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.

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 modifications 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.

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 the results of the latter (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 receiver 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 receivers).

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 report 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 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. For the second setting, three, essentially equivalent, algorithm variations are presented.

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 stream where the capacity is achieved for arbitrary values of erasure probability.

This document 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, named `CODE1`, 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 overhead required for sending feedback information to the receivers and the corresponding reduction in the achieveable region are also addressed. Section V contains three modifications, named `CODE2` through `CODE4`, of the previous algorithm that achieve capacity for 3 users under arbitrary channels and differ only in their implementation, while Section VI concludes the report. The proofs of all stated results are gathered in the Appendix.

II. System model and definitions

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. We denote with $\mathcal{K}_i$ the set of packets intended for receiver (i.e. user) i. The channel is modeled as 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 user erasures 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

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

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 packet 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 [13], 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 and can be written as $p(\boldsymbol{Y}|X)$. In the rest of this document, 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 $P_e = \Pr\left(\cup_{i\in\mathcal{N}}\{g_i(Y_i^n) \neq W_i\}\right)$, 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 system 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 [14] 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 [15]–[17]. We initially state a general result on the capacity of broadcast erasure channels *without feedback* [18].

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

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

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

$$\hat{Z}_{\pi(i),l} = \prod_{j=1}^{i} Z_{\pi(j),l} \quad \forall\, i \in \mathcal{N}, \tag{4}$$

while the output of user $\pi(i)$ in slot l for $\hat{C}_\pi$ is given by

$$\hat{Y}_{\pi(i),l} = \hat{Z}_{\pi(i),l} E + (1 - \hat{Z}_{\pi(i),l}) X_l. \tag{5}$$

In words, a user $\pi(i)$ erases a symbol in channel $\hat{C}_\pi$ if and only if all users $\pi(j)$, with $j \leq i$, erase the symbol in channel C. This occurs with probability $\hat{\varepsilon}_{\pi(i)} \triangleq \varepsilon_{\cup_{j=1}^{i}\{\pi(j)\}}$. Equivalently, a user $\pi(i)$ in $\hat{C}_\pi$ receives the input symbol successfully as long as at least one user $\pi(j)$, with $j \leq i$ receives it in C. In a sense, each user $\pi(i)$ sends its output $\hat{Y}_{\pi(i)}$ to user $\pi(i+1)$ (through a virtual error-free and zero-delay channel as shown in Fig. 1), which in turn sends it to the next "downstream" user.

Denote with $\tilde{Y}_{\pi(i)}$ the symbol (viewed as a random variable) that appears at the output of channel $\hat{C}_\pi$ as input to user $\pi(i)$, while $\hat{Y}_{\pi(i)}$ is the "final" symbol seen by user $\pi(i)$ and computed recursively as follows

$$\begin{aligned} \hat{Y}_{\pi(1)} &= \tilde{Y}_{\pi(1)}, \\ \hat{Y}_{\pi(i)} &= \begin{cases} E & \text{if } \tilde{Y}_{\pi(i)} = \hat{Y}_{\pi(i-1)} = E, \\ X & \text{otherwise.} \end{cases} \end{aligned} \tag{6}$$

Denoting with $p(\boldsymbol{Y}|X)$ the transition probability for channel C, it follows that $\tilde{p}(\tilde{\boldsymbol{Y}}|X)$, $p(\boldsymbol{Y}|X)$ are identical so that $\hat{C}_\pi$ can indeed be regarded as a regular broadcast erasure channel, with erasure probability $\varepsilon_{\pi(i)}$ for user $\pi(i)$,

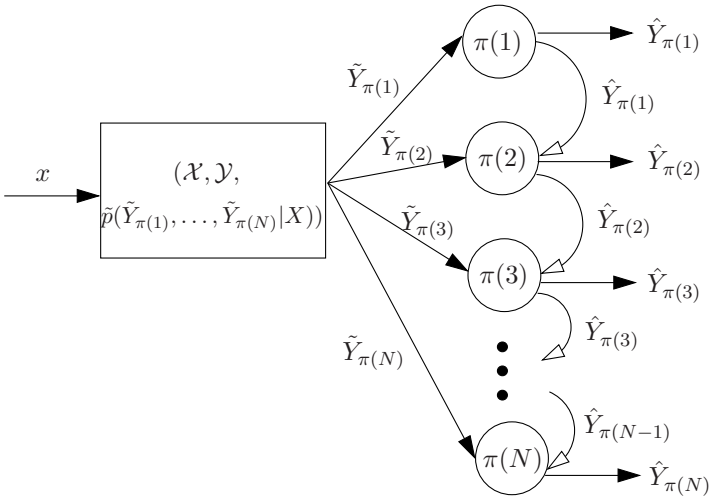

Fig. 1. Schematic of channel $\hat{C}_\pi$. White arrows represent virtual error-free channels.

with the addition of error-free channels sequentially connecting the users. The following two results are proved in Appendix A.

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

$$\hat{\mathcal{C}}_{\pi,f} = \left\{ \boldsymbol{R} \geq \mathbf{0} : \sum_{i \in \mathcal{N}} \frac{R_{\pi(i)}}{1 - \hat{\varepsilon}_{\pi(i)}} \leq L \right\}. \tag{7}$$

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

$$\mathcal{C}_f \subseteq \mathcal{C}_{out} \triangleq \cap_{\pi \in \mathcal{P}} \hat{C}_{\pi,f}, \tag{8}$$

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

IV. Coding algorithm CODE1

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 condition can be easily satisfied in practice while the second one will be removed in a later section.

Before the description of the algorithm, 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, each transmitted symbol s is an element of $\mathbb{F}_q$ and has the form

$$s = \sum_{p \in \cup_{i \in \mathcal{N}} \mathcal{K}_i} a_s(p) p, \tag{9}$$

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

$$s = \sum_{p \in \mathcal{K}_i} b_s(p)p + c_s, \tag{10}$$

where $\boldsymbol{b}_s = (b_s(p),\ p \in \mathcal{K}_i)$, 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 linear equation (with the packets in $\mathcal{K}_i$ as unknowns since 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 a 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. 2 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 (10)) that user i must successfully recover 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 and show later how this can be achieved. 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. 2). 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 $a_s(p)$ 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 conditions hold:

[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.

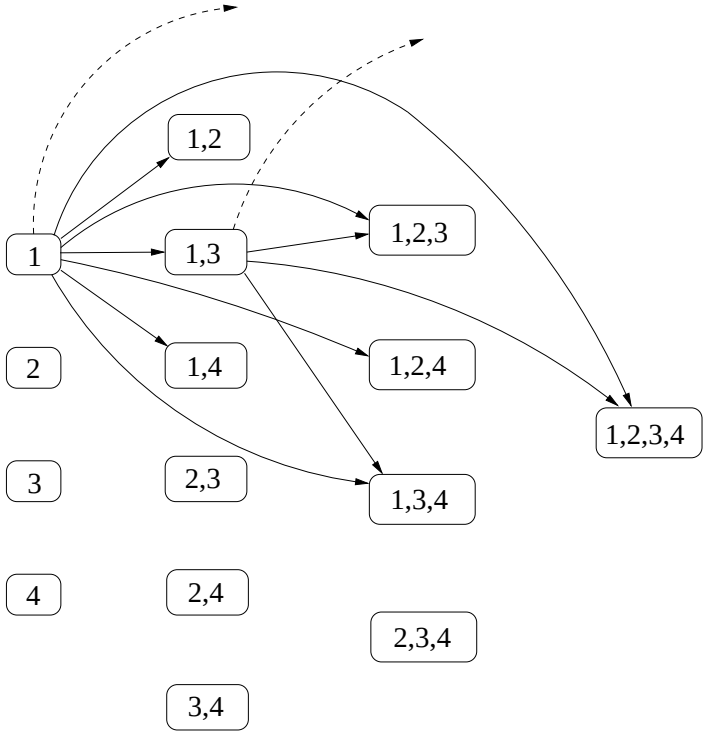

Fig. 2. Transmitter virtual queues required for 4 users and some possible index transitions.

- 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 $Q_{\mathcal{S}}$, $\mathcal{S} \subseteq \mathcal{N}$, at all times)
- the set of coefficient vectors $(a_s(p) : p \in Q_{\mathcal{S}})$, for all packets (i.e. linear combinations) s tranmitted from $Q_{\mathcal{S}}$, is a linearly independent set of vectors over $\mathbb{F}_q$.

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

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. 2 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 $n+1$. The algorithm terminates at the end of phase N.

B. Properties and correctness of CODE1

The second statement in the following Lemma, proved in Appendix B, 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 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 $T^i_{\mathcal{S}} > 0$ and there remain tokens to receive) from $Q_{\mathcal{S}}$, the (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 a 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`.

$$\sum_{\mathcal{S}:i\in\mathcal{S}} |R^i_{\mathcal{S}}(t)| + \sum_{\mathcal{S}:i\in\mathcal{S}} T^i_{\mathcal{S}}(t) = const = |\mathcal{K}_i|, \quad \forall\, i \in \mathcal{N}, \tag{11}$$

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, 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. Additionally, no knowledge of channel parameters (such as $\varepsilon_{\mathcal{I}}$, for $\mathcal{I} \subseteq \mathcal{N}$) is required for its execution.

C. Performance of `CODE1`

The analysis of the performance of `CODE1` for symmetric channels is straightforward and consists of determining the number of slots required to complete all N phases. 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 weak law of large numbers. We also 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

$$R^c_{\mathcal{G}} = \biguplus_{\mathcal{H}\neq\varnothing:\mathcal{H}\subseteq\mathcal{G}} \left(E_{\mathcal{H}} \cap R_{\mathcal{G}-\mathcal{H}}\right), \tag{12}$$

where ^{c} stands for set complement and $\uplus$ for disjoint union. 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 (12) yields

$$\Pr(E_{\mathcal{S}}) = \Pr(E_{\mathcal{S}} \cap R_{\mathcal{G}}) + \sum_{\mathcal{H}\neq\varnothing:\mathcal{H}\subseteq\mathcal{G}} \Pr(E_{\mathcal{S}\cup\mathcal{H}} \cap R_{\mathcal{G}-\mathcal{H}}). \tag{13}$$

By definition of symmetric channels, all probabilities depend only on the cardinality of the corresponding set, so that we introduce the notation $p_{e,\rho} \triangleq \Pr(E_{\mathcal{S}} \cap R_{\mathcal{G}})$ for any sets $\mathcal{S}, \mathcal{G}$ with $|\mathcal{S}| = e$, $|\mathcal{G}| = \rho$. This allows us to rewrite (13) as

$$\begin{aligned} p_{e,0} &= \epsilon_e, \\ p_{e,\rho} &= \epsilon_e - \sum_{l=1}^{\rho} \binom{\rho}{l} p_{e+l,\rho-l} \quad \forall\, \rho \geq 1, \end{aligned} \tag{14}$$

where we used the fact that there are $\binom{\rho}{l}$ distinct sets $\mathcal{H} \subseteq \mathcal{G}$ with cardinality l. The following result is easily proved by induction in Appendix C.

Lemma 7: It holds

$$p_{e,\rho} = \sum_{l=0}^{\rho} \binom{\rho}{l} \epsilon_{e+l} (-1)^l, \quad \forall\, e \geq 0,\ \forall\, \rho \geq 0. \tag{15}$$

We also need to compute the exact values of $T^i_{\mathcal{S}}$, with $i \in \mathcal{S}$, at the beginning of phase $n = |\mathcal{S}|$ (i.e. before any transmissions from queues in $\mathcal{Q}_n$ take place). We denote these values with $k^i_{\mathcal{S}}$ and exploit symmetry to note that $k^i_{\mathcal{S}}$ 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 relation

$$k^i_{\mathcal{S}} = \sum_{\substack{\mathcal{I}\neq\emptyset:\, i\in\mathcal{I},\\ \mathcal{I}\subset\mathcal{S}}} T^i_{\mathcal{I},\mathcal{S}}, \tag{16}$$

where $T^i_{\mathcal{I},\mathcal{S}}$ is the number of indices moved from $\mathcal{I}$ to $\mathcal{S}$ in step 3 (i.e. while processing $Q_{\mathcal{I}}$). This is given by

$$T^i_{\mathcal{I},\mathcal{S}} = \frac{k^i_{\mathcal{I}}}{\Pr(T^i_{\mathcal{I}} \text{ is decreased by } 1)} \Pr\left(T^i_{\mathcal{I}} \text{ is decreased by } 1,\ T^i_{\mathcal{S}} \text{ is increased by } 1\right). \tag{17}$$

By the algorithm's construction, $T^i_{\mathcal{I}}$ is not decreased for a packet transmission if the packet is erased by i *and* all users in $\mathcal{N}-\mathcal{I}$, which occurs with probability $\Pr(E_{\{i\}\uplus(\mathcal{N}-\mathcal{I})}) = \epsilon_{N-|\mathcal{I}|+1}$. Also, the rightmost probability in the RHS of (17) is equal to $\Pr(E_{\{i\}\uplus(\mathcal{N}-\mathcal{S})} \cap R_{\mathcal{S}-\mathcal{I}}) = p_{N-|\mathcal{S}|+1,|\mathcal{S}|-|\mathcal{I}|}$. Setting $|\mathcal{S}| = l$ and substituting the probability values in (16) yields

$$k^i_l = \sum_{m=1}^{l-1} \binom{l-1}{m-1} \frac{k^i_m}{1-\epsilon_{N-m+1}} p_{N-l+1,l-m}, \quad \forall\, l \geq 2, \tag{18}$$

where we summed over all sets $\mathcal{I} \subseteq \mathcal{S}$ containing i with $|\mathcal{I}| = m$ and used the fact that there exist $\binom{l-1}{m-1}$ such sets. Eq. (18) holds for all $i \in \mathcal{N}$ and is accompanied by the initial condition $k^i_1 = |\mathcal{K}_1|$. The computation of k^i_l is essential for the subsequent analysis, so we immediately present the following result.

Lemma 8: It holds for all $i \in \mathcal{N}$ and all l with $1 \leq l \leq N$,

$$\frac{k^i_l}{1-\epsilon_{N-l+1}} = |\mathcal{K}_i| \sum_{m=0}^{l-1} \binom{l-1}{m} \frac{(-1)^m}{1-\epsilon_{N-l+m+1}} = (-1)^{l+1}|\mathcal{K}_i| \sum_{m=0}^{l-1} \binom{l-1}{m} \frac{(-1)^m}{1-\epsilon_{N-m}}. \tag{19}$$

Proof: Proof appears in Appendix D. ∎

Since k^i_l depends on i only via the multiplicative term $|\mathcal{K}_i|$, we introduce the quantity $\tilde{k}_l = k^i_l/|\mathcal{K}_i|$, which is independent of i. The number of slots required by `CODE1` to process all queues $Q_{\mathcal{S}} \in \mathcal{Q}_l$ (i.e. complete phase l) is, by construction,

$$T^*_l = \sum_{\mathcal{S}:|\mathcal{S}|=l} \frac{1}{1-\epsilon_{N-l+1}} \left(\max_{i\in\mathcal{S}} k^i_{\mathcal{S}}\right). \tag{20}$$

Since $\max_{i\in\mathcal{S}} k^i_{\mathcal{S}} = k^{\min_{i\in\mathcal{S}} i}_{\mathcal{S}}$, and index r is the minimum index in $\binom{N-r}{l-1}$ sets of cardinality l, we can rewrite (20) as

$$T^*_l = \frac{\tilde{k}_l}{1-\epsilon_{N-l+1}} \sum_{r=1}^{N-l+1} \binom{N-r}{l-1} |\mathcal{K}_r| = \sum_{m=0}^{l-1} \binom{l-1}{m} \frac{(-1)^m}{1-\epsilon_{N-l+m+1}} \sum_{r=1}^{N-l+1} \binom{N-r}{l-1} |\mathcal{K}_r|, \tag{21}$$

where the last equality is due to Lemma 8. The number of slots required to complete all N phases is obviously

$$T^{**} = \sum_{l=1}^{N} T^*_l = \sum_{l=1}^{N} \sum_{r=1}^{N-l+1} \sum_{m=0}^{l-1} \binom{N-r}{l-1} \binom{l-1}{m} \frac{(-1)^m}{1-\epsilon_{N-l+m+1}} |\mathcal{K}_r|. \tag{22}$$

The last expression can be considerably simplified through the following result, proved in Appendix E.

Lemma 9: The number of slots required for the execution of `CODE1` is

$$T^{**} = \sum_{r=1}^{N} \frac{|\mathcal{K}_r|}{1-\epsilon_r}. \tag{23}$$

Hence, under `CODE1`, each user i achieves a rate $R_i = |\mathcal{K}_i|/T^{**}$, which implies (via the assumption $|\mathcal{K}_1| \geq \ldots \geq |\mathcal{K}_N|$) that $R_1 \geq \ldots \geq R_N$. Lemma 9 has the following obvious consequence.

Corollary 1: The achievable rate region of `CODE1`, measured in information bits per transmitted symbol, is a superset of the set

$$\left\{ \boldsymbol{R} \geq \boldsymbol{0} : \sum_{i=1}^{N} \frac{R_i}{1-\epsilon_i} \leq L \right\},$$

for any $\boldsymbol{R}$ such that $R_i \geq R_i$ for any $i < j$.

This leads to the first important result in this document

Theorem 2: For symmetric channels, the capacity region outer bound $\mathcal{C}_{out}$ defined in Theorem 1 is given by

$$\mathcal{C}_{out} = \left\{ \boldsymbol{R} \geq \boldsymbol{0} : \sum_{i \in \mathcal{N}} \frac{R_{\pi^*(i)}}{1-\epsilon_i} \leq L \right\}, \tag{24}$$

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`.

Proof: Proof appears in Appendix F. ■

D. Taking the overhead into account

The above analysis rests on the assumption that complete feedback is instantaneously available to all users at the end of each slot. 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 overhead scheme is proposed, under the assumption that the coefficients $a_s(p)$ of all transmitted linear combinations are known to all users, including those that *do not* receive the packet.

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 OQ, where group bit OQ_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 OQ 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 transmitter writes the OQ group that corresponds to the last transmitted packet with $h = 1$ (after that, the OQ group for the current packet with $h = 0$ is also written in the log, due to the previous rule).

In other words, the transmitter appends a single N-bit feedback entry OQ to the log whenever a packet is transmitted only once. For a packet that is retransmitted due to step 4, the transmitter writes two entries in the log, the first one containing the feedback for the first packet (the one with $h = 0$) and the second one containing the feedback for the last transmitted packet with $h = 1$. Notice that these two packets may have been transmitted in arbitrarily distant (in the temporal sense) slots, so that this scheme ensures the feedback log will not grow arbitrarily large. In fact, some thought reveals that the log will contain at most $2N \sum_{n=1}^{N} \sum_{\mathcal{S}:|\mathcal{S}|=n} \left(\max_{i \in \mathcal{S}} k_{\mathcal{S}}^i\right) \leq 2N^2 \sum_{i=1}^{N} |\mathcal{K}_i|$ entries.

During the packet transmission and log creation, the users store all their received packets in a single queue, since they can do nothing more until they know the complete feedback. However, the following simple procedure is applied: for each packet s received by user i, a single bit flag M is attached to s if packet s has $h = 1$ and there was no erasure for user i in the immediately previous slot (i.e. user i received two consecutive packets, the second one, named s, having $h = 1$). When this occurs, we say that s is "marked". 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 queues $Q_{\mathcal{S}} \in \mathcal{Q}_n$ are visited is known, and the user can deduce, from the feedback log, the values of $T_{\mathcal{S}}^i$ 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, to create $R_{\mathcal{S}}^i$ and recover all necessary innovative tokens, according to the following scheme.

Each user has local copies of $T_{\mathcal{S}}^i$ for all $i \in \mathcal{S}$ and $\mathcal{S} \subseteq \mathcal{N}$ which are initialized to 0 (except for $T_{\{i\}}^i$ which are initialized to $|\mathcal{K}_i|$).[3] User i keeps the following virtual queues, which are initially empty:

[3]for example, if $N = 3$, each user has local copies of the following indices: $T_{\{1\}}^1$, $T_{\{2\}}^2$, $T_{\{3\}}^3$, $T_{\{1,2\}}^1$, $T_{\{1,2\}}^2$, $T_{\{1,3\}}^1$, $T_{\{1,3\}}^3$, $T_{\{2,3\}}^2$, $T_{\{2,3\}}^3$, $T_{\{1,2,3\}}^1$, $T_{\{1,2,3\}}^2$, $T_{\{1,2,3\}}^3$.

- queues $R^i_{\mathcal{S}}$ for any $\mathcal{S} \subseteq \mathcal{N}$ with $i \in \mathcal{S}$. These queues are intended for storing the innovative packets of user i that are received from $Q_{\mathcal{S}}$ as well as the necessary information for decoding (i.e. the cofficients of the linear combinations).
- queues $\tilde{Q}^i_{\mathcal{S}}$ for any $\mathcal{S} \subseteq \mathcal{N}$ such that $i \notin \mathcal{S}$. These queues hold the packets that are received by i when a linear combination of packets from queue $Q_{\mathcal{S}}$ is transmitted.
- queues $M^i_{\mathcal{S},\mathcal{T}}$ for any sets $\mathcal{S}, \mathcal{T} \subseteq \mathcal{N}$ with $i \in \mathcal{S}$ and $\mathcal{S} \subset \mathcal{T}$. These queues store information about linear combinations of packets from $Q_{\mathcal{S}}$ that are added to $Q_{\mathcal{T}}$ in step 3 of `CODE1`. Hence, these packets are *not* received by i and they are received by all users in $\mathcal{T} - \mathcal{S}$. Additionally, since a linear combination of packets from $Q_{\mathcal{S}}$ is a token for all $i \in \mathcal{S}$, any packet $p \in Q_{\mathcal{S}}$ can be written as $p = c_p + \sum_{u \in \mathcal{K}_i} b_p(u) u$, where $b_p(u), c_p$ are known to i. The values $(b_p(u))_{u \in \mathcal{K}_i}$, c_p are actually stored, as a single entry, in $M^i_{\mathcal{S},\mathcal{T}}$.

In addition to the above, user i also keeps local variables $\|Q_{\mathcal{S}}\|$ (which are equal to the current size of $Q_{\mathcal{S}}$ for all $\mathcal{S} \subseteq \mathcal{N}$) and a special binary variable I_i. These variables are initialized as $\|Q_{\mathcal{S}}\| = |\mathcal{K}_j| \cdot \mathbb{I}[\mathcal{S} = \{j\}]$ and $I_i = 0$.

User i now examines the feedback log and the single queue containing all received packets (the two structures are accessed by indices $indf$, $indq$, respectively) and, potentially, moves each packet of the single queue to one of the queues $R^i_{\mathcal{S}}$, $\tilde{Q}^i_{\mathcal{S}}$, $M^i_{\mathcal{S},\mathcal{T}}$ according to the following rules, collectively referred to as `REPFB`. We assume w.l.o.g. that the current packet comes from $Q_{\mathcal{S}}$:

1) if it holds $I_i = 0$, user i examines the current OQ group pointed to by $indf$ and creates the set $\mathcal{G} = \{k \in \mathcal{N} - \mathcal{S} : OQ_k = 1\}$.
   a) if it holds $\{k \in \mathcal{N} : OQ_k = 1\} \subseteq \{k \in \mathcal{S} : T^k_{\mathcal{S}} = 0\}$, i.e. all users that received the packet have already recovered all innovative tokens from $Q_{\mathcal{S}}$, then I_i is set to 1. Also, if it holds $T^i_{\mathcal{S}} = 0$, user i peeks at the packet w immediately after the one pointed to by $indq$ (without actually changing $indq$). If the h bit of packet w is 1 and w is marked, user i advances $indq$ until it finds the last consecutively marked packet with $h = 1$. If w is not marked, $indq$ is not advanced.
   b) if $OQ_i = 1$ and $i \notin \mathcal{S}$, user i moves the packet currently pointed to by $indq$ to $\tilde{Q}^i_{\mathcal{S}}$. It also advances $indq$.
   c) for each user $j \in \mathcal{S} - \{i\}$ with $OQ_j = 1$ and $T^j_{\mathcal{S}} > 0$, $T^j_{\mathcal{S}}$ is decreased by 1. Additionally, if it holds $\mathcal{G} \neq \varnothing$, for each user $j \in \mathcal{S} - \{i\}$ with $OQ_j = 0$ and $T^j_{\mathcal{S}} > 0$, $T^j_{\mathcal{S}}$ is decreased by 1 and $T^j_{\mathcal{S}\cup\mathcal{G}}$, $\|Q_{\mathcal{S}\cup\mathcal{G}}\|$ are increased by 1 (all T indices refer to the local copies stored in user i).
   d) if $OQ_i = 1$ and $i \in \mathcal{S}$, user i examines the packet x currently pointed to by $indq$. The following actions are performed only if it holds $T^i_{\mathcal{S}} > 0$:
      - $T^i_{\mathcal{S}}$ is decreased by 1.
      - user i constructs the set $\mathfrak{I} = \{\mathcal{I} : \varnothing \neq \mathcal{I} \subset \mathcal{S}\}$, consisting of all non-empty proper subsets of $\mathcal{S}$, and orders this set according to the predefined order of processing the various queues $Q_{\mathcal{I}}$ during `CODE1`.[4]
      - user i computes the following quantities

$$\begin{aligned} b_x(u) &= \sum_{\substack{\mathcal{I}\in\mathfrak{I}:\\ i\in\mathcal{I}}} \sum_{f \in M^i_{\mathcal{I},\mathcal{S}}} a_x(f) b_f(u), \quad \forall\, u = 1, \ldots, |\mathcal{K}_i|, \\ c_x &= \sum_{\substack{\mathcal{I}\in\mathfrak{I}:\\ i\in\mathcal{I}}} \sum_{f \in M^i_{\mathcal{I},\mathcal{S}}} a_x(f) c_f + \sum_{\substack{\mathcal{I}\in\mathfrak{I}:\\ i\notin\mathcal{I}}} \sum_{f \in \tilde{Q}^i_{\mathcal{I}}} a_x(f) f, \end{aligned} \tag{25}$$

        where $a_x(f)$ is known to i by assumption and $b_f(u), c_f$ are the components of the entries in $M^i_{\mathcal{I},\mathcal{S}}$.
      - The quantities $(b_x(u))_{u \in \mathcal{K}_i}$, c_x, x are stored, as a single entry, in $R^i_{\mathcal{S}}$.

      Regardless of the value of $T^i_{\mathcal{S}}$, $indq$ is advanced.
   e) if $OQ_i = 0$, $i \in \mathcal{S}$ and $\mathcal{G} \neq \varnothing$, then the packet (call it x) referred to by OQ was erased at i. User i repeats the calculations in (25) and stores the quantities $\{b_x(u)\}_{u \in \mathcal{K}_i}$, c_x, as a single entry, in $M^i_{\mathcal{S},\mathcal{S}\cup\mathcal{G}}$.

   At this point, $indf$ is advanced.
2) if it holds $I_i = 1$, user i examines the current OQ group pointed to by $indf$ (by construction of the feedback log, the user knows that this OQ group corresponds to the last consecutively transmitted packet with $h = 1$,

[4]for example, if $N = 5$, $S = \{1, 2, 4\}$ and the queues are processed in lexicographic order, it holds $\mathfrak{I} = \{\{1\}, \{2\}, \{4\}, \{1, 2\}, \{1, 4\}, \{2, 4\}\}$, where the elements of $\mathfrak{I}$ are written in the order of their processing.

for which step 4 does not occur) and creates the set $\mathcal{G} = \{k \in \mathcal{N} - \mathcal{S} : OQ_k = 1\}$. It now performs the following:

- if $OQ_i = 0$, $i \in \mathcal{S}$, and the currently packet pointed to by $indq$ is marked (which implies that $T^i_{\mathcal{S}} = 0$ and the last transmitted packet with $h = 1$ was *erased* by i), the packet is discarded, and $indq$ is advanced.
- I_i is set to 0.
- user i now applies verbatim steps 1b)–1e) of the procedure for $I_i = 0$. Step 1a) need not be examined since it cannot occur at this point.
- $indf$ is advanced.

Note that the actions in REPFB are performed in parallel and asynchronously by all users based on their privately stored feedback log; the receivers essentially replay the transmitter's decisions in the 4 steps of ACTFB1. The validity of this overhead scheme is justified by Lemma 6. Specifically, assume that a packet x, transmitted as a linear combination of all packets in $Q_{\mathcal{S}}$, is received by $i \in \mathcal{S}$. Then it holds

$$x = \sum_{p \in Q_{\mathcal{S}}} a_x(p) p,$$

where $a_x(p)$ are known to all users. Each packet p in $Q_{\mathcal{S}}$ is, by Lemma 6, a linear combination of all packets in $Q_{\mathcal{I}_p}$ (with $i \in \mathcal{I}_p$, $\mathcal{I}_p \subseteq \mathcal{S}$) that is received by *exactly* all users in $\mathcal{S} - \mathcal{I}_p$, so that the last sum can be written as

$$x = \sum_{\substack{p \in Q_{\mathcal{S}}: \\ i \notin \mathcal{I}_p}} a_x(p) p + \sum_{\substack{p \in Q_{\mathcal{S}}: \\ i \in \mathcal{I}_p}} a_x(p) p. \tag{26}$$

By the rules in REPFB, it follows that any packet $p \in Q_{\mathcal{S}}$ that comes from $Q_{\mathcal{I}_p}$ with $i \notin \mathcal{I}_p$ is received by i, so that p is stored in queue $\tilde{Q}^i_{\mathcal{I}_p}$. For any packet $p \in Q_{\mathcal{S}}$ with $i \in \mathcal{I}_p$, it follows that the packet p was erased by i and received by all users in $\mathcal{S} - \mathcal{I}_p$. Hence, p can be written as

$$p = \sum_{u \in \mathcal{K}_i} b_p(u) u + c_p, \quad \forall\, p \in Q_{\mathcal{S}},\, i \in \mathcal{I}_p,$$

where $b_p(u)$, c_p are stored in $M^i_{\mathcal{I}_p, \mathcal{S}}$. Inserting the last equation into (26) yields

$$x = \sum_{u \in \mathcal{K}_i} \underbrace{\left(\sum_{\substack{p \in Q_{\mathcal{S}}: \\ i \in \mathcal{I}_p}} a_x(p) b_p(u) \right)}_{b_x(u)} u + \underbrace{\left[\sum_{\substack{p \in Q_{\mathcal{S}}: \\ i \notin \mathcal{I}_p}} a_x(p) p + \sum_{\substack{p \in Q_{\mathcal{S}}: \\ i \in \mathcal{I}_p}} a_x(p) c_p \right]}_{c_x} = \sum_{u \in \mathcal{K}_i} b_x(u) u + c_x, \tag{27}$$

where all parameters, except u, are known to user i. Hence, i can recover a token (i.e. form a linear equation containing $u \in \mathcal{K}_i$) by storing $b_x(u)$, c_x, x in $R^i_{\mathcal{S}}$. The reader will notice that the expressions for $b_x(u)$, c_x in (27) are identical to the ones in (25).[5]

V. The 3-receiver case

Although CODE1 achieves capacity for symmetric channels, for sufficiently large L, this is not always true for arbitrary channels. This is rigorously demonstrated in Appendix G although it can be intuitively understood for the 3-receiver case by the following argument (note that, for $N = 3$, the network corresponding to Fig. 2 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 of ACTFB1.

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

[5]the fact that the summations in (27), (25) are over different sets is natural since they correspond to different parts of the system (transmitter/receivers, respectively) which store information in different placeholders. However, the numerical results of the summations in (27), (25) are the same.

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 a class of closely related algorithms, named CODE2 through CODE4, that achieve capacity for arbitrary channels without any knowledge of channel parameters.[6] As will be seen, the main difference between the algorithms is the order in which the queues in $\mathcal{Q}_2$ are combined with $Q_{\{1,2,3\}}$. Specifically, CODE4 imposes no order on the processing of queues in $\mathcal{Q}_2$ but may require each queue to be processed more than once, whereas CODE3 and CODE2 impose a certain structure on the order with which the queues are processed. In the following sections we describe each algorithm separately, starting from CODE4.

A. Algorithm CODE4

CODE4 operates in phases as follows. Phase 1 of CODE4 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 CODE4). In phase 2 of CODE4, 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 $su(i) \triangleq |\{\mathcal{S} : |\mathcal{S}| = 2,\ T^i_{\mathcal{S}}(t_s) > 0\}|$. 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 $i \in \{1,2,3\}$, CODE4 reverts to CODE1, starting at phase 3.
- if it holds $su(i) = 1$ for all $i \in \{1,2,3\}$, CODE4 reverts to CODE1, starting at phase 2. The following crucial result, proved in Appendix H shows that this event occurs with arbitrarily small probability for sufficiently large $|\mathcal{K}_i|_{i=1}^3$, so that the capacity region is unaffected by any actions taken henceforth.
  Lemma 10: Denote $K \triangleq \min_{i\in\mathcal{N}}|\mathcal{K}_i|$. It holds $\Pr\left(su(i) = 1,\ \forall\, i \in \mathcal{N}\right) \to 0$, as $K \to \infty$.

If none of the above two occurs, we define $\tilde{\mathcal{Q}} \triangleq \{Q_{\{i,j\}} : T^i_{\{i,j\}} > 0 \vee T^j_{\{i,j\}} > 0\}$ as the set of all queues in $\mathcal{Q}_2$ with exactly one surviving index (at epoch t_s, it clearly holds $\tilde{\mathcal{Q}} \neq \varnothing$). We also introduce the auxiliary variables F_i for each user i and initialize them to 0. The following statement, which is proved in Appendix I, is important.

Lemma 11: If $\tilde{\mathcal{Q}} \neq \varnothing$, then either (or both) of the following conditions is true

1) there exists at least one i^* with $F_{i^*} = 0$ such that $T^{i^*}_{\mathcal{S}} = 0$ for all $\mathcal{S} \subseteq \mathcal{N}$ with $i^* \in \mathcal{S}$.
2) there exists at least one queue $Q_{\{i,j\}} \in \tilde{\mathcal{Q}}$ such that $T^i_{\{i,j\}} > 0$, $T^j_{\{i,j\}} = 0$, $T^j_{\{1,2,3\}} > 0$.

It should be noted that the crucial part in 1) is the specification "with $F_{i^*} = 0$", since there may exist other users j who have also recovered all tokens but for whom it holds $F_j = 1$. The transmitter now applies the following procedure.

Procedure 1: The transmitter checks Lemma 11 to determine which condition applies. Note that condition 1) takes precedence over condition 2) in the sense that, if both conditions are true, the transmitter will only take the actions dictated by condition 1) below.

- If condition 1) is true, the transmitter constructs the set $\mathcal{Q}^*_1 = \{Q_{\mathcal{S}} \in \tilde{\mathcal{Q}} : i^* \in \mathcal{S},\ i^* \text{ satisfies 1)}\}$ as the set of queues that contain a surviving index and an index corresponding to a user i^* that has recovered all tokens. Also, for all i^* that satisfy 1), F_{i^*} is set to 1. Then, for each $Q_{\mathcal{S}} \in \mathcal{Q}^*_1$ the transmitter transmits linear combinations of packets from $Q_{\mathcal{S}}$, applying the rules of ACTFB1, until it holds $T^i_{\mathcal{S}} = 0$ for all $i \in \mathcal{S}$. When this occurs, the transmitter moves to the next queue in $\mathcal{Q}^*_1$ until it has visited all queues. The relative order of the queues within $\mathcal{Q}^*_1$ is arbitrary.
- If condition 2) is true, the transmitter constructs the set $\mathcal{Q}^*_2 = \{Q_{\{i,j\}} \in \tilde{\mathcal{Q}} : Q_{\{i,j\}} \text{ satisfies 2)}\}$. It then picks the first queue in $\mathcal{Q}^*_2$ (denote it with $Q_{\{i,j\}}$; relative order within $\mathcal{Q}^*_2$ is arbitrary) and transmits a packet s that

[6] the names of the algorithms are chosen in accordance with [19].

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 ACTFB4, are taken (all 4 cases must be examined).

1) if no user in $\mathcal{N}$ receives s, it is retransmitted.
2) if user i receives s, $T^i_{\{i,j\}}$ is decreased by 1.
3) if user j receives s, $T^j_{\{1,2,3\}}$ is decreased by 1.
4) if i erases s and $k \in \{1,2,3\} - \{i,j\}$ receives it, s is added to $Q_{\{1,2,3\}}$, $T^i_{\{i,j\}}$ is decreased by 1 and $T^i_{\{1,2,3\}}$ is increased by 1.

Transmission continues until it holds $T^i_{\{i,j\}} = 0$ *or* $T^j_{\{1,2,3\}} = 0$, whichever occurs first. If $T^i_{\{i,j\}} = 0$ occurs first, the transmitter moves to the next queue in $\mathcal{Q}_2^*$, otherwise (i.e. if $T^j_{\{1,2,3\}} = 0$ occurs first) it aborts further processing of $\mathcal{Q}_2^*$.

At this point, the transmitter updates $\tilde{\mathcal{Q}}$, using the current values of $T^i_{\mathcal{S}}$ for all $\mathcal{S}$ with $|\mathcal{S}| = 2$, and checks whether $\tilde{\mathcal{Q}} = \varnothing$. If this is true (which implies that $T^i_{\mathcal{S}} = 0$ for all $\mathcal{S}$ with $|\mathcal{S}| = 2$ and $i \in \mathcal{S}$), CODE4 reverts to CODE1 starting at phase 3, otherwise it repeats Procedure 1, creating all relevant entities (such as $\mathcal{Q}_1^*$, $\mathcal{Q}_2^*$) from scratch. In this case, it is proved in Appendix I that Lemma 11 becomes a loop-invariant (i.e. a condition that is true at the beginning of each call to Procedure 1), which guarantees that eventually it will hold $\tilde{\mathcal{Q}} = \varnothing$, at which point CODE4 continues with phase 3 of CODE1. Hence, Procedure 1 is repeated for a finite number of times.

1) Correctness of CODE4: Since CODE4 is identical to CODE1 up to epoch t_s (i.e. until all queues in $\mathcal{Q}_2$ have at most one surviving index), it is clear that Lemma 6 holds for all queues $Q_{\mathcal{S}}$ up to epoch t_s. Hence, in order to show the correctness of CODE4, it suffices to prove that any packet stored in $Q_{\{1,2,3\}}$ after epoch t_s is a token for all $i \in \{1,2,3\}$ (i.e. the second statement of Lemma 6 holds for the entire operation of CODE4), at which point we can repeat the arguments of Section IV-B verbatim.

By examining the logic of Procedure 1, we conclude that, since the queues in $\mathcal{Q}_1^*$ are handled exactly as in CODE1, we need only consider the case when the transmitter sends a linear combination s of all packets in a queue $Q_{\{i,j\}} \in \mathcal{Q}_2^*$ *and* $Q_{\{1,2,3\}}$. Hence, we need to prove the following result.

Lemma 12: Denote with $\mathcal{P}_{\{1,2,3\}}(t)$ the set of packets that are stored in $Q_{\{1,2,3\}}$ at the beginning of slot t, with $t > t_s$. The following implication is true

$$(\forall\, p \in \mathcal{P}_{\{1,2,3\}}(t))(p \text{ is token for all } l \in \{1,2,3\}) \wedge (s \text{ is added to } Q_{\{1,2,3\}} \text{ in slot } t \text{ due to step 4 of ACTFB4})$$
$$\Rightarrow (\forall\, p \in \mathcal{P}_{\{1,2,3\}}(t+1))(p \text{ is token for all } l \in \{1,2,3\}).$$

Proof: Clearly, we only need to prove that s is a token for all $l \in \{1,2,3\}$. Assume w.l.o.g. that s is a linear combination of all packets stored in $Q_{\{i,j\}}$ and $Q_{\{1,2,3\}}$ at the beginning of slot t. Then, by construction of ACTFB4, we conclude that s was received by $k \in \{1,2,3\} - \{i,j\}$ so that s is a token for k. Additionally, since any packet stored in queues $Q_{\{i,j\}}$, $Q_{\{1,2,3\}}$ at the beginning of slot t is a token for both i,j (this is true by construction for the former queue and by assumption for the latter), and s has the form $s = \sum_{p \in \mathcal{P}_{\{1,2,3\}}(t)} a_s(p)p + \sum_{p \in Q_{\{i,j\}}} a_s(p)p$, it is apparent that s is also a token for both i,j. ∎

The correctness of the second statement of Lemma 6 now follows from Lemma 12 and induction on time (actually, slot index), starting from the beginning of the slot immediately after epoch t_s.

B. Algorithm CODE3

Although the order of the queues in the sets $\mathcal{Q}_1^*$, $\mathcal{Q}_2^*$ is unimportant, the presented implementation of CODE4 allows for the possibility of a queue $Q_{\mathcal{S}} \in \mathcal{Q}_2$ being processed in more than one calls to Procedure 1. A slight modification, hereafter referred to as CODE3, of CODE4 allows for each $Q_{\mathcal{S}}$ to be processed only once by imposing a specific order relation on the processing of $\mathcal{Q}_2$. The overhead scheme for CODE3 is also considerably easier to implement (in a manner similar to that described in Section IV-D) compared to CODE4, although it will be seen that both CODE4 and CODE3 achieve capacity when complete feedback is known to all users.

CODE3 is identical to CODE4 up until epoch t_s, i.e. until each queue in $\mathcal{Q}_2$ has at most one surviving index. The survival index $su(i)$ is computed for each user i at this point and the two cases $su(i) = 0 \;\; \forall\, i \in \{1,2,3\}$ and $su(i) = 1 \;\; \forall\, i \in \{1,2,3\}$ are handled exactly as in CODE4. If none of these cases occurs, we introduce a binary relation $\prec$ in $\{1,2,3\}$ as follows

$$i \prec j \Leftrightarrow (su(i) < su(j)) \vee \left(T^j_{\{i,j\}} > 0 \wedge T^i_{\{i,j\}} = 0\right) \vee (su(i) = su(j) \wedge i < j), \tag{28}$$

where $T^i_{\{i,j\}}$, $T^j_{\{i,j\}}$ are evaluated at t_s. It is easy to see by direct enumeration of all possible cases that $\prec$ has a transitive property so that it can be interpreted as an order relation. Hence, at epoch t_s, the transmitter can order the queues in $\mathcal{Q}_2$ according to $\prec$. For example, if it holds $2 \prec 1 \prec 3$, the queues can be ordered as $\{1,2\} \prec \{2,3\} \prec \{1,3\}$, while the other cases are similarly handled through a permutation on $\{1,2,3\}$.

The transmitter now initiates a subphase, called 2.1, in which the following actions are performed:

1) the transmitter visits each queue $Q_{\{i,j\}} \in \mathcal{Q}_2$ according to the $\prec$ order. If it holds $T^i_{\{i,j\}} = T^j_{\{i,j\}} = 0$ for queue $Q_{\{i,j\}}$, this queue is skipped.
2) assume w.l.o.g. that queue $Q_{\{i,j\}}$ has one surviving index and it holds $j \prec i$ (this implies that $T^i_{\{i,j\}} > 0$ and $T^j_{\{i,j\}} = 0$). The transmitter transmits a packet s which is either a linear combination of all packets in queues $Q_{\{i,j\}}$ *and* $Q_{\{1,2,3\}}$ (if it holds $T^j_{\{1,2,3\}} > 0$) or a linear combination of all packets in $Q_{\{i,j\}}$ (if it holds $T^j_{\{1,2,3\}} = 0$).[7] Depending on the received feedback, the following actions, collectively referred to as ACTFB3, are taken
   a) if i receives s, $T^i_{\{i,j\}}$ is decreased by 1.
   b) if j receives s *and* it holds $T^j_{\{1,2,3\}} > 0$, $T^j_{\{1,2,3\}}$ is decreased by 1.
   c) if i erases s and $k \in \{1,2,3\} - \{i,j\}$ receives it, s is added to $Q_{\{1,2,3\}}$, $T^i_{\{i,j\}}$ is decreased by 1 and $T^i_{\{1,2,3\}}$ is increased by 1.
   d) if s is erased by all users or is received only by j when it holds $T^j_{\{1,2,3\}} = 0$, s is retransmitted.

   Notice that ACTFB4 and ACTFB3 differ only in their conditions for retransmission (steps 1, 2d, respectively) and, since condition 2 of Lemma 11 ensures that it always holds $T^j_{\{1,2,3\}} > 0$ when ACTFB4 is applied, we conclude that the two rule sets are effectively indentical.

   The above procedure is repeated until it holds $T^i_{\{i,j\}} = 0$, at which point the next queue in $\mathcal{Q}_2$ is visited according to $\prec$. Obviously, since $T^j_{\{1,2,3\}}$ is dynamically updated depending on feedback, the decision of whether to combine packets in $Q_{\{i,j\}}$ with $Q_{\{1,2,3\}}$ is also dynamic. By construction of CODE3, the transmitter starts subphase 2.1 by combining $Q_{\{i,j\}}$ with $Q_{\{1,2,3\}}$ and switches to $Q_{\{i,j\}}$ only as soon as $T^j_{\{1,2,3\}} = 0$.

At the end of subphase 2.1, it holds $T^i_{\mathcal{S}} = 0$ for all $i \in \mathcal{S}$ with $|\mathcal{S}| = 2$. CODE3 now reverts to CODE1, starting at phase 3. The crucial characteristic of CODE3, which ensures that no throughput is lost, is the following result, which is proved by simple enumeration

Lemma 13: Because queues in $\mathcal{Q}_2$ are visited according to the $\prec$ order in subphase 2.1, user j has recovered all tokens available to it as soon as it holds $T^j_{\{1,2,3\}} = 0$

The last statement, combined with Lemma 12 (which is still true), implies the correctness of CODE3. It also guarantees that each queue in $\mathcal{Q}_2$ will only be processed once.

C. Algorithm CODE2

Algorithm CODE2 falls in between CODE4 and CODE3 in the sense that, although it does not impose a rigid order in the processing of the queues in $\mathcal{Q}_2$ (as CODE3 does), it enforces a "weak" priority on the processing of queues in $\mathcal{Q}_2$. Specifically, CODE2 is identical to CODE3 up until epoch t_s, i.e. until each queue $Q_{\mathcal{S}} \in \mathcal{Q}_2$ has at most one surviving index. The transmitter now constructs the values $su(i)$ for $i \in \{1,2,3\}$ *exactly* as in CODE3. The two cases, $su(i) = 0$ for all $i \in \{1,2,3\}$ and $su(i) = 1$ for all $i \in \{1,2,3\}$, are handled exactly as in CODE3.

If none of the above 2 cases occur, then 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 example, Fig. 3 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

[7] strictly speaking, the receivers still receive innovative tokens even if the transmitter continues combining $Q_{\{i,j\}}$ with $Q_{\{1,2,3\}}$ even after $T^j_{\{1,2,3\}}$ becomes zero. However, in terms of complexity, it is advantageous to switch to transmitting combinations of packets in $Q_{\{i,j\}}$ only after $T^j_{\{1,2,3\}}$ becomes 0 since this requires the generation of fewer coefficients $a_s(p)$.

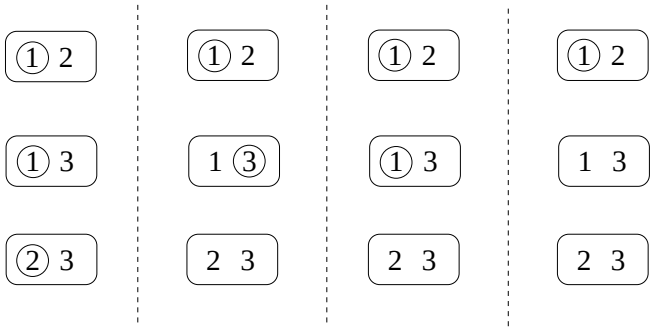

Fig. 3. Possible states of innovative token indices for the queues in $\mathcal{Q}_2$ at epoch t_s.

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. 3 correspond to a single permutation; the other permutations are handled similarly as described next.

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. 3, 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 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 that 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 received only by i^* when it holds $T^{i^*}_{\{1,2,3\}} = 0$, s is retransmitted.

  The transmitter keeps sending linear combinations from $Q_{\{i^*,j\}}$ and $Q_{\{1,2,3\}}$ until it holds $T^j_{\{i^*,j\}} = 0$, at which point the next queue in $\mathcal{Q}_{su}$ is processed and the entire procedure is repeated for the new queue.
- 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 $\mathcal{Q}_{su} = \varnothing$.

By construction of $\mathcal{Q}_{su}$, it is easy to verify (by inspecting each configuration in Fig. 3) 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 available innovative tokens (i.e. it holds $T^{i^*}_{\mathcal{S}} = 0$ for all $\mathcal{S} \subseteq \mathcal{N}$ with $i^* \in \mathcal{S}$). Hence, i^* cannot gain any more innovative tokens by combining $Q_{\{i^*,j\}}$ with $Q_{\{1,2,3\}}$ and no efficiency is lost. The last statement is essentially a repetition of Lemma 13. The correctness of CODE2 now follows from the validity of Lemmas 6, 12.

D. An example of application of CODE2, CODE3, CODE4

It is obvious from the previous section that CODE2–CODE4 have similar operation. Essentially, all 3 algorithms duplicate the operation of CODE1 until each queue in $\mathcal{Q}_2$ has at most one surviving index and then suitably combine each surviving queue in $\mathcal{Q}_2$ with $Q_{\{1,2,3\}}$ so that no slot is wasted. Since the difference in their operation lies in how this combining is performed, we present the following concrete example (as mentioned, CODE2–CODE4 are identical up to epoch t_s where each queue in $\mathcal{Q}_2$ has at most one surviving index; hence we concentrate on what

happens after t_s). Assume w.l.o.g. that the state of the indices at t_s is

$$\begin{aligned} &T^2_{\{1,2\}} > 0, \quad T^1_{\{1,2\}} = 0, \\ &T^3_{\{1,3\}} > 0, \quad T^1_{\{1,3\}} = 0, \\ &T^3_{\{2,3\}} > 0, \quad T^2_{\{2,3\}} = 0, \\ &T^i_{\{1,2,3\}} > 0, \quad \forall\, i \in \{1,2,3\}. \end{aligned} \tag{29}$$

CODE4 operates as follows. It sets $F_1 = F_2 = F_3 = 0$, determines that condition 2) of Lemma 11 is true and constructs $\mathcal{Q}_2^* = \{Q_{\{2,3\}}, Q_{\{1,2\}}, Q_{\{1,3\}}\}$. It now picks $Q_{\{2,3\}}$ and combines it with $Q_{\{1,2,3\}}$ until $T^3_{\{2,3\}}$ or $T^2_{\{1,2,3\}}$ become 0 (whichever occurs first). Say it happens that $T^2_{\{1,2,3\}} = 0$ occurs first. The transmitter aborts further processing of $\mathcal{Q}_2^*$, checks Lemma 11 again, determines that condition 2) is still true and constructs $\mathcal{Q}_2^* = \{Q_{\{1,3\}}, Q_{\{1,2\}}\}$. It now combines $Q_{\{1,3\}}$ with $Q_{\{1,2,3\}}$ until either $T^3_{\{1,3\}}$ or $T^1_{\{1,2,3\}}$ first become 0. Say it happens that $T^1_{\{1,2,3\}} = 0$ occurs first. The transmitter checks Lemma 11, determines that user 1 satisfies condition 1), sets $F_1 = 1$ and constructs $\mathcal{Q}_1^* = \{Q_{\{1,2\}}, Q_{\{1,3\}}\}$. It transmits linear combinations of packets from $Q_{\{1,2\}}$ until $T^2_{\{1,2\}} = 0$ and, when this occurs, moves to $Q_{\{1,3\}}$ and processes it until it holds $T^3_{\{1,3\}} = 0$. Notice that no efficiency is lost during this stage if any packets are only received by 1, since 1 has already recovered all tokens and can gain nothing more. The transmitter checks Lemma 11 again and determines that 2 satisfies condition 1). Hence, it sets $F_2 = 1$ and constructs $\mathcal{Q}_1^* = \{Q_{\{2,3\}}\}$. It then sends linear combinations of packets from $Q_{\{2,3\}}$ (which is the only queue in $\mathcal{Q}_2$ with remaining tokens) until $T^3_{\{2,3\}} = 0$. Again, no efficiency is lost if the packet is received only by 1 or 2, since both users have recovered their tokens. When $T^3_{\{2,3\}} = 0$ occurs, CODE4 determines that all innovative tokens have been recovered from $\mathcal{Q}_2$ and switches to CODE1, starting at phase 3.

CODE3 operates as follows. The transmitter computes the surviving indices at epoch t_s as $su(1) = 0$, $su(2) = 1$, $su(3) = 2$, so that it holds $1 \prec 2 \prec 3$ and, therefore, $\{1,2\} \prec \{1,3\} \prec \{2,3\}$. It then transmits linear combinations of packets in $Q_{\{1,2\}}$ until $T^2_{\{1,2\}} = 0$. If it happens that $T^1_{\{1,2,3\}} = 0$ while $T^2_{\{1,2\}} > 0$, the transmitter continues processing $Q_{\{1,2\}}$ since, at this point, user 1 has recovered all tokens and there is nothing more to gain. Once $T^2_{\{1,2\}} = 0$, the transmitter moves to $Q_{\{1,3\}}$ (the next queue according to the $\prec$ order) and processes it until $T^3_{\{1,3\}} = 0$. Finally, it processes $Q_{\{2,3\}}$ until $T^3_{\{2,3\}} = 0$. When this occurs, all innovative tokens have been recovered from $\mathcal{Q}_2$ so CODE3 reverts to CODE1, starting at phase 3.

CODE2 operates as follows. The transmitter computes at epoch t_s the values $su(1) = 0$, $su(2) = 1$, $su(3) = 2$ and constructs the set $\mathcal{Q}_{su} = \{Q_{\{1,2\}}, Q_{\{1,3\}}\}$. It combines each queue in $\mathcal{Q}_{su}$ with $Q_{\{1,2,3\}}$ until all T indices ($T^2_{\{1,2\}}$, $T^3_{\{1,3\}}$, respectively) in the queue of $\mathcal{Q}_{su}$ become zero. When this occurs, the values $su(i)$ are computed again as $su(1) = su(2) = 0$ and $su(3) = 1$, so that the new set $\mathcal{Q}_{su}$ is $\mathcal{Q}_{su} = \{Q_{\{2,3\}}\}$ since it holds $T^3_{\{2,3\}} > 0$. The transmitter then combines $Q_{\{2,3\}}$ with $Q_{\{1,2,3\}}$ until $T^3_{\{2,3\}}$ becomes 0. At this point, CODE2 switches to CODE1 starting at phase 3.

E. Performance of CODE2–CODE4

The performance of CODE2–CODE4 can be analyzed similarly to CODE1 by computing the number of slots required for all phases, paying special attention to the number of tokens moved between the queues when combinining queues in $\mathcal{Q}_2$ with $Q_{\{1,2,3\}}$. The procedure, described in detail in Appendix J, is straightforward but tedious so we only state the final result.

Theorem 3: CODE2–CODE4 achieve the capacity region outer bound of Theorem 1, 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 document presented coding algorithms that achieve the feedback capacity of N-user broadcast erasure channels with multiple unicast streams for the following cases

- symmetric channels, for arbitrary N.

- arbitrary channels, for $N \leq 3$.

For the second case three different implementations, `CODE2`–`CODE4`, were presented (in contrast to the first case, which is handled by a single algorithm). The main characteristic of the algorithms is the introduction of virtual queues, on the transmitter side, for storing packets depending on received feedback, and the appropriate mixing of the packets to allow for simultaneous reception of innovative tokens (i.e. linearly indepedent combinations of the unknown 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 algoriths 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.

APPENDIX
PROOFS OF VARIOUS RESULTS

A. Proof of Lemmas 3, 4

For Lemma 3, it suffices to prove the existence of a permutation $\hat{\pi}$ such that the sequence of random variables $X \to \hat{Y}_{\hat{\pi}(\pi(1))} \to \ldots \to \hat{Y}_{\hat{\pi}(\pi(N))}$ forms a Markov chain. We claim that a suitable permutation is the reverse ordering $\hat{\pi}(\pi(i)) = \pi(N-i+1)$ so that we need to prove the sequence $X \to \hat{Y}_{\pi(N)} \to \ldots \to \hat{Y}_{\pi(1)}$ to be a Markov chain. We need only prove the following equality for all $i \geq 1$

$$\Pr\left(\hat{Y}_{\pi(i)} = \hat{y}_{\pi(i)} | \hat{Y}_{\pi(i+1)} = \hat{y}_{\pi(i+1)}, \ldots, \hat{Y}_{\pi(N)} = \hat{y}_{\pi(N)}, X = x\right) = \Pr\left(\hat{Y}_{\pi(i)} = \hat{y}_{\pi(i)} | \hat{Y}_{\pi(i+1)} = \hat{y}_{\pi(i+1)}\right), \quad (30)$$

where the uppercase letters denote random variables and the corresponding lowercase letters their actual values. If (30) holds, the Markov chain property can then be proved by repeated application of (30). We start from the LHS and distinguish cases

- if there exists some $j > i$ such that $\hat{y}_{\pi(j)} = E$ (equivalently, $\hat{Z}_{\pi(j)} = 1$), it follows from (4), (5) that $\hat{Z}_{\pi(i)} = \hat{Z}_{\pi(i+1)} = 1$, which implies $\hat{Y}_{\pi(i)} = \hat{Y}_{\pi(i+1)} = E$.
- otherwise, if for all $j > i$ it holds $\hat{y}_{\pi(j)} \neq E$, it follows from (4), (5) that it must hold $\hat{y}_{\pi(i+1)} = \ldots = \hat{y}_{\pi(N)} = x$ for the conditioning of the LHS to occur on a non-null set. We denote this event as $\mathcal{E}_i \triangleq \{\hat{Y}_{\pi(i+1)} = \ldots = \hat{Y}_{\pi(N)} = X = \hat{y}_{\pi(i+1)}\}$ so that the LHS of (30) becomes $\Pr(\hat{Y}_{\pi(i)} = \hat{y}_{\pi(i)} | \mathcal{E}_i)$.

Combining the two previous cases yields

$$\text{LHS of (30)} = \begin{cases} \mathbb{I}[\hat{y}_{\pi(i)} = E] & \text{if} \quad \hat{y}_{\pi(i+1)} = E, \\ \Pr(\hat{Y}_{\pi(i)} = \hat{y}_{\pi(i)} | \mathcal{E}_i) & \text{if} \quad \hat{y}_{\pi(i+1)} \neq E. \end{cases} \quad (31)$$

For the RHS of (30), we know that $\hat{Y}_{\pi(i+1)} = E$ implies $\hat{Y}_{\pi(i)} = E$ so that it holds $\Pr(\hat{Y}_{\pi(i)} = \hat{y}_{\pi(i)} | \hat{Y}_{\pi(i+1)} = \hat{y}_{\pi(i+1)}) = \mathbb{I}[\hat{y}_{\pi(i)} = E]$ for $\hat{y}_{\pi(i+1)} = E$. For the case $\hat{y}_{\pi(i+1)} \neq E$ (equivalently $\hat{Z}_{\pi(i+1)} = 0$), we note the following set inclusion due to (4), (5)

$$\{\hat{Y}_{\pi(i+1)} = \hat{y}_{\pi(i+1)}\} \subseteq \{\hat{Y}_{\pi(i+1)} = \ldots = \hat{Y}_{\pi(N)} = X = \hat{y}_{\pi(i+1)}\} = \mathcal{E}_i, \quad (32)$$

and since $\mathcal{E}_i \subseteq \{\hat{Y}_{\pi(i+1)} = \hat{y}_{\pi(i+1)}\}$, we conclude that $\mathcal{E}_i = \{\hat{Y}_{\pi(i+1)} = \hat{y}_{\pi(i+1)}\}$ for $\hat{y}_{\pi(i+1)} \neq E$. Hence, we can write

$$\text{RHS of (30)} = \Pr(\hat{Y}_{\pi(i)} = \hat{y}_{\pi(i)} | \mathcal{E}_i) \quad \forall\, \hat{y}_{\pi(i+1)} \neq E. \quad (33)$$

Comparing the derived expression for the two sides completes the proof of Lemma 3.

Lemma 4 is easily proved by recalling that Fig. 1 implies that $\hat{C}_\pi$ can be regarded as a regular broadcast channel with erasure probability $\varepsilon_{\pi(i)}$ for user $\pi(i)$ along with error-free channels between successive users. Hence, any rate that is achieved in channel C can also be achieved in $\hat{C}_\pi$ by applying the code for channel C to channel $\hat{C}_\pi$ and forcing the users to discard any information received through the error-free channels.

B. Proof of Lemma 6

The first statement of the Lemma is true by the algorithm's construction. For the second statement, we use strong induction on $|\mathcal{S}|$. Specifically, we consider the queues in set $\mathcal{Q}_2$ and pick an arbitrary queue of that set, say $Q_{\{i,j\}}$ with $i \neq j$. By the algorithm's construction, any packet $s \in Q_{\{i,j\}}$ has exactly one of the following forms

$$s = \begin{cases} \left(\sum_{p \in \mathcal{K}_i} a_s(p) p\right)^{\{j\}}, \\ \left(\sum_{p \in \mathcal{K}_j} \tilde{a}_s(p) p\right)^{\{i\}}, \end{cases} \quad (34)$$

where the set superscript denotes the users that have succesfully received this packet. Since the coefficients $a_s(p)$, $\tilde{a}_s(p)$ are known to all users (including those that erase the packet), it is apparent that s is a token for both i, j (and for no other user), and since i, j are arbitrary, this conclusion can be extended to all queues in $\mathcal{Q}_2$. We now assume that all packets in the queues of sets $\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_n$ are tokens (for the appropriate users), and we prove that the packets in the queues of $\mathcal{Q}_{n+1}$ are also tokens.

Consider some packet $s \in Q_{\mathcal{S}}$ with $|\mathcal{S}| = n+1$. By the first statement of the Lemma there exists some non-empty $\mathcal{I}_s \subset \mathcal{S}$ (which implies $1 \leq |\mathcal{I}_s| \leq n$) such that it holds $s = \sum_{p \in Q_{\mathcal{I}_s}} a_s(p)p$, where $a_s(p)$ are known to all users. We now show that s is a token for all users in $\mathcal{S}$. Since the inductive hypothesis is true for $\mathcal{I}_s$ (meaning that s is a token for the set $\mathcal{I}_s$), we know that for any $p \in Q_{\mathcal{I}_s}$ and any $i \in \mathcal{I}_s$ it holds

$$p = \sum_{u \in \mathcal{K}_i} b_{i,p}(u)u + c_{i,p}, \tag{35}$$

where $b_{i,p}, c_{i,p}$ are known to user i. Hence, s can be written as

$$s = \sum_{p \in Q_{\mathcal{I}_s}} a_s(p) \sum_{u \in \mathcal{K}_i} (b_{i,p}(u)u + c_{i,p}) = \sum_{u \in \mathcal{K}_i} \left(\sum_{p \in Q_{\mathcal{I}_s}} a_s(p) b_{i,p}(u) \right) u + \sum_{p \in Q_{\mathcal{I}_s}} a_s(p) c_{i,p}, \tag{36}$$

so that s is a token for any $i \in \mathcal{I}_s$. Consider now some $i \in \mathcal{S} - \mathcal{I}_s$. Again, by the algorithm's construction, a linear combination of packets from $Q_{\mathcal{I}_s}$ is added to queue $Q_{\mathcal{S}}$ when exactly the users in $\mathcal{S} - \mathcal{I}_s$ receive the packet. In that case s is known to all users $i \in \mathcal{S} - \mathcal{I}_s$. Since the following relation is trivially true

$$s = \sum_{p \in \mathcal{K}_i} 0 \cdot p + (s)^{\mathcal{S} - \mathcal{I}_s}, \quad \forall\, i \in \mathcal{S} - \mathcal{I}_s, \tag{37}$$

we conclude that s is also a token for $\mathcal{S} - \mathcal{I}_s$. Hence, s is a token for any $\mathcal{S}$ with $|\mathcal{S}| = n+1$ such that $i \in \mathcal{S}$ and the proof is complete.

C. Proof of Lemma 7

Proof is by strong induction on ρ. Specifically, for $\rho = 0$, the result is true for all e. We assume that the hypothesis holds for all e and $\rho = 0, \ldots, n$ and prove that it holds for all e and $\rho = n+1$. Specifically, it holds by (14)

$$p_{e,n+1} = \epsilon_e - \sum_{l=1}^{n+1} \binom{n+1}{l} p_{e+l,n+1-l} = \epsilon_e - \sum_{l=1}^{n+1} \sum_{m=0}^{n+1-l} \binom{n+1}{l} \binom{n+1-l}{m} \epsilon_{e+l+m} (-1)^m, \tag{38}$$

where we used the inductive hypothesis to substitute for $p_{e+l,n-l+1}$ in the second equality. Setting $m' = l + m$ and renaming m' to m yields

$$p_{e,n+1} = \epsilon_e - \sum_{l=1}^{n+1} \sum_{m=l}^{n+1} \binom{n+1}{l} \binom{n+1-l}{m-l} \epsilon_{e+m} (-1)^{m+l}. \tag{39}$$

The index domain is changed to $1 \leq m \leq n+1$ and $1 \leq l \leq m$ to get

$$\begin{aligned} p_{e,n+1} &= \epsilon_e - \sum_{m=1}^{n+1} \sum_{l=1}^{m} \binom{n+1}{l} \binom{n+1-l}{m-l} \epsilon_{e+m} (-1)^{m+l} \\ &= \epsilon_e + \sum_{m=1}^{n+1} \epsilon_{e+m} (-1)^m \left[\sum_{l=1}^{m} \binom{n+1}{l} \binom{n+1-l}{m-l} (-1)^{l+1} \right]. \end{aligned} \tag{40}$$

At this point, the following identities from [20] involving binomial coefficients will be useful.

$$\binom{r}{k} = \binom{r}{r-k}, \tag{41}$$

$$\binom{r}{m} \binom{m}{k} = \binom{r}{k} \binom{r-k}{m-k}, \tag{42}$$

$$\sum_k \binom{l}{m+k} \binom{s+k}{n} (-1)^k = (-1)^{l+m} \binom{s-m}{n-l}, \quad \forall\, l \geq 0. \tag{43}$$

All quantities in (41)–(43) are assumed to be non-negative integers and the summation in (43) extends over all k such that the binomial coefficients appearing in the summand are non-zero (by definition, $\binom{n}{k} = 0$ if $k > n$).

The sum in square brackets is computed as follows (barring access to symbolic math packages such as Maple)

$$\sum_{l=1}^{m}\binom{n+1}{l}\binom{n+1-l}{m-l}(-1)^{l+1}=\sum_{l=1}^{m}\binom{n+1}{n+1-l}\binom{n+1-l}{n+1-m}(-1)^{l+1}=\sum_{l=1}^{m}\binom{n+1}{n+1-m}\binom{m}{m-l}(-1)^{l+1}$$
$$=\sum_{l=1}^{m}\binom{n+1}{m}\binom{m}{l}(-1)^{l+1}=\binom{n+1}{m}\left[\sum_{l=0}^{m}\binom{m}{l}(-1)^{l+1}+1\right]=\binom{n+1}{m}, \tag{44}$$

where we applied, in immediate succession, (41), (42), (41) and used the binomial expansion for $(1+(-1))^m$. Combining this with (40) yields

$$p_{e,n+1}=\epsilon_e+\sum_{m=1}^{n+1}\epsilon_{e+m}(-1)^m\binom{n+1}{m}=\sum_{m=0}^{n+1}\binom{n+1}{m}\epsilon_{e+m}(-1)^m, \tag{45}$$

where the last equality was achieved by inserting the first term into the sum. The last form is the exact form of the inductive hypothesis for $\rho=n+1$ and the proof is complete.

D. Proof of Lemma 8

The last equality in the RHS of (19) is easily verified by setting $m'=l-1-m$ and renaming m' back to m. We fix i and prove the result with strong induction on l. For $l=1$, (19) is easily seen to be true. We now assume that the hypothesis is true for k_1^i, k_2^i up to k_{l-1}^i (for $l\geq 2$) and we prove the hypothesis true for k_l^i. Starting from the RHS of (18) we have

$$\begin{aligned}k_l^i&=\sum_{m=1}^{l-1}\binom{l-1}{m-1}\frac{k_m^i}{1-\epsilon_{N-m+1}}p_{N-l+1,l-m}=|\mathcal{K}_i|\sum_{m=1}^{l-1}\binom{l-1}{m-1}(-1)^{m+1}\sum_{j=0}^{m-1}\binom{m-1}{j}\frac{(-1)^j}{1-\epsilon_{N-j}}p_{N-l+1,l-m}\\&=|\mathcal{K}_i|\sum_{m=1}^{l-1}\sum_{j=0}^{m-1}\binom{l-1}{m-1}\binom{m-1}{j}\frac{(-1)^{m+j+1}}{1-\epsilon_{N-j}}p_{N-l+1,l-m}\\&=|\mathcal{K}_i|\sum_{j=0}^{l-2}\binom{l-1}{j}\frac{(-1)^{j+1}}{1-\epsilon_{N-j}}\left[\sum_{m=j+1}^{l-1}\binom{l-1-j}{m-1-j}(-1)^m p_{N-l+1,l-m}\right],\end{aligned} \tag{46}$$

where we used, in succession, the inductive hypothesis for k_m^i with $m<l$, (42), and the change of indices to $0\leq j\leq l-2$, $j+1\leq m\leq l-1$.

We denote with D_j the sum inside the square brackets and concentrate on its calculation. Expanding the term $p_{N-l+1,l-m}$ according to Lemma 7 yields

$$\begin{aligned}D_j&=\sum_{m=j+1}^{l-1}\sum_{k=0}^{l-m}\binom{l-1-j}{m-1-j}\binom{l-m}{k}(-1)^{m+k}\epsilon_{N-l+1+k}\\&=\sum_{m=j+1}^{l-1}\binom{l-1-j}{m-1-j}(-1)^m\epsilon_{N-l+1}+\sum_{m=j+1}^{l-1}\sum_{k=1}^{l-m}\binom{l-1-j}{m-1-j}\binom{l-m}{k}(-1)^{m+k}\epsilon_{N-l+1+k}\\&=\epsilon_{N-l+1}\sum_{m=0}^{l-j-2}\binom{l-1-j}{m}(-1)^{m+j+1}+\sum_{m=j+1}^{l-1}\sum_{k=1}^{l-m}\binom{l-1-j}{m-1-j}\binom{l-m}{k}(-1)^{m+k}\epsilon_{N-l+1+k},\end{aligned} \tag{47}$$

where the term $k=0$ was extracted from the k summation in the transition from the first to the second line. The first sum in the third line results from the first sum of the second line after setting $m'=m-j-1$ (and renaming m' back to m). The binomial theorem yields for any j with $0\leq j\leq l-2$

$$\sum_{m=0}^{l-j-2}\binom{l-1-j}{m}(-1)^m=\sum_{m=0}^{l-j-1}\binom{l-1-j}{m}(-1)^m-\binom{l-1-j}{l-1-j}(-1)^{l-j-1}=0^{l-j-1}-(-1)^{l-j-1}=(-1)^{l+j}. \tag{48}$$

Inserting (48) back to (47) yields

$$\begin{aligned} D_j =&(-1)^{l+1}\epsilon_{N-l+1} + \sum_{m=j+1}^{l-1}\sum_{k=1}^{l-m}\binom{l-1-j}{m-1-j}\binom{l-m}{k}(-1)^{m+k}\epsilon_{N-l+1+k} \\ =&(-1)^{l+1}\epsilon_{N-l+1} + \sum_{k=1}^{l-j-1}\sum_{m=j+1}^{l-k}\binom{l-1-j}{m-1-j}\binom{l-m}{k}(-1)^{m+k}\epsilon_{N-l+1+k} \\ =&(-1)^{l+1}\epsilon_{N-l+1} + \sum_{k=1}^{l-j-1}(-1)^{k}\epsilon_{N-l+k+1}\sum_{m=0}^{l-k-1-j}\binom{l-1-j}{l-m-1-j}\binom{l-m-j-1}{k}(-1)^{m+j+1}, \end{aligned} \tag{49}$$

where we changed indices again between the first and second lines and set $m' = m - j - 1$ (and renamed m' to m) between the second and third lines. The rightmost binomial can be written as $\binom{l-m-j-1}{l-m-j-1-k}$ so that performing the substitution $m' = l - m - k - 1 - j$ (and renaming back to m) finally yields

$$D_j = (-1)^{l+1}\epsilon_{N-l+1} + \sum_{k=1}^{l-j-1}(-1)^{j+k+1}\epsilon_{N-l+1+k}\sum_{m=0}^{l-k-1-j}\binom{l-1-j}{k+m}\binom{k+m}{k}(-1)^{m}. \tag{50}$$

Applying (43) to the rightmost sum of (50) yields

$$\sum_{m=0}^{l-k-1-j}\binom{l-1-j}{k+m}\binom{k+m}{k}(-1)^{m} = (-1)^{l+1+j+k}\binom{0}{k-(l-1-j)} = \mathbb{I}[k = l-1-j], \tag{51}$$

so that the k summation in D_j essentially contains only the term $k = l - j - 1$ and D_j becomes

$$D_j = (-1)^{l+1}\epsilon_{N-l+1} + (-1)^{l}\epsilon_{N-j}. \tag{52}$$

Inserting D_j back to (46) yields

$$\begin{aligned} k_l^i =&|\mathcal{K}_i|\sum_{j=0}^{l-2}\binom{l-1}{j}\frac{(-1)^{j+1}}{1-\epsilon_{N-j}}\left[(-1)^{l+1}\epsilon_{N-l+1} + (-1)^{l}\epsilon_{N-j}\right] \\ =&|\mathcal{K}_i|\sum_{j=0}^{l-2}\binom{l-1}{j}\frac{(-1)^{j+1}}{1-\epsilon_{N-j}}\left[(-1)^{l+1}(\epsilon_{N-l+1}-1) + (-1)^{l+1}(1-\epsilon_{N-j})\right] \\ =&|\mathcal{K}_i|(1-\epsilon_{N-l+1})\sum_{j=0}^{l-2}\binom{l-1}{j}\frac{(-1)^{j+l+1}}{1-\epsilon_{N-j}} + |\mathcal{K}_i|\sum_{j=0}^{l-2}\binom{l-1}{j}(-1)^{j+l}. \end{aligned} \tag{53}$$

The right term in the last line becomes

$$|\mathcal{K}_i|\,(-1)^{l}\left[\sum_{j=0}^{l-1}\binom{l-1}{j}(-1)^{j} - (-1)^{l-1}\right] = |\mathcal{K}_i|\,(-1)^{l}(-1)^{l} = |\mathcal{K}_i|, \tag{54}$$

so that

$$\frac{k_l^i}{1-\epsilon_{N-l+1}} = |\mathcal{K}_i|\left[\sum_{j=0}^{l-1}\binom{l-1}{j}\frac{(-1)^{j+l+1}}{1-\epsilon_{N-j}} - \frac{1}{1-\epsilon_{N-l+1}}\right] + \frac{|\mathcal{K}_i|}{1-\epsilon_{N-l+1}} = |\mathcal{K}_i|\sum_{j=0}^{l-1}\binom{l-1}{j}\frac{(-1)^{j+l+1}}{1-\epsilon_{N-j}}. \tag{55}$$

The last result is the RHS of (19) so that the inductive hypothesis also holds for k_l^i and the proof is complete.

E. Proof of Lemma 9

Changing indices in (22) to $1 \le r \le N$, $1 \le l \le N - r + 1$, $0 \le m \le l - 1$ results in

$$T^{**} = \sum_{r=1}^{N}|\mathcal{K}_r|\left[\sum_{l=1}^{N-r+1}\sum_{m=0}^{l-1}\binom{N-r}{l-1}\binom{l-1}{m}\frac{(-1)^{m}}{1-\epsilon_{N-l+m+1}}\right]. \tag{56}$$

We restrict attention to the sum inside the square brackets and sum diagonally through a change of indices $d = l - m - 1$, $k = m$, which yields

$$\sum_{d=0}^{N-r}\sum_{k=0}^{N-r-d}\binom{N-r}{d+k}\binom{d+k}{k}\frac{(-1)^k}{1-\epsilon_{N-d}} = \sum_{d=0}^{N-r}\frac{1}{1-\epsilon_{N-d}}\sum_{k=0}^{N-r-d}\binom{N-r}{d+k}\binom{d+k}{d}(-1)^k, \tag{57}$$

where we also used (41) to change the last binomial coefficient. Extracting the term $d = N - r$ from the last sum yields

$$\sum_{d=0}^{N-r}\sum_{k=0}^{N-r-d}\binom{N-r}{d+k}\binom{d+k}{d}\frac{(-1)^k}{1-\epsilon_{N-d}} = \frac{1}{1-\epsilon_r} + \sum_{d=0}^{N-r-1}\frac{1}{1-\epsilon_{N-d}}\sum_{k=0}^{N-r-d}\binom{N-r}{d+k}\binom{d+k}{d}(-1)^k. \tag{58}$$

Hence, in order to prove the desired equality, it suffices to show that

$$\sum_{k=0}^{N-r-d}\binom{N-r}{d+k}\binom{d+k}{d}(-1)^k \stackrel{?}{=} 0 \quad \forall\, d = 0, \ldots, (N-r-1). \tag{59}$$

This is achieved by invoking (43) (through the obvious substitutions), which provides

$$\sum_{k=0}^{N-r-d}\binom{N-r}{d+k}\binom{d+k}{d}(-1)^k = (-1)^{N-r+d}\binom{0}{d-(N-r)} = 0, \quad \forall\, d = 0, \ldots, (N-r-1). \tag{60}$$

This completes the proof.

F. Proof of Theorem 2

By Theorem 1, the capacity region outer bound $\mathcal{C}_{out}$ is equal to

$$\mathcal{C}_{out} = \left\{\boldsymbol{R} \geq \mathbf{0} : \max_{\pi\in\mathcal{P}}\frac{R_{\pi(i)}}{1-\hat{\varepsilon}_{\pi(i)}} \leq L\right\}, \tag{61}$$

where $\mathcal{P}$ is the set of all possible permutations π on $\mathcal{N}$ and $\hat{\varepsilon}_{\pi(i)} = \varepsilon_{\cup_{j=1}^{i}\{\pi(j)\}} = \epsilon_i$, where the last equality is due to the symmetric channels property. Since it holds $\epsilon_1 \geq \ldots \geq \epsilon_N$, which in turn implies,

$$\frac{1}{1-\epsilon_1} \geq \cdots \geq \frac{1}{1-\epsilon_N},$$

it is apparent that the π^* that achieves the maximum in (61) satisfies the condition $R_{\pi^*(1)} \geq \ldots \geq R_{\pi^*(N)}$. Hence, $\mathcal{C}_{out}$ is indeed given by (24) and the achievability of $\mathcal{C}_{out}$ by `CODE1` follows immediately from Corollary 1.

G. Proof of the suboptimality of `CODE1` for 3 users

We examine the case of general channels with equal rates $R_i = R$ (which implies that $|\mathcal{K}_i| = K$ for all $i \in \{1,2,3\}$). Considering all possible permutations in the set $\{1,2,3\}$, Theorem 1 yields the following outer bound

$$\begin{aligned}\mathcal{C}_{eq,out} = \Bigg\{R\mathbf{1} : \max\Bigg[&\frac{R}{1-\varepsilon_1}+\frac{R}{1-\varepsilon_{\{1,2\}}}+\frac{R}{1-\varepsilon_{\{1,2,3\}}},\ \frac{R}{1-\varepsilon_1}+\frac{R}{1-\varepsilon_{\{1,3\}}}+\frac{R}{1-\varepsilon_{\{1,2,3\}}}\\ &\frac{R}{1-\varepsilon_2}+\frac{R}{1-\varepsilon_{\{1,2\}}}+\frac{R}{1-\varepsilon_{\{1,2,3\}}},\ \frac{R}{1-\varepsilon_2}+\frac{R}{1-\varepsilon_{\{2,3\}}}+\frac{R}{1-\varepsilon_{\{1,2,3\}}},\\ &\frac{R}{1-\varepsilon_3}+\frac{R}{1-\varepsilon_{\{1,3\}}}+\frac{R}{1-\varepsilon_{\{1,2,3\}}},\ \frac{R}{1-\varepsilon_3}+\frac{R}{1-\varepsilon_{\{2,3\}}}+\frac{R}{1-\varepsilon_{\{1,2,3\}}}\Bigg] \leq L\Bigg\}.\end{aligned} \tag{62}$$

We now assume that it holds $\varepsilon_1 = \varepsilon_2 = \varepsilon_3$ and $\varepsilon_{\{1,2\}} > \varepsilon_{\{1,3\}} > \varepsilon_{\{2,3\}}$ so that (62) is reduced to

$$\mathcal{C}_{eq,out} = \left\{R\mathbf{1} : R\left(\frac{1}{1-\varepsilon_1}+\frac{1}{1-\varepsilon_{\{1,2\}}}+\frac{1}{1-\varepsilon_{\{1,2,3\}}}\right) \leq L\right\}, \tag{63}$$

and proceed to show that, under the previous assumptions, the region in (63) is not achievable by CODE1, i.e. there exists some $R\mathbf{1} \in \mathcal{C}_{eq,out}$ that cannot be achieved by CODE1. In order to prove the last claim, we must compute the number of slots required for the operation of CODE1 under asymmetric channels.

Initially, we compute the values of $T^i_{\mathcal{S}}$ for all $\mathcal{S}$ with $|\mathcal{S}| = 2$ and $i \in \mathcal{S}$ at the beginning of phase 2 (equivalently, end of phase 1) of CODE1 (i.e. as soon as it holds $T^i_{\mathcal{S}} = 0$ for all $\mathcal{S} = \{i\}$. The number of slots NS_1 required to recover all tokens from $Q_{\{1\}}$ (i.e. reduce $T^1_{\{1\}}$ from $|\mathcal{K}_1|$ to 0) is

$$NS_1 = \frac{|\mathcal{K}_1|}{1 - \varepsilon_{\{1,2,3\}}} \quad \text{in probability as } K \to \infty, \tag{64}$$

since a token is removed from $Q_{\{1\}}$ whevenever at least one user receives the packet. Therefore, the total number of tokens moved from $Q_{\{1\}}$ to $Q_{\{1\}\cup\mathcal{S}}$ in step 3 of CODE1 is (all subsequent statements denote convergence in probability so we henceforth omit this declaration)

$$T^1_{\{1\},\{1\}\cup\mathcal{S}} = NS_1 \left(\varepsilon_1 - \varepsilon_{\{1\}\cup\mathcal{S}}\right). \tag{65}$$

Iterating the last relation over all appropriate sets $\mathcal{S}$ yields

$$\begin{aligned} T^1_{\{1\},\{1,2\}} &= |\mathcal{K}_1| \, \frac{\varepsilon_{\{1,3\}} - \varepsilon_{\{1,2,3\}}}{1 - \varepsilon_{\{1,2,3\}}}, \\ T^1_{\{1\},\{1,3\}} &= |\mathcal{K}_1| \, \frac{\varepsilon_{\{1,2\}} - \varepsilon_{\{1,2,3\}}}{1 - \varepsilon_{\{1,2,3\}}}, \\ T^1_{\{1\},\{1,2,3\}} &= |\mathcal{K}_1| \, \frac{\varepsilon_1 - \varepsilon_{\{1,2\}} - \varepsilon_{\{1,3\}} + \varepsilon_{\{1,2,3\}}}{1 - \varepsilon_{\{1,2,3\}}}. \end{aligned} \tag{66}$$

Similarly, the number of slots NS_2, NS_3 required to recover all tokens from $Q_{\{2\}}$, $Q_{\{3\}}$, respectively, in phase 1 is

$$\begin{aligned} NS_2 &= \frac{|\mathcal{K}_2|}{1 - \varepsilon_{\{1,2,3\}}}, \\ NS_3 &= \frac{|\mathcal{K}_3|}{1 - \varepsilon_{\{1,2,3\}}}, \end{aligned} \tag{67}$$

while the number of tokens moved to each queue due to step 3 is

$$\begin{aligned} T^2_{\{2\},\{1,2\}} &= |\mathcal{K}_2| \, \frac{\varepsilon_{\{2,3\}} - \varepsilon_{\{1,2,3\}}}{1 - \varepsilon_{\{1,2,3\}}}, \\ T^2_{\{2\},\{2,3\}} &= |\mathcal{K}_2| \, \frac{\varepsilon_{\{1,2\}} - \varepsilon_{\{1,2,3\}}}{1 - \varepsilon_{\{1,2,3\}}}, \\ T^2_{\{2\},\{1,2,3\}} &= |\mathcal{K}_2| \, \frac{\varepsilon_2 - \varepsilon_{\{1,2\}} - \varepsilon_{\{2,3\}} + \varepsilon_{\{1,2,3\}}}{1 - \varepsilon_{\{1,2,3\}}}, \end{aligned} \tag{68}$$

and

$$\begin{aligned} T^3_{\{3\},\{1,3\}} &= |\mathcal{K}_3| \, \frac{\varepsilon_{\{2,3\}} - \varepsilon_{\{1,2,3\}}}{1 - \varepsilon_{\{1,2,3\}}}, \\ T^3_{\{3\},\{2,3\}} &= |\mathcal{K}_3| \, \frac{\varepsilon_{\{1,3\}} - \varepsilon_{\{1,2,3\}}}{1 - \varepsilon_{\{1,2,3\}}}, \\ T^3_{\{3\},\{1,2,3\}} &= |\mathcal{K}_3| \, \frac{\varepsilon_3 - \varepsilon_{\{1,3\}} - \varepsilon_{\{2,3\}} + \varepsilon_{\{1,2,3\}}}{1 - \varepsilon_{\{1,2,3\}}}. \end{aligned} \tag{69}$$

The previous expressions give the number of tokens remaining in the queues at the beginning of phase 2 of CODE1. Specifically, if we denote with $k^i_{\mathcal{S}}$ the values of $T^i_{\mathcal{S}}$ at the beginning of phase 2 of CODE1, it holds $k^i_{\mathcal{S}} = T^i_{\{i\},\mathcal{S}}$. We now compute the number of tokens moved between the queues in $\mathcal{Q}_2$ and $Q_{\{1,2,3\}}$ during phase

2 of CODE1. Denote with $NS_{\{1,2\}}$ the number of slots required to reduce to zero all indices in $Q_{\{1,2\}}$ (i.e. $k^1_{\{1,2\}}$, $k^2_{\{1,2\}}$). Analogous interpretations are given to $NS_{\{1,3\}}$, $NS_{\{2,3\}}$. It holds

$$\begin{aligned} NS_{\{1,2\}} &= \max\left(\frac{k^1_{\{1,2\}}}{1-\varepsilon_{\{1,3\}}}, \frac{k^2_{\{1,2\}}}{1-\varepsilon_{\{2,3\}}}\right), \\ NS_{\{1,3\}} &= \max\left(\frac{k^1_{\{1,3\}}}{1-\varepsilon_{\{1,2\}}}, \frac{k^3_{\{1,3\}}}{1-\varepsilon_{\{2,3\}}}\right), \\ NS_{\{2,3\}} &= \max\left(\frac{k^2_{\{2,3\}}}{1-\varepsilon_{\{1,2\}}}, \frac{k^3_{\{2,3\}}}{1-\varepsilon_{\{1,3\}}}\right). \end{aligned} \tag{70}$$

The number of tokens moved from $Q_{\{1,2\}}$, $Q_{\{1,3\}}$, $Q_{\{2,3\}}$ to $Q_{\{1,2,3\}}$ during phase 2 is given by

$$\begin{aligned} T^1_{\{1,2\},\{1,2,3\}} &= NS_{\{1,2\}}(\varepsilon_1 - \varepsilon_{\{1,3\}}), \\ T^2_{\{1,2\},\{1,2,3\}} &= NS_{\{1,2\}}(\varepsilon_2 - \varepsilon_{\{2,3\}}), \end{aligned} \tag{71}$$

$$\begin{aligned} T^1_{\{1,3\},\{1,2,3\}} &= NS_{\{1,3\}}(\varepsilon_1 - \varepsilon_{\{1,2\}}), \\ T^3_{\{1,3\},\{1,2,3\}} &= NS_{\{1,3\}}(\varepsilon_3 - \varepsilon_{\{2,3\}}), \end{aligned} \tag{72}$$

$$\begin{aligned} T^2_{\{2,3\},\{1,2,3\}} &= NS_{\{2,3\}}(\varepsilon_2 - \varepsilon_{\{1,2\}}), \\ T^3_{\{2,3\},\{1,2,3\}} &= NS_{\{2,3\}}(\varepsilon_3 - \varepsilon_{\{1,3\}}), \end{aligned} \tag{73}$$

so that at the beginning of phase 3 of CODE1 (denote this epoch with t_3), the values of the indices are as follows

$$\begin{aligned} T^1_{\{1,2,3\}}(t_3) &= T^1_{\{1\},\{1,2,3\}} + T^1_{\{1,2\},\{1,2,3\}} + T^1_{\{1,3\},\{1,2,3\}}, \\ T^2_{\{1,2,3\}}(t_3) &= T^2_{\{2\},\{1,2,3\}} + T^2_{\{1,2\},\{1,2,3\}} + T^2_{\{2,3\},\{1,2,3\}}, \\ T^3_{\{1,2,3\}}(t_3) &= T^3_{\{3\},\{1,2,3\}} + T^3_{\{1,3\},\{1,2,3\}} + T^3_{\{2,3\},\{1,2,3\}}. \end{aligned} \tag{74}$$

Hence, the number of slots required to process queue $Q_{\{1,2,3\}}$ is

$$NS_{\{1,2,3\}} = \max\left[\frac{T^1_{\{1,2,3\}}(t_3)}{1-\varepsilon_1}, \frac{T^2_{\{1,2,3\}}(t_3)}{1-\varepsilon_2}, \frac{T^3_{\{1,2,3\}}(t_3)}{1-\varepsilon_3}\right], \tag{75}$$

and the number of slots required for the entire execution of CODE1 is

$$\tilde{T}^{**} = NS_1 + NS_2 + NS_3 + NS_{\{1,2\}} + NS_{\{1,3\}} + NS_{\{2,3\}} + NS_{\{1,2,3\}}. \tag{76}$$

Using the conditions $\varepsilon_1 = \varepsilon_2 = \varepsilon_3$, $\varepsilon_{\{1,2\}} > \varepsilon_{\{1,3\}} > \varepsilon_{\{2,3\}}$ and $|\mathcal{K}_i| = K$ provides the following significantly simplified expressions

$$\begin{aligned} NS_{\{1,2\}} &= K\left(\frac{1}{1-\varepsilon_{\{1,3\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right), \\ NS_{\{1,3\}} &= NS_{\{2,3\}} = K\left(\frac{1}{1-\varepsilon_{\{1,2\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right), \end{aligned} \tag{77}$$

$$\begin{aligned} T^1_{\{1,2,3\}}(t_3) &= K\left(\frac{\varepsilon_1 - \varepsilon_{\{1,2\}}}{1-\varepsilon_{\{1,2\}}} + \frac{\varepsilon_1 - \varepsilon_{\{1,3\}}}{1-\varepsilon_{\{1,3\}}} - \frac{\varepsilon_1 - \varepsilon_{\{1,2,3\}}}{1-\varepsilon_{\{1,2,3\}}}\right), \\ T^2_{\{1,2,3\}}(t_3) &= K\left(\frac{\varepsilon_2 - \varepsilon_{\{2,3\}}}{1-\varepsilon_{\{1,3\}}} + \frac{\varepsilon_2 - \varepsilon_{\{1,2\}}}{1-\varepsilon_{\{1,2\}}} - \frac{\varepsilon_2 - \varepsilon_{\{1,2,3\}}}{1-\varepsilon_{\{1,2,3\}}}\right), \\ T^3_{\{1,2,3\}}(t_3) &= K\left(\frac{\varepsilon_3 - \varepsilon_{\{2,3\}}}{1-\varepsilon_{\{1,2\}}} + \frac{\varepsilon_3 - \varepsilon_{\{1,3\}}}{1-\varepsilon_{\{1,2\}}} - \frac{\varepsilon_3 - \varepsilon_{\{1,2,3\}}}{1-\varepsilon_{\{1,2,3\}}}\right), \end{aligned} \tag{78}$$

$$NS_{\{1,2,3\}} = K \max\left[\frac{1}{1-\varepsilon_1} - \frac{1}{1-\varepsilon_{\{1,2\}}} - \frac{1}{1-\varepsilon_{\{1,3\}}},\ \frac{\varepsilon_2 - \varepsilon_{\{2,3\}}}{(1-\varepsilon_2)(1-\varepsilon_{\{1,3\}})},\ \frac{2\varepsilon_3 - \varepsilon_{\{1,3\}} - \varepsilon_{\{2,3\}}}{(1-\varepsilon_{\{1,2\}})(1-\varepsilon_3)}\right] + \frac{K}{1-\varepsilon_{\{1,2,3\}}}. \tag{79}$$

Therefore, the total number of slots required by `CODE1` is (after some algebra)

$$\tilde{T}^{**} = K \max\left[\frac{1}{1-\varepsilon_1} + \frac{1}{1-\varepsilon_{\{1,2\}}} + \frac{1}{1-\varepsilon_{\{1,2,3\}}},\ \frac{1-\varepsilon_{\{2,3\}}}{(1-\varepsilon_2)(1-\varepsilon_{\{1,3\}})} + \frac{1}{1-\varepsilon_{\{1,2\}}} + \frac{1}{1-\varepsilon_{\{1,2,3\}}},\ (\cdot)\right], \tag{80}$$

where the third term is denoted as $(\cdot)$ because its actual value does not affect the fact that

$$\tilde{T}^{**} > K\left(\frac{1}{1-\varepsilon_{\{1\}}} + \frac{1}{1-\varepsilon_{\{1,2\}}} + \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right), \tag{81}$$

since it holds $\varepsilon_1 = \varepsilon_2$ and $1-\varepsilon_{\{2,3\}} > 1-\varepsilon_{\{1,3\}}$. Hence, each user achieves a rate of $R = KL/\tilde{T}^{**}$ information bits per transmitted symbol, which is strictly smaller than the bound in (63). This clearly demonstrates the suboptimal performance of `CODE1` for general channels.

H. Proof of Lemma 10

Proof is by contradiction, specifically we assume that there exists some $\delta > 0$ such that $\Pr\left(su(i) = 1,\ \forall\, i \in \mathcal{N}\right) > \delta$ for all values of $K = \min_{i\in\mathcal{N}}|\mathcal{K}_i|$. This implies that at least one of the following events has probability larger than $\delta/2$ for all K

$$\begin{aligned}&\left\{T^1_{\{1,2\}}(t_s) > 0, T^2_{\{1,2\}}(t_s) = 0, T^3_{\{1,3\}}(t_s) > 0, T^1_{\{1,3\}}(t_s) = 0, T^2_{\{2,3\}}(t_s) > 0, T^3_{\{2,3\}}(t_s) = 0\right\},\\ &\left\{T^2_{\{1,2\}}(t_s) > 0, T^1_{\{1,2\}}(t_s) = 0, T^1_{\{1,3\}}(t_s) > 0, T^3_{\{1,3\}}(t_s) = 0, T^3_{\{2,3\}}(t_s) > 0, T^2_{\{2,3\}}(t_s) = 0\right\},\end{aligned} \tag{82}$$

Assume without loss of generality that the first event in (82) has probability larger than $\delta/2$ (the second event is handled similarly). To arrive at a contradiction, we use the weak law of large numbers to compute the values of all $T^i_{\mathcal{S}}(t_s)$, essentially repeating a part of the analysis performed in Appendix G.

Specifically, since phase 1 is the same for both `CODE1` and `CODE4`, (64), (66), (67), (68), (69) are still applicable. Again, we denote with $k^i_{\mathcal{S}}$ the values of $T^i_{\mathcal{S}}$ at the beginning of phase 2 of `CODE2` so that it holds $k^i_{\mathcal{S}} = T^i_{\{i\},\mathcal{S}}$ (remember that `CODE4` is identical to `CODE1` up to epoch t_s, so that (16) is still applicable). We now compute the number of tokens moved between the queues during phase 2 of `CODE4` until we reach epoch t_s. Denote with $NS_{\{1,2\}}$ the number of slots required to reduce to zero at least one of $k^1_{\{1,2\}}$, $k^2_{\{1,2\}}$ (analogous interpretations are given to $NS_{\{1,3\}}$, $NS_{\{2,3\}}$. It holds

$$NS_{\{1,2\}} = \min\left(\frac{k^1_{\{1,2\}}}{1-\varepsilon_{\{1,3\}}}, \frac{k^2_{\{1,2\}}}{1-\varepsilon_{\{2,3\}}}\right),$$

and the remaining indices at epoch t_s are

$$\begin{aligned}Rk^1_{\{1,2\}} &\triangleq T^1_{\{1,2\}}(t_s) = \max\left(k^1_{\{1,2\}} - k^2_{\{1,2\}}\frac{1-\varepsilon_{\{1,3\}}}{1-\varepsilon_{\{2,3\}}}, 0\right),\\ Rk^2_{\{1,2\}} &\triangleq T^2_{\{1,2\}}(t_s) = \max\left(k^2_{\{1,2\}} - k^1_{\{1,2\}}\frac{1-\varepsilon_{\{2,3\}}}{1-\varepsilon_{\{1,3\}}}, 0\right).\end{aligned} \tag{83}$$

Similarly, we have for the other queues $Q_{\{1,3\}}$, $Q_{\{2,3\}}$

$$\begin{aligned}NS_{\{1,3\}} &= \min\left(\frac{k^1_{\{1,3\}}}{1-\varepsilon_{\{1,2\}}}, \frac{k^3_{\{1,3\}}}{1-\varepsilon_{\{2,3\}}}\right),\\ NS_{\{2,3\}} &= \min\left(\frac{k^2_{\{2,3\}}}{1-\varepsilon_{\{1,2\}}}, \frac{k^3_{\{2,3\}}}{1-\varepsilon_{\{1,3\}}}\right),\end{aligned}$$

while the remaining indices are

$$\begin{aligned} Rk^1_{\{1,3\}} &\triangleq T^1_{\{1,3\}}(t_s) = \max\left(k^1_{\{1,3\}} - k^3_{\{1,3\}}\frac{1-\varepsilon_{\{1,2\}}}{1-\varepsilon_{\{2,3\}}}, 0\right), \\ Rk^3_{\{1,3\}} &\triangleq T^3_{\{1,3\}}(t_s) = \max\left(k^3_{\{1,3\}} - k^1_{\{1,3\}}\frac{1-\varepsilon_{\{2,3\}}}{1-\varepsilon_{\{1,2\}}}, 0\right), \end{aligned} \tag{84}$$

and

$$\begin{aligned} Rk^2_{\{2,3\}} &\triangleq T^2_{\{2,3\}}(t_s) = \max\left(k^2_{\{2,3\}} - k^3_{\{2,3\}}\frac{1-\varepsilon_{\{1,2\}}}{1-\varepsilon_{\{1,3\}}}, 0\right), \\ Rk^3_{\{2,3\}} &\triangleq T^3_{\{2,3\}}(t_s) = \max\left(k^3_{\{2,3\}} - k^2_{\{2,3\}}\frac{1-\varepsilon_{\{1,3\}}}{1-\varepsilon_{\{1,2\}}}, 0\right). \end{aligned} \tag{85}$$

We have assumed that $\Pr\left(Rk^1_{\{1,2\}} > 0,\, Rk^3_{\{1,3\}} > 0,\, Rk^2_{\{2,3\}} > 0,\, Rk^2_{\{1,2\}} = Rk^1_{\{1,3\}} = RK^3_{\{2,3\}}\right) > \delta/2$ for all K, whence it follows from (83)–(85) that, for sufficiently large K (large enough for all previous expressions to hold), there exists a sample path such that the following inequalities are simultaneously true

$$\frac{k^1_{\{1,2\}}}{1-\varepsilon_{\{1,3\}}} > \frac{k^2_{\{1,2\}}}{1-\varepsilon_{\{2,3\}}} \Rightarrow |\mathcal{K}_1|\left(\frac{1}{1-\varepsilon_{\{1,3\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right) > |\mathcal{K}_2|\left(\frac{1}{1-\varepsilon_{\{2,3\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right), \tag{86}$$

$$\frac{k^3_{\{1,3\}}}{1-\varepsilon_{\{2,3\}}} > \frac{k^1_{\{1,3\}}}{1-\varepsilon_{\{1,2\}}} \Rightarrow |\mathcal{K}_3|\left(\frac{1}{1-\varepsilon_{\{2,3\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right) > |\mathcal{K}_1|\left(\frac{1}{1-\varepsilon_{\{1,2\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right), \tag{87}$$

$$\frac{k^2_{\{2,3\}}}{1-\varepsilon_{\{1,2\}}} > \frac{k^3_{\{2,3\}}}{1-\varepsilon_{\{1,3\}}} \Rightarrow |\mathcal{K}_2|\left(\frac{1}{1-\varepsilon_{\{1,2\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right) > |\mathcal{K}_3|\left(\frac{1}{1-\varepsilon_{\{1,3\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right), \tag{88}$$

where the RHS inequalitites follow from (66)–(69). Since all RHS terms inside the parentheses are positive, we multiply the RHS of (86)–(88) by terms and arrive at the contradiction $|\mathcal{K}_1||\mathcal{K}_2||\mathcal{K}_3| > |\mathcal{K}_1||\mathcal{K}_2||\mathcal{K}_3|$, which completes the proof. As a final note, the number of tokens moved from $Q_{\{1,2\}}$, $Q_{\{1,3\}}$, $Q_{\{2,3\}}$ to $Q_{\{1,2,3\}}$ during phase 2 up until epoch t_s is still given by (71)–(73).

I. Proof of Lemma 11

We initially prove that Lemma 11 holds the first time Procedure 1 is called, i.e. when it holds $F_i = 0$ for all $i \in \mathcal{N}$. Without loss of generality, we assume $T^1_{\{1,2\}} > 0$, $T^1_{\{1,3\}} > 0$, $T^2_{\{2,3\}} > 0$ so that $\tilde{\mathcal{Q}} = \mathcal{Q}_2$ (all other cases, excluding the cases $su(i) = 0$ for all $i \in \mathcal{N}$ and $su(i) = 1$ for all $i \in \mathcal{N}$, can be similarly handled through an appropriate permutation on $\{1,2,3\}$). This is depicted in Fig. 4, where the circles indicate surviving indices in the queues of $\mathcal{Q}_2$ (the surviving indices in $Q_{\{1,2,3\}}$, if any, are not indicated). It is simple to observe that at least one of the following conditions is true

- it holds $T^3_{\{1,2,3\}} = 0$. In this case, 3 has recovered all available innovative tokens and, since $F_3 = 0$, is a valid index for condition 1) of Lemma 11.
- it holds $T^3_{\{1,2,3\}} > 0$. In this case, both queues $Q_{\{1,3\}}$, $Q_{\{2,3\}}$ satisfy condition 2) of Lemma 11.

Hence, Lemma 11 is true upon the first call of Procedure 1.

We now prove that Lemma 11 is actually a loop-invariant by examining the actions taken for each of the above cases:

- if $T^3_{\{1,2,3\}} = 0$, the transmitter constructs the set $\mathcal{Q}^*_1 = \{Q_{\{1,3\}}, Q_{\{2,3\}}\}$, sets $F_3 = 1$ (so that it still holds $F_1 = F_2 = 0$), and transmits linear combinations of packets from each individual queue in $\mathcal{Q}^*_1$ until all relevant indices $T^i_{\mathcal{S}}$ become zero. When this occurs, Procedure 1 is called again. In the new call, it holds $\tilde{\mathcal{Q}} = \{Q_{\{1,2\}}\}$ so that at least one of the following conditions is true
  1) it holds $T^2_{\{1,2,3\}} = 0$, in which case 2 has recovered all innovative tokens and, since $F_2 = 0$, satisfies condition 1) of Lemma 11.
  2) it holds $T^2_{\{1,2,3\}} > 0$, in which case $Q_{\{1,2\}}$ satisfies condition 2) of Lemma 11.

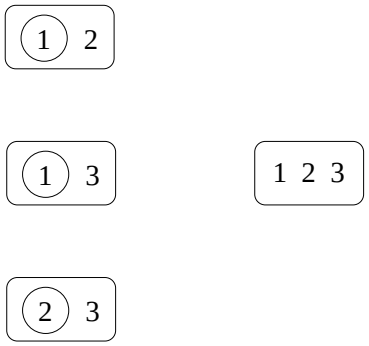

Fig. 4. Sample case for proving Lemma 11.

- if $T^3_{\{1,2,3\}} > 0$, the transmitter constructs the set $\mathcal{Q}^*_2 = \{Q_{\{1,3\}}, Q_{\{2,3\}}\}$ and combines each queue in $\mathcal{Q}^*_2$ with $Q_{\{1,2,3\}}$ until $T^1_{\{1,3\}}$, $T^2_{\{2,3\}}$ become both zero or $T^3_{\{1,2,3\}} = 0$, whichever occurs first. Hence, at least one of the following is true
  1) all indices in $\mathcal{Q}^*_2$ become zero. In the next call to Procedure 1 it holds $\tilde{\mathcal{Q}} = \{Q_{\{1,2\}}\}$ and $F_1 = F_2 = F_3 = 0$. Then, it either holds $T^2_{\{1,2,3\}} = 0$, in which case index 2 satisfies condition 1) of Lemma 11, or $T^2_{\{1,2,3\}} > 0$, in which case $Q_{\{1,2\}}$ satisfies condition 2) of Lemma 11.
  2) it holds $T^3_{\{1,2,3\}} = 0$. In the next call to Procedure 1, it still holds $F_1 = F_2 = F_3 = 0$ and $\tilde{\mathcal{Q}}$ contains at least $Q_{\{1,2\}}$. Clearly, 3 satisfies condition 1) of Lemma 11.

Since all of the above cases satisfy at least one condition of Lemma 11 at the beginning of the next call to Procedure 1, Lemma 11 is a loop-invariant. Additionally, it is easy to see by extending the previous analysis that Procedure 1 is called a finite number of times.

J. Proof of Theorem 3

We assume w.l.o.g. that the state of the indices at epoch t_s is as follows (the analysis is similar for any of the other cases)

$$\begin{aligned} &T^2_{\{1,2\}}(t_s) > 0, \quad T^1_{\{1,2\}}(t_s) = 0, \\ &T^3_{\{1,3\}}(t_s) > 0, \quad T^1_{\{1,3\}}(t_s) = 0, \\ &T^3_{\{2,3\}}(t_s) > 0, \quad T^2_{\{2,3\}}(t_s) = 0. \end{aligned} \tag{89}$$

Using the computations performed in Appendix H, the conditions in (89) imply the following inequalities

$$\begin{aligned} &\frac{k^1_{\{1,2\}}}{1-\varepsilon_{\{1,3\}}} \leq \frac{k^2_{\{1,2\}}}{1-\varepsilon_{\{2,3\}}} \Rightarrow |\mathcal{K}_1| \left(\frac{1}{1-\varepsilon_{\{1,3\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right) \leq |\mathcal{K}_2| \left(\frac{1}{1-\varepsilon_{\{2,3\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right), \\ &\frac{k^1_{\{1,3\}}}{1-\varepsilon_{\{1,2\}}} \leq \frac{k^3_{\{1,3\}}}{1-\varepsilon_{\{2,3\}}} \Rightarrow |\mathcal{K}_1| \left(\frac{1}{1-\varepsilon_{\{1,2\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right) \leq |\mathcal{K}_3| \left(\frac{1}{1-\varepsilon_{\{2,3\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right), \\ &\frac{k^2_{\{2,3\}}}{1-\varepsilon_{\{1,2\}}} \leq \frac{k^3_{\{2,3\}}}{1-\varepsilon_{\{13\}}} \Rightarrow |\mathcal{K}_2| \left(\frac{1}{1-\varepsilon_{\{1,2\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right) \leq |\mathcal{K}_3| \left(\frac{1}{1-\varepsilon_{\{1,3\}}} - \frac{1}{1-\varepsilon_{\{1,2,3\}}}\right). \end{aligned} \tag{90}$$

The values of the indices in $Q_{\{1,2,3\}}$ at epoch t_s are as follows:

$$\begin{aligned} T^1_{\{1,2,3\}}(t_s) &= k^1_{\{1,2,3\}} + k^1_{\{1,2\}} \frac{\varepsilon_1 - \varepsilon_{\{1,3\}}}{1-\varepsilon_{\{1,3\}}} + k^1_{\{1,3\}} \frac{\varepsilon_1 - \varepsilon_{\{1,2\}}}{1-\varepsilon_{\{1,2\}}}, \\ T^2_{\{1,2,3\}}(t_s) &= k^2_{\{1,2,3\}} + k^1_{\{1,2\}} \frac{\varepsilon_2 - \varepsilon_{\{2,3\}}}{1-\varepsilon_{\{1,3\}}} + k^2_{\{2,3\}} \frac{\varepsilon_2 - \varepsilon_{\{1,2\}}}{1-\varepsilon_{\{1,2\}}}, \\ T^3_{\{1,2,3\}}(t_s) &= k^3_{\{1,2,3\}} + k^1_{\{1,3\}} \frac{\varepsilon_3 - \varepsilon_{\{2,3\}}}{1-\varepsilon_{\{1,2\}}} + k^2_{\{2,3\}} \frac{\varepsilon_3 - \varepsilon_{\{1,3\}}}{1-\varepsilon_{\{1,2\}}}, \end{aligned} \tag{91}$$

while the remaining indices in $Q_{\{1,2\}}$, $Q_{\{1,3\}}$, $Q_{\{2,3\}}$ are

$$\begin{aligned} Rk^2_{\{1,2\}} &= k^2_{\{1,2\}} - k^1_{\{1,2\}} \frac{1-\varepsilon_{\{2,3\}}}{1-\varepsilon_{\{1,3\}}}, \\ Rk^3_{\{1,3\}} &= k^3_{\{1,3\}} - k^1_{\{1,3\}} \frac{1-\varepsilon_{\{2,3\}}}{1-\varepsilon_{\{1,2\}}}, \\ Rk^3_{\{2,3\}} &= k^3_{\{2,3\}} - k^2_{\{2,3\}} \frac{1-\varepsilon_{\{1,3\}}}{1-\varepsilon_{\{1,2\}}}. \end{aligned} \tag{92}$$

The number of slots expended up until epoch t_s is given up

$$NS(t_s) = \frac{|\mathcal{K}_1| + |\mathcal{K}_2| + |\mathcal{K}_3|}{1-\varepsilon_{\{1,2,3\}}} + \frac{k^1_{\{1,2\}}}{1-\varepsilon_{\{1,3\}}} + \frac{k^1_{\{1,3\}}}{1-\varepsilon_{\{1,2\}}} + \frac{k^2_{\{2,3\}}}{1-\varepsilon_{\{1,2\}}}. \tag{93}$$

Denote with $\Delta^+_{\mathcal{S},\{1,2,3\}} k^i_{\{1,2,3\}}$, $\Delta^-_{\mathcal{S},\{1,2,3\}} k^i_{\{1,2,3\}}$ the number of tokens moved into/out of, respectively, $Q_{\{1,2,3\}}$ while combining $Q_{\mathcal{S}}$ with $Q_{\{1,2,3\}}$. With this definition, we can write

$$\begin{aligned} \Delta^+_{\{1,2\},\{1,2,3\}} k^2_{\{1,2,3\}} &= Rk^2_{\{1,2\}} \frac{\varepsilon_2 - \varepsilon_{\{2,3\}}}{1-\varepsilon_{\{2,3\}}}, \\ \Delta^-_{\{1,2\},\{1,2,3\}} k^1_{\{1,2,3\}} &= Rk^2_{\{1,2\}} \frac{1-\varepsilon_1}{1-\varepsilon_{\{2,3\}}}, \end{aligned} \tag{94}$$

$$\begin{aligned} \Delta^+_{\{1,3\},\{1,2,3\}} k^3_{\{1,2,3\}} &= Rk^3_{\{1,3\}} \frac{\varepsilon_3 - \varepsilon_{\{2,3\}}}{1-\varepsilon_{\{2,3\}}}, \\ \Delta^-_{\{1,2\},\{1,2,3\}} k^1_{\{1,2,3\}} &= Rk^3_{\{1,3\}} \frac{1-\varepsilon_1}{1-\varepsilon_{\{2,3\}}}, \end{aligned} \tag{95}$$

$$\begin{aligned} \Delta^+_{\{2,3\},\{1,2,3\}} k^3_{\{1,2,3\}} &= Rk^3_{\{2,3\}} \frac{\varepsilon_3 - \varepsilon_{\{1,3\}}}{1-\varepsilon_{\{1,3\}}}, \\ \Delta^-_{\{2,3\},\{1,2,3\}} k^2_{\{1,2,3\}} &= Rk^3_{\{2,3\}} \frac{1-\varepsilon_2}{1-\varepsilon_{\{1,3\}}}. \end{aligned} \tag{96}$$

Denote with t_3 the epoch when phase 3 of `CODE4` (or `CODE3` or `CODE2`) begins. The values of the indices at t_3 are given by

$$\begin{aligned} T^1_{\{1,2,3\}}(t_3) &= \left[T^1_{\{1,2,3\}}(t_s) - \Delta^-_{\{1,2\},\{1,2,3\}} k^1_{\{1,2,3\}} - \Delta^-_{\{1,3\},\{1,2,3\}} k^1_{\{1,2,3\}} \right]^+, \\ T^2_{\{1,2,3\}}(t_3) &= \left[T^2_{\{1,2,3\}}(t_s) + \Delta^+_{\{1,2\},\{1,2,3\}} k^2_{\{1,2,3\}} - \Delta^-_{\{2,3\},\{1,2,3\}} \right]^+, \\ T^3_{\{1,2,3\}}(t_3) &= \left[T^3_{\{1,2,3\}}(t_s) + \Delta^+_{\{1,3\},\{1,2,3\}} k^3_{\{1,2,3\}} + \Delta^+_{\{2,3\},\{1,2,3\}} k^3_{\{1,2,3\}} \right]^+, \end{aligned} \tag{97}$$

where $[x]^+ \triangleq \max(x, 0)$. The number of slots contained in the interval $[t_s \;\; t_3]$ is given by

$$\frac{Rk^2_{\{1,2\}}}{1-\varepsilon_{\{2,3\}}} + \frac{Rk^3_{\{1,3\}}}{1-\varepsilon_{\{2,3\}}} + \frac{Rk^3_{\{2,3\}}}{1-\varepsilon_{\{1,3\}}}, \tag{98}$$

while the number of slots $NS_{\{1,2,3\}}$ required to complete phase 3 (i.e. recover all tokens from $Q_{\{1,2,3\}}$) is

$$NS_{\{1,2,3\}} = \max \left[\frac{T^1_{\{1,2,3\}}(t_3)}{1-\varepsilon_1}, \frac{T^2_{\{1,2,3\}}(t_3)}{1-\varepsilon_2}, \frac{T^3_{\{1,2,3\}}(t_3)}{1-\varepsilon_3} \right]. \tag{99}$$

Hence, the total number of time slots required by `CODE2`–`CODE4` is

$$NS_{total} = \text{(93)+(98)+(99)}. \tag{100}$$

After some algebra, and taking (90) into account, we find

$$NS_{total} = \max\left[\frac{|\mathcal{K}_1|}{1-\varepsilon_{\{1,2,3\}}} + \frac{|\mathcal{K}_2|}{1-\varepsilon_{\{2,3\}}} + \frac{|\mathcal{K}_3|}{1-\varepsilon_3}, \frac{|\mathcal{K}_1|}{1-\varepsilon_{\{1,2,3\}}} + \frac{|\mathcal{K}_2|}{1-\varepsilon_2} + \frac{|\mathcal{K}_3|}{1-\varepsilon_{\{2,3\}}} \right. \\ \left. \frac{|\mathcal{K}_1|}{1-\varepsilon_1} + \frac{|\mathcal{K}_2|}{1-\varepsilon_{\{1,2,3\}}} + \frac{|\mathcal{K}_3|}{1-\varepsilon_{\{1,3\}}}\right]. \tag{101}$$

The capacity region outer bound of Theorem 1 is

$$\mathcal{C}_{out} = \left\{\boldsymbol{R} \geq \mathbf{0} : \max\left[\frac{R_1}{1-\varepsilon_1} + \frac{R_2}{1-\varepsilon_{\{1,2\}}} + \frac{R_3}{1-\varepsilon_{\{1,2,3\}}}, \frac{R_1}{1-\varepsilon_1} + \frac{R_2}{1-\varepsilon_{\{1,2,3\}}} + \frac{R_3}{1-\varepsilon_{\{1,3\}}}, \right.\right. \\ \frac{R_1}{1-\varepsilon_{\{1,2\}}} + \frac{R_2}{1-\varepsilon_2} + \frac{R_3}{1-\varepsilon_{\{1,2,3\}}}, \frac{R_1}{1-\varepsilon_{\{1,2,3\}}} + \frac{R_2}{1-\varepsilon_2} + \frac{R_3}{1-\varepsilon_{\{2,3\}}}, \\ \left.\left.\frac{R_1}{1-\varepsilon_{\{1,3\}}} + \frac{R_2}{1-\varepsilon_{\{1,2,3\}}} + \frac{R_3}{1-\varepsilon_3}, \frac{R_1}{1-\varepsilon_{\{1,2,3\}}} + \frac{R_2}{1-\varepsilon_{\{2,3\}}} + \frac{R_3}{1-\varepsilon_3}\right] \leq L\right\}, \tag{102}$$

where all permutations in $\{1,2,3\}$ were considered. Due to (90), (which implies analogous inequalities for R_1, R_2, R_3 by replacing $|\mathcal{K}_i|$ with R_i in (90)), (102) is further simplified to

$$\mathcal{C}_{out} = \left\{\boldsymbol{R} \geq \mathbf{0} : \max\left[\frac{R_1}{1-\varepsilon_{\{1,2,3\}}} + \frac{R_2}{1-\varepsilon_{\{2,3\}}} + \frac{R_3}{1-\varepsilon_3}, \frac{R_1}{1-\varepsilon_{\{1,2,3\}}} + \frac{R_2}{1-\varepsilon_2} + \frac{R_3}{1-\varepsilon_{\{2,3\}}} \right.\right. \\ \left.\left.\frac{R_1}{1-\varepsilon_1} + \frac{R_2}{1-\varepsilon_{\{1,2,3\}}} + \frac{R_3}{1-\varepsilon_{\{1,3\}}}\right] \leq L\right\}. \tag{103}$$

Comparing the last expression with (100) reveals that C_{out} is achieved by `CODE2`–`CODE4`, which completes the proof.

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

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

[15] 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.

[16] 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.

[17] 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.
[18] 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.
[19] M. Gatzianas, L. Georgiadis, and L. Tassiulas, "Multiuser broadcast erasure channel with feedback — capacity and algorithms," submitted to NetCoop 2010.
[20] R. Graham, D. Knuth, and O. Patashnik, *Concrete mathematics*, 2nd ed. Addison Wesley, 1994.