleq \{Z_i = 1,\ \forall\, i \in \mathcal{S}\}$ and $R_\mathcal{G} \triangleq \{Z_i = 0,\ \forall\, i \in \mathcal{G}\}$, which imply (^{c} stands for set complement and $\uplus$ for disjoint union)

$$R^c_\mathcal{G} = \biguplus_{\mathcal{H}\neq\varnothing:\mathcal{H}\subseteq\mathcal{G}} \left(E_\mathcal{H} \cap R_{\mathcal{G}-\mathcal{H}}\right). \tag{10}$$

For completeness, we define $E_\varnothing = R_\varnothing = \Omega$ (the sample space). Combining the identity $E_\mathcal{S} = (E_\mathcal{S} \cap R_\mathcal{G}) \uplus (E_\mathcal{S} \cap R^c_\mathcal{G})$ with (10) yields

$$\Pr(E_\mathcal{S}) = \Pr(E_\mathcal{S} \cap R_\mathcal{G}) + \sum_{\mathcal{H}\neq\varnothing:\mathcal{H}\subseteq\mathcal{G}} \Pr(E_{\mathcal{S}\cup\mathcal{H}} \cap R_{\mathcal{G}-\mathcal{H}}). \tag{11}$$

Noting that, for symmetric channels, all relevant probabilities depend only on the cardinality of the corresponding set, and introducing the notation $p_{e,\rho} \triangleq \Pr(E_\mathcal{S} \cap R_\mathcal{G})$ for any sets $\mathcal{S}, \mathcal{G}$ with $|\mathcal{S}| = e$, $|\mathcal{G}| = \rho$, allows us to rewrite (11) as

$$p_{e,\rho} = \epsilon_e - \sum_{l=1}^{\rho} \binom{\rho}{l} p_{e+l,\rho-l}, \tag{12}$$

where we used the fact that there are $\binom{\rho}{l}$ distinct sets $\mathcal{H} \subseteq \mathcal{G}$ with cardinality l. Denote with $k^i_\mathcal{S}$ the value of $T^i_\mathcal{S}$ at the beginning of phase $n = |\mathcal{S}|$ (i.e. before any transmissions from queues in $\mathcal{Q}_n$ take place). Due to symmetry, $k^i_\mathcal{S}$ depends only on $|\mathcal{S}|$. Hence, denoting $k^i_l = k^i_\mathcal{S}$ for any $\mathcal{S}$ with $|\mathcal{S}| = l$ and $i \in \mathcal{S}$, the construction of CODE1 implies the following recursive relation (which can be interpreted as conservation of innovative tokens) for $l \geq 2$

$$k^i_l = \sum_{m=1}^{l-1} \binom{l-1}{m-1} \frac{k^i_m}{1 - \epsilon_{N-m+1}} p_{N-l+1,l-m}, \quad \forall\, i \in \mathcal{N}, \tag{13}$$

along with the initial condition $k^i_1 = |\mathcal{K}_i|$. The number of slots T^*_n required to complete phase n, i.e. recover all innovative tokens from the queues of $\mathcal{Q}_n$, is given by

$$T^*_n = \sum_{\mathcal{S}:|\mathcal{S}|=n} \frac{1}{1 - \epsilon_{N-n+1}} \left(\max_{i\in\mathcal{S}} k^i_\mathcal{S}\right). \tag{14}$$

After some tedious algebra, which involves the explicit solution of the recursion in (13), the number of slots required for the entire execution of CODE1 is computed as

$$T^{**} = \sum_{n=1}^{N} T^*_n = \sum_{n=1}^{N} \frac{|\mathcal{K}_n|}{1 - \epsilon_n}. \tag{15}$$

Hence, each user i achieves a rate $R_i = |\mathcal{K}_i|/T^{**}$, which combined with Theorem 1 yields the following result [13].

Theorem 2: For symmetric channels, the capacity region outer bound $\mathcal{C}_{out}$ defined in Theorem 1 is given by (units are information bits per transmitted symbol)

$$\mathcal{C}_{out} = \left\{ \boldsymbol{R} \geq \boldsymbol{0} : \sum_{i\in\mathcal{N}} \frac{R_{\pi^*(i)}}{1 - \epsilon_i} \leq L \right\}, \tag{16}$$

where $\pi^*(i)$ is the order permutation, i.e. $R_{\pi^*(i)} \geq R_{\pi^*(j)}$ for $i < j$. Furthermore, $\mathcal{C}_{out}$ is achieved by CODE1.

D. *Taking the overhead into account*

The previous analysis rests on the assumption that complete feedback is available to all users. To remove this assumption (so that each user need only know its own feedback), the feedback information must be conveyed to the users by the transmitter at the expense of channel capacity (i.e. the incorporation of overhead) and increased complexity at the receivers. The following procedure is proposed, under the assumption that the users can execute the coefficient-generator algorithm and reproduce the coefficient values if needed.

A single overhead bit h is reserved in each packet of length L. This bit is 0, unless step 4 of CODE1 occurred in the transmission of the (immediately) previous packet, in which case it is set to 1. Essentially, bit h is the indicator bit of step 4 for the previously transmitted packet. The transmitter now applies CODE1 normally (taking feedback into account according to ACTFB1), and keeps a feedback log as follows:

- if the transmitted packet is erased by all users, nothing is written in the log.
- for each transmitted packet with $h = 0$ that is received by at least one user, the transmitter writes in the log an N-bit group, where group bit i is set to 1 or 0, depending on whether user i received the packet or not, respectively.
- for each transmitted packet with $h = 1$ that is received by at least one user, the transmitter creates the N-bit group as in the case $h = 0$, but writes nothing in the log *until it eventually transmits a packet with* $h = 0$. When this occurs, the N-bit group corresponding to the last transmitted packet with $h = 1$ is written in the log (after which the N-bit group corresponding to the current packet with $h = 0$ is also written in the log, due to the previous rule). This scheme is necessary in order to avoid arbitrarily large log sizes.

The receivers store all their received packets in a single queue, since they can do nothing more at this point until they know the complete feedback.

When CODE1 terminates, the transmitter transmits the entire feedback log until all users have received it. Once the users have the feedback log, they can essentially "replay" the execution of CODE1. Specifically, since the order in which the queues $Q_{\mathcal{S}} \in \mathcal{Q}_n$ are visited is known, and the user can deduce, from the feedback log, the values of $T^i_{\mathcal{S}}$ for all $i \in \mathcal{S}$, $\mathcal{S} \subseteq \mathcal{N}$ (so that the phase boundaries are distinguishable), the users always know which queue the received packet comes from. This allows them, with some extra bookkeeping [13], to create $R^i_{\mathcal{S}}$ and recover all available innovative tokens. It is easy to see that the number of packets required to transmit the feedback log to all users is at most $(2N/L)\cdot\sum_{n=1}^{N}\sum_{\mathcal{S}:|\mathcal{S}|=n}\left(\max_{i\in\mathcal{S}} k^i_{\mathcal{S}}\right)$. This number is upper bounded by $(2N^2/L)\sum_{i=1}^{N}|\mathcal{K}_i|$, and since these packets must be received by all users, the number of additional slots required for the log transmission is $\frac{2N^2}{L(1-\epsilon_1)}\sum_{i=1}^{N}|\mathcal{K}_i|$. The analogue of Theorem 2 is the following.

Theorem 3: Under the overhead scheme described above, CODE1 achieves the following rate region for symmetric channels

$$\mathcal{C} = \left\{\boldsymbol{R} \geq \boldsymbol{0} : \sum_{i\in\mathcal{N}} R_{\pi^*(i)}\left(\frac{1}{1-\epsilon_i} + \frac{2N^2}{L(1-\epsilon_1)}\right) \leq L-1\right\}, \tag{17}$$

where $\pi^*(i)$ is the order permutation. $\mathcal{C}$ approximates $\mathcal{C}_{out}$ very closely as $L \gg 2N^2/(1-\epsilon_1)$.

V. The 3-receiver case for arbitrary channels

Although CODE1 achieves the capacity outer bound of Theorem 1 for symmetric channels, for sufficiently large L, this is not always true for arbitrary channels, i.e. there exist rates $\boldsymbol{R} \in \mathcal{C}_{out}$ that are *not* achievable by CODE1. This is easily verified in the following scenario: consider the case of equal rates, i.e. $R_i = R$ for all $i \in \{1,2,3\}$ (which implies that $|\mathcal{K}_i| = K$ for all i), and assume that it holds $\varepsilon_1 = \varepsilon_2 = \varepsilon_3$ and $\varepsilon_{\{1,2\}} > \varepsilon_{\{1,3\}} > \varepsilon_{\{2,3\}}$. Considering all possible permutations on $\{1,2,3\}$ and applying Theorem 1 yields the following bound

$$\mathcal{C}_{eq,out} = \left\{R\boldsymbol{1} : R\left(\frac{1}{1-\varepsilon_1} + \frac{1}{1-\varepsilon_{\{1,2\}}} + \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right) \leq L\right\}. \tag{18}$$

The number of slots $\tilde{T}^{**}$ required for the application of CODE1 in this setting is computed in [13] as

$$\tilde{T}^{**} = K\cdot\max\left[\frac{1}{1-\varepsilon_1} + \frac{1}{1-\varepsilon_{\{1,2\}}} + \frac{1}{1-\varepsilon_{\{1,2,3\}}},\right.\\ \frac{1-\varepsilon_{\{2,3\}}}{(1-\varepsilon_2)(1-\varepsilon_{\{1,3\}})} + \frac{1}{1-\varepsilon_{\{1,2\}}} + \\ \left.\frac{1}{1-\varepsilon_{\{1,2,3\}}}, (\cdot)\right]. \tag{19}$$

The third term appearing in (19) is written as $(\cdot)$ since it does not influence the fact that the second term is strictly larger than the first (since it holds $\varepsilon_1 = \varepsilon_2$ and $1-\varepsilon_{\{2,3\}} > 1-\varepsilon_{\{1,3\}}$). This implies that the rate (in information bits per transmitted symbol) $R = KL/\tilde{T}^{**}$ achieved by CODE1 is strictly smaller than the bound in (18), which demonstrates the suboptimality of CODE1.

A more intuitive explanation for the suboptimal performance of CODE1 under asymmetric channels for the 3-receiver case can also be given by the following argument (note that, for $N = 3$, the network corresponding to Fig. 1 contains only queues for sets $\mathcal{S} \in \{\{1\},\{2\},\{3\},\{1,2\},\{1,3\},\{2,3\},\{1,2,3\}\}$). Assume that in phase 2 of CODE1, the order in which the queues are visited is $\{1,2\},\{1,3\},\{2,3\}$. When the transmitter sends linear combinations of packets from $Q_{\{1,2\}}$, it is quite possible that the indices $T^1_{\{1,2\}}$, $T^2_{\{1,2\}}$ do not become zero simultaneously. Say it happens that $T^1_{\{1,2\}} = 0$ and $T^2_{\{1,2\}} > 0$. By construction, CODE1 will continue to transmit linear combinations from $Q_{\{1,2\}}$ until $T^2_{\{1,2\}}$ also becomes 0. However, this creates a source of inefficiency, as implied by step 4.

Specifically, if a transmitted packet s is only received by 1, step 4 will force s to be retransmitted until either 2 or 3 receive it, in a sense "wasting" this slot. We claim that there exists potential for improvement at this point, by combining the packets in $Q_{\{1,2\}}$ with the packets in $Q_{\{1,2,3\}}$. A linear combination of packets in these queues creates a token for both 1 and 2. Hence, even if the packet is received only by 1, the slot is not wasted, since 1 recovers an innovative token (provided that $T^1_{\{1,2,3\}} > 0$). Unfortunately, the previous reasoning implies that the rule of always combining packets from a single queue must be discarded if the objective is to achieve capacity. For $N > 3$, it is not even clear what structure a capacity achieving algorithm should have. However, for $N = 3$, we present the following algorithm, named CODE2, which achieves capacity for arbitrary channels.

CODE2 operates in phases as follows. Phase 1 of CODE2 is identical to phase 1 of CODE1, with the transmitter acting according to the rules in ACTFB1 (note that step 4 cannot occur in this phase of CODE2). In phase 2 of CODE2, the transmitter orders the queues $Q_\mathcal{S}$ in $\mathcal{Q}_2$ according to an arbitrary rule and transmits linear combinations from $Q_\mathcal{S}$ until *at least one* user $i \in \mathcal{S}$ recovers all innovative tokens from $Q_\mathcal{S}$ (i.e. $T^i_\mathcal{S} = 0$). When this occurs, the transmitter moves to the next queue in $\mathcal{Q}_2$. Again, the rules in ACTFB1 are applied. When all queues in $\mathcal{Q}_2$ have been visited, each $Q_\mathcal{S} \in \mathcal{Q}_2$ has at most one surviving index (meaning some $i \in \mathcal{S}$ with $T^i_\mathcal{S} > 0$). For convenience, we denote this epoch with t_s and define the survival number $su(i)$ of index $i \in \{1,2,3\}$ as $su(i) \triangleq |\{\mathcal{S} : |\mathcal{S}| = 2,\ T^i_\mathcal{S}(t_s) > 0\}|$, where $T^i_\mathcal{S}(t_s)$ is the value of the index at time t_s. In words, $su(i)$ is equal to the number of queues in $\mathcal{Q}_2$ which contain unrecovered innovative tokens for user i at time t_s. By definition, it holds $0 \leq su(i) \leq 2$ for all $i \in \{1,2,3\}$. The transmitter now distinguishes cases as follows

- if it holds $su(i) = 0$ for all $i \in \{1,2,3\}$, CODE2 reverts to CODE1, starting at phase 3.
- if it holds $su(i) = 1$ for all $i \in \{1,2,3\}$, CODE2 reverts to CODE1, starting at phase 2. It can be shown [13] that, for sufficiently large $|\mathcal{K}_i|^3_{i=1}$, the probability of this event is arbitrarily small, so that the capacity region is unaffected by any actions taken henceforth.
- otherwise, there exists at least one user i^* such that $su(i^*) = 0$. In fact, simple enumeration reveals that all possible configurations for $su(i)$ fall in exactly one of the following 4 categories:
  1) there exist distinct users $i^*, j^*, k^* \in \{1,2,3\}$ such that $su(i^*) = 0$, $su(j^*) = 1$, $su(k^*) = 2$.
  2) there exist distinct users $i^*, j^*, k^* \in \{1,2,3\}$ such that $su(i^*) = 0$, $su(j^*) = su(k^*) = 1$.
  3) there exist distinct users $i^*, j^*, k^* \in \{1,2,3\}$ such that $su(i^*) = su(j^*) = 0$ and $su(k^*) = 2$.
  4) there exist distinct users $i^*, j^*, k^* \in \{1,2,3\}$ such that $su(i^*) = su(j^*) = 0$ and $su(k^*) = 1$.

  To provide some concrete examples, Fig. 2 contains 4 possible configurations (each belonging, from left to right, to one of the above categories), where circles are used to denote surviving indices. The values (i^*, j^*, k^*) for each configuration are $(3,2,1)$, $(2,1,3)$, $(3,2,1)$, $(3,2,1)$, respectively. Clearly, each category contains multiple configurations (obtainable via permutations on $\{1,2,3\}$) that satisfy the above conditions. The configurations that appear in Fig. 2 correspond to a single permutation; the other permutations are handled similarly, as described next.

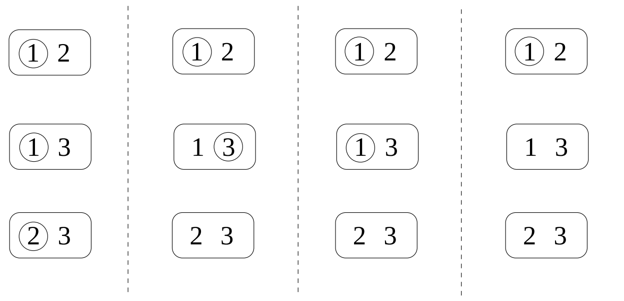

Fig. 2. Possible states of innovative token indices for the queues in $\mathcal{Q}_2$ at epoch t_s.

The transmitter now constructs the set $\mathcal{Q}_{su} = \{Q_{\{i^*,j\}} : su(i^*) = 0,\ T^j_{\{i^*,j\}} > 0\}$ consisting of all queues in $\mathcal{Q}_2$ that contain a surviving index j and an index i^* with $su(i^*) = 0$. Referring to Fig. 2, the constructed set $\mathcal{Q}_{su}$ for each category is, respectively, $\{Q_{\{2,3\}}, Q_{\{1,3\}}\}$, $\{Q_{\{1,2\}}\}$, $\{Q_{\{1,2\}}, Q_{\{1,3\}}\}$, $\{Q_{\{1,2\}}\}$. Relative order within $\mathcal{Q}_{su}$ is unimportant. A subphase, called 2.1, is now initiated, in which the following actions are performed:

- the transmitter visits each queue $Q_{\{i^*,j\}}$ in $\mathcal{Q}_{su}$ and transmits a packet s which is a linear combination of all packets in queues $Q_{\{i^*,j\}}$ *and* $Q_{\{1,2,3\}}$. Depending on the received feedback, the following actions, collectively referred to as ACTFB2, are taken
  1) if j receives s, $T^j_{\{i^*,j\}}$ is decreased by 1.
  2) if i^* receives s *and* it holds $T^{i^*}_{\{1,2,3\}} > 0$, $T^{i^*}_{\{1,2,3\}}$ is decreased by 1.
  3) if j drops s and $k \in \{1,2,3\} - \{i^*, j\}$ receives it, s is added to $Q_{\{1,2,3\}}$, $T^j_{\{i^*,j\}}$ is decreased by 1 and $T^j_{\{1,2,3\}}$ is increased by 1.
  4) if s is dropped by all users or is received only by i^* when it holds $T^{i^*}_{\{1,2,3\}} = 0$, s is retransmitted.

  Notice that ACTFB2 is similar to ACTFB1, with the addition of step 2). The above procedure is repeated until it holds $T^j_{\{i^*,j\}} = 0$, at which point the next queue in $\mathcal{Q}_{su}$ is visited. The above procedure is repeated until all queues in $\mathcal{Q}_{su}$ have been visited.
- once all queues in $\mathcal{Q}_{su}$ have been processed, the transmitter computes the new values of $su(i)$ for $i \in \{1,2,3\}$ and constructs $\mathcal{Q}_{su}$ from scratch. If $\mathcal{Q}_{su} = \varnothing$, CODE2 reverts to CODE1 starting at phase 3, otherwise it repeats the above procedure verbatim for the new $\mathcal{Q}_{su}$. It is easy to verify that at most 2 iterations of this procedure will be performed until it holds $\mathcal{Q}_{su} = \varnothing$.

As a final comment, step 4 of ACTFB2 is similar to step 4 of ACTFB1 so one could argue that CODE2 still performs inefficiently. However, by construction of $\mathcal{Q}_{su}$, it is easy to verify that if, during the combination of $Q_{\{i^*,j\}} \in \mathcal{Q}_{su}$ with $Q_{\{1,2,3\}}$, $T^{i^*}_{\{1,2,3\}}$ becomes 0 before $T^j_{\{i^*,j\}}$ does, then i^* has recovered all innovative tokens (i.e. it holds $T_{\mathcal{S}}^{i^*} = 0$ for all $\mathcal{S} \subseteq \mathcal{N}$). Hence, i^* cannot gain any more innovative tokens by combining $Q_{\{i^*,j\}}$ with $Q_{\{1,2,3\}}$ and no efficiency is lost.

To provide a concrete justification for the last statement, consider the application of subphase 2.1 to the leftmost configuration in Fig. 2. It holds $\mathcal{Q}_{su} = \{Q_{\{1,3\}}, Q_{\{2,3\}}\}$ and the transmitter starts combining $Q_{\{1,3\}}$ with $Q_{\{1,2,3\}}$ until $T_{\{2,3\}}^2$ becomes 0. If it happens that $T_{\{1,2,3\}}^3$ becomes 0 before $T_{\{2,3\}}^2$, then 3 has indeed recovered all innovative tokens so that, even if step 4 occurs, no efficiency gain is possible. The same conclusion is reached by examining the 3 other categories shown in Fig. 2. Hence, at the end of subphase 2.1, it holds $T_{\mathcal{S}}^i = 0$ for all $i \in \mathcal{S}$ with $|\mathcal{S}| = 2$ and `CODE2` reverts to `CODE1` starting at phase 3. Reference [13] contains the proof of the following important result, which ensures the correctness of `CODE2` (i.e. guarantees that each user i will receive $|\mathcal{K}_i|$ innovative packets)

Lemma 7: Assume that, at the beginning of subphase 2.1, it holds $T_{\{i,j\}}^i = 0$, $T_{\{i,j\}}^j > 0$. During subphase 2.1, any transmitted packet s that is a linear combination of all packets in queues $Q_{\{i,j\}}$, $Q_{\{1,2,3\}}$ is also an innovative token for j, i, as long as it holds $T_{\{i,j\}}^j > 0$, $T_{\{1,2,3\}}^i > 0$, respectively.

The analysis of the performance of `CODE2` is relatively straightforward (essentially being a repetition of the analysis of `CODE1`, with a careful calculation of the number of indices moved during the combination of the queues in $\mathcal{Q}_2$ with $Q_{\{1,2,3\}}$) but lengthy so we only present the final result [13].

Theorem 4: `CODE2` achieves the capacity outer bound of $\hat{C}_{out}$, assuming complete feedback is known to all users.

The assumption of complete feedback known to all users can be removed by overhead mechanisms essentially identical to the one described in Section IV-D, with a similar reduction in the achievable region. This issue will not be pursued any further.

VI. Conclusions

This paper presented 2 coding algorithms, `CODE2` and `CODE1`, that achieve the feedback capacity of N-user broadcast erasure channels with multiple unicast streams for the following cases 1) arbitrary channels, for $N \leq 3$, and 2) symmetric channels and arbitrary N, respectively. The main characteristic of the algorithms is the introduction of virtual queues to store packets, depending on received feedback, and the appropriate mixing of the packets to allow for simultaneous reception of innovative packets by multiple users, while none of them requires knowledge of channel statistics. Since only an outer bound to the capacity region is known for $N \geq 4$ and arbitrary channels, future research may involve the search for capacity achieving algorithms for $N \geq 4$. It is expected that such algorithms cannot be constructed through minor modifications of `CODE1` and may possibly require complete knowledge of channel statistics. If this is the case, adaptive algorithms that essentially "learn" the relevant statistics may be pursued. Suboptimal algorithms with guaranteed performance bounds in the spirit of [12] may also be of interest.

References

[1] T. Cover, "Broadcast channels," *IEEE Trans. Inf. Theory*, vol. 18, no. 1, pp. 2–14, January 1972.

[2] P. Bergmans, "Random coding theorem for broadcast channels with degraded components," *IEEE Trans. Inf. Theory*, vol. 19, no. 2, pp. 197–207, March 1973.

[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] R. Ahlswede, C. Ning, S. Li, and R. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, July 2000.

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

[6] 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, pp. 80–86.

[7] C. Wang, "On the capacity of wireless 1-hop intersession network coding — a broadcast packet erasure channel approach," in *Proc. International Symposium on Information Theory (ISIT)*, June 2010, pp. 1893–1897.

[8] P. Larsson and N. Johansson, "Multi-user ARQ," in *Proc. Vehicular Technology Conference*, May 2006, pp. 2052–2057.

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

[10] E. Rozner, A. Iyer, Y. Mehta, L. Qiu, and M. Jafry, "ER: efficient retransmission scheme for wireless LANs," in *Proc. ACM CoNEXT*, December 2007.

[11] 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.

[12] Y. Sagduyu and A. Ephremides, "On broadcast stability of queue-based dynamic network coding over erasure channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 12, pp. 5463–5478, December 2009.

[13] M. Gatzianas, L. Georgiadis, and L. Tassiulas, "Multiuser broadcast erasure channel — capacity and algorithms." [Online]. Available: http://users.auth.gr/~leonid/public/TechReports/tecreport_bec.pdf

[14] T. Cover and J. Thomas, *Elements of information theory*, 2nd ed. John Wiley, 2006.

[15] A. E. Gamal, "The feedback capacity of degraded broadcast channels," *IEEE Trans. Inf. Theory*, vol. 24, no. 3, pp. 379–381, May 1978.

[16] L. Ozarow and S. Leung-Yan-Cheong, "An achievable region and outer bound for the gaussian broadcast channel with feedback," *IEEE Trans. Inf. Theory*, vol. 30, no. 4, pp. 667–671, July 1984.

[17] S. Vishwanath, G. Kramer, S. Shamai, S. Jafar, and A. Goldsmith, "Capacity bounds for gausian vector broadcast channels," in *DIMACS Workshop on Signal Processing for Wireless Transmission*, October 2002, pp. 107–122.

[18] R. Liu and H. Poor, "Secrecy capacity region of a mutiple-antenna gaussian broadcast channel with conditional messages," *IEEE Trans. Inf. Theory*, vol. 55, no. 3, pp. 1235–1249, March 2009.

[19] A. Dana and B. Hassibi, "The capacity region of multiple input erasure broadcast channels," in *Proc. International Symposium on Information Theory (ISIT)*, September 2005, pp. 2315–2319.