Coding schemes for broadcast erasure channels with feedback: the two multicast case

Efe Onaran[*‡], Marios Gatzianas[†§] and Christina Fragouli[†]

[*] Department of Electrical and Electronics Engineering, Bilkent University, Ankara, Turkey.
[†] School of Computer and Communication Sciences, EPFL, Switzerland.

Abstract

We consider the single hop broadcast packet erasure channel (BPEC) with two multicast sessions (each of them destined to a different group of N users) and regularly available instantaneous receiver ACK/NACK feedback. Using the insight gained from recent work on BPEC with unicast and degraded messages [1], [2], we propose a virtual queue based session-mixing algorithm, which does not rely on knowledge of channel statistics and achieves capacity for $N = 2$ and iid erasures. Since the optimal extension of this algorithm to $N > 2$ is not straightforward, we then describe a low complexity algorithm which outperforms standard timesharing for arbitrary N and is actually asymptotically better than timesharing, for any finite N, as the erasure probability goes to zero. We finally provide, through an information-theoretic analysis, sufficient but not necessary asymptotic conditions between N and n (the number of transmissions) for which the achieved sum rate, under *any* coding scheme, is essentially identical to that of timesharing.

I. Introduction

This document examines a scenario where a source must transmit 2 distinct multicast messages to 2 groups (of N users each), such that all users in each group decode the corresponding multicast message. We consider broadcast transmissions through a BPEC and wish to investigate the potential benefits, in terms of achieved rates, of using ACK/NACK feedback. The above setting is motivated by increasingly popular applications such as wireless delivery of subscription content, where multiple users may ask for the same content (file, video, etc.) and multiple distinct sessions may be simultaneously active.

Although, in the absence of feedback, timesharing between capacity achieving schemes (say, via network coding [3]) for each multicast group is rate-optimal, recent work on BPEC under similar settings has shown that feedback can actually increase the capacity region beyond what is achieved by timesharing, at the cost of increased encoding complexity. The latter is due to the fact that the transmitter must now keep track of the entire erasure event history, as obtained through feedback, and properly combine packets for transmission in the spirit of network coding.

Apart from exploring the inherent performance/complexity tradeoff of various feedback schemes, this document also examines the special case $N \to \infty$ (which is motivated by the fact that the number of subscribed users in a content delivery system may be more than 100) to determine whether feedback still offers rate benefits in this asymptotic regime. A negative answer to this question would indicate that timesharing is asymptotically optimal, which would greatly simplify the employed encoding algorithms for large N.

Our contribution is as follows:

- for iid erasures, we show that a well-known feedback capacity upper bound, which is tight for $N = 1$, is also tight for $N = 2$ by proposing a virtual queue-based coding algorithm that achieves the bound for any $0 < \epsilon < 1$, where ϵ is the erasure probability.
- since a direct extension of the algorithm for $N = 2$ to higher N requires an exponential number of virtual queues, we propose a simple (suboptimal) algorithm which only operates on 3 queues, for *arbitrary* N, and still outperforms timesharing for any finite N. The determination of capacity achieving algorithms for $N > 2$ remains an open problem.

[‡] This work was performed while the first author was hosted by the ARNI lab at EPFL under a summer internship.
[§] This work was supported by the ERC Starting Grant Project NOWIRE ERC-2009-StG-240317.

- since the performance of the above algorithm, as well as any other algorithm we have devised so far, becomes identical to timesharing as $N \to \infty$, we conjecture that timesharing is, in fact, asymptotically optimal as $N \to \infty$. We provide a partial result to support this conjecture by computing an upper bound on the sum rate, under a special relation between N and n (number of transmissions), and showing that it matches the timesharing sum rate.

A. *Related work*

The N-user broadcast packet erasure channel (BPEC) has been traditionally used as a non-trivial abstract model for lossy wireless networks. Although its general capacity remains unknown, important special cases have been solved, including the case of N unicast sessions with feedback [4], [1] (where it is shown that feedback can increase the capacity region) and the case of multiple sources/multiple destinations in a directed acyclic graph [5], where each destination must decode the messages from *all* sources and the destinations know the exact erasure events in all links. For technical reasons, which will be explained later, the problem examined in the current work cannot be cast into the setting of [5]. Furthermore, the two message sets in our paper are non-degraded, so that we cannot invoke results from relevant literate on degraded messages [6] (most of which does not take feedback into account in the first place).

Nevertheless, the proposed token-based approach in [1], [2] still provides some general insight and guidelines which can be applied here as well. The key insight in these works is to exploit the ACK/NACK feedback in an erasure channel to keep track (via queues) of which user received which symbols and then suitably combine multiple symbols for transmission, in the spirit of network coding [3], to provide "useful" symbols for multiple users. This is a general idea which has been applied in [7] for two unicast sessions with distinct sources and saturated traffic, where only one source can transmit in each slot (and each source can overhear the other source's transmission), as well as in [8], which considers broadcast messages with stochastic arrivals. The difference between the last two works and the current work lies in the fact that the efficient processing of the various queues (i.e. the packet combining), which is crucial towards achieving high rates, greatly depends on the assumed message structure and is quite different in each case.

This document is structured as follows: Section II contains the exact system model, while Section III contains a capacity outer bound, which is shown to be tight for $N = 2$. The description of the capacity achieving scheme is also provided. Section IV presents a low complexity algorithm which outperforms timesharing, for any finite N, by using only 3 queues, while Section V contains an asymptotic analysis of the sum rate as $N \to \infty$ and provides a partial result regarding the asymptotic optimality of timesharing, assuming that N is allowed to vary with n (number of transmitted symbols). Section VI concludes this document.

II. System model

We consider a time-slotted system where a single source/transmitter wants to transmit multicast messages to 2 groups, namely $\mathcal{G}_1$ and $\mathcal{G}_2$, consisting of N users each, as shown in Fig. 1. Hence, all users in $\mathcal{G}_1 \triangleq \{1, \dots, N\}$ should receive message W_1 while all users in $\mathcal{G}_2 \triangleq \{N+1, \dots, 2N\}$ should receive message W_2, where W_1, W_2 are independent. Each transmission is of a broadcast type, i.e. the source transmits one symbol per slot, which may be received by any subset of $\mathcal{G} \triangleq \mathcal{G}_1 \cup \mathcal{G}_2$. Notice that this model cannot be directly handled by [5], since each group has a distinct multicast session, and cannot be converted into a setting compatible with [5] without introducing cycles, thus invalidating the main assumption of that work.

The channel between the source and each user is modeled as memoryless erasure, i.e. either the transmitted symbol is received unaltered by the user with probability $1 - \epsilon$, or the symbol is "erased" by the user with probability ϵ. The latter case is equivalent to considering that the user received a special symbol E, which is distinct from any possible broadcast symbol. At the end of each slot, each user sends feedback information (through a separate error-free and zero delay channel) to inform the transmitter whether the broadcast symbol was successfully received, i.e. feedback consists of a simple ACK/NACK reply.

In information theoretic terms, the above system is described by the tuple $(\mathcal{X}, (\mathcal{Y}_i : i \in \mathcal{G}), p(\boldsymbol{Y}_l, X_l))$, where $\mathcal{X}$ is the input symbol alphabet, $\mathcal{Y}_i = \mathcal{X} \cup \{E\}$ is the output symbol alphabet for user i (including the erasure symbol $E \notin \mathcal{X}$) and $p(\boldsymbol{Y}_l | X_l)$ is the probability of having, at slot l, output $\boldsymbol{Y}_l \triangleq (Y_{i,l} : i \in \mathcal{G})$ for a transmitted

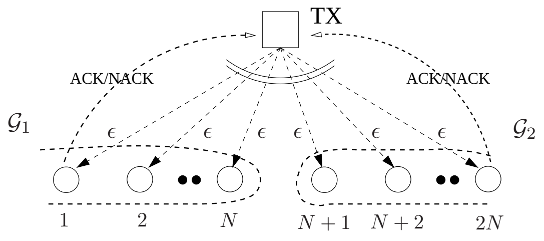

Fig. 1. The system under investigation.

(input) symbol X_l. At the end of slot l, each user i sends back a one bit ACK/NACK $Z_{i,l} = \mathbb{I}[Y_{i,l} \neq E]$ indicating whether the packet was successfully received or not.

A channel code (M_1, M_2, n) with feedback is defined for this system as the aggregate of the following components (the following is a natural extension of the standard definitions in [9] and is taken directly from [1]):

- message sets $\mathcal{W}_j$, with $|\mathcal{W}_j| = M_j$ for $j = 1, 2$, intended for all users in group $\mathcal{G}_j$, respectively, where $|\cdot|$ denotes set cardinality. We denote with $\boldsymbol{W} \triangleq (W_1, W_2)$ the message that needs to be transmitted and assume that this message is uniformly distributed in $\mathcal{W} \triangleq \mathcal{W}_1 \times \mathcal{W}_2$. Equivalently, we can identify the message set $\mathcal{W}_j$ as a set of packets $\mathcal{K}_j$ that *all* users in $\mathcal{G}_j$ should receive. We also denote $K_j = |\mathcal{K}_j|$.
- an encoder that selects a symbol $X_l = f_l(\boldsymbol{W}, \boldsymbol{Y}^{l-1})$ for transmission at slot l, for $1 \leq l \leq n$, based on message $\boldsymbol{W}$ and all previously gathered feedback $\boldsymbol{Y}^{l-1} \triangleq (\boldsymbol{Y}_1, \dots, \boldsymbol{Y}_l)$. X_1 is obviously a function of $\boldsymbol{W}$ only. Notice that, although the source only receives $\boldsymbol{Z}_l = (Z_{i,l} : i \in \mathcal{G})$ as feedback from the users, it can always deduce $\boldsymbol{Y}_l$ from $\boldsymbol{Z}_l$ since it knows X_l (i.e. knowledge of $\boldsymbol{Y}^{l-1}$ implies knowledge of $\boldsymbol{Z}^{l-1}$ and vice versa). This justifies the previous selection for the encoding function f_l.
- decoding functions (i.e. decoders), one for each user $i \in \mathcal{G}$, of the form $g_i : \mathcal{Y}_i^n \to \mathcal{W}_1$ for $i \in \mathcal{G}_1$ and $g_i : \mathcal{Y}_i^n \to \mathcal{W}_2$ for $i \in \mathcal{G}_2$. Hence, the reconstructed symbol at user i is $\hat{W}_i = g_i(Y_i^n)$, where $Y_i^n \triangleq (Y_{i,1}, \dots, Y_{i,n})$ is the sequence of symbols received by user i (including any erasures E) after n slots.

The probability of error for message $\boldsymbol{W}$ is $\lambda_n(\boldsymbol{W}) = \Pr(\cup_{i\in\mathcal{G}_1}\{g_i(Y_i^n) \neq W_i\} \cup \cup_{i\in\mathcal{G}_2}\{g_i(Y_i^n) \neq W_i\} | \boldsymbol{W})$ while the rate for this code, in information bits per transmitted symbol, is $\boldsymbol{R} = (R_1, R_2)$, where $R_j = (\log_2 M_j)/n$. Let $\mathfrak{C}$ be any class of codes (M_1, M_2, n). Then, rate $\boldsymbol{R} = (R_1, R_2)$ is achievable *under* $\mathfrak{C}$ (equivalently, $\mathfrak{C}$ achieves $\boldsymbol{R}$) if there exists a sequence of $(\lceil 2^{nR_1} \rceil, \lceil 2^{nR_2} \rceil, n)$ codes in $\mathfrak{C}$ such that $\bar{P}_e = \frac{1}{|\mathcal{W}|}\sum_{\boldsymbol{W}\in\mathcal{W}} \lambda_n(\boldsymbol{W}) \to 0$ as $n \to \infty$. We also define rate $\boldsymbol{R}$ to be achievable if there exists some class $\mathfrak{C}$ of codes that achieves $\boldsymbol{R}$. The closure of the set of rates that are achievable under $\mathfrak{C}$ constitutes the achievable region of $\mathfrak{C}$, denoted as $\mathcal{R}_{\mathfrak{C}}$, while the capacity region $\mathcal{C}$ of the channel is the closure of the set of achievable rates. We will also write $\mathcal{C}(N)$ to emphasize the fact that the capacity region is an implicit function of N; it clearly holds $\mathcal{C}(N) \supseteq \mathcal{C}(N+1)$ for all N.

III. Achieving capacity for $N = 2$

Although the feedback capacity region of the above system is not known in general, the property $\mathcal{C}(N) \supseteq \mathcal{C}(N+1)$ implies that a global outer bound $\mathcal{C}^{out}$ is equal to $\mathcal{C}(1)$, i.e. the capacity region for a 2-user system with 2 unicast sessions. This problem has been solved in [10], whence the next bound follows

$$\mathcal{C}^{out} = \mathcal{C}(1) = \left\{ (R_1, R_2) \geq \mathbf{0} : \max\left(\frac{R_1}{1-\epsilon} + \frac{R_2}{1-\epsilon^2}, \frac{R_1}{1-\epsilon^2} + \frac{R_2}{1-\epsilon} \right) \leq \log_2|\mathcal{X}| \right\}, \tag{1}$$

where R_1, R_2 are measured in bits per information symbol and capacity is achieved by an inter-session mixing algorithm. This bound is independent of N, which raises the question of whether it is tight for $N \geq 2$. A direct extension of the optimal algorithm in [10] to $N \geq 2$ is non-trivial since there is no obvious way for determining the most "efficient" way of combining symbols due to the exploding combinatorial nature of the problem. However, in this Section, we prove the following result

Theorem 1: The capacity outer bound $\mathcal{C}(1)$ is also tight for $N = 2$, i.e. $\mathcal{C}(2) = \mathcal{C}(1)$, for all $0 < \epsilon < 1$ and this bound is achieved by algorithm `OPT2` described below.

Proof: The achievability of $\mathcal{C}(1)$ by `OPT2` will be determined after the detailed description of the algorithm. ■

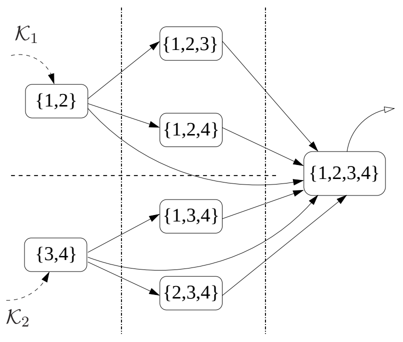

Fig. 2. Queue structure for `OPT2`. Ovals denote queues, the sets inside the ovals denote the $\mathcal{S}$ corresponding to $Q_{\mathcal{S}}$ and lines with arrows indicate possible index transition under the proposed algorithm.

A. *Description of algorithm* `OPT2`

The transmitter maintains a group of virtual queues $Q_{\mathcal{S}}$, indexed by the non-empty set $\mathcal{S}$, where $\mathcal{S} \subseteq \mathcal{G}$ and *exactly* one of $\mathcal{S} \supseteq \mathcal{G}_1$, $\mathcal{S} \supseteq \mathcal{G}_2$ is true (see Fig. 2 for a graphical depiction). A non-negative integer index $K^i_{\mathcal{S}}$, for each $i \in \mathcal{S}$, is associated to queue $Q_{\mathcal{S}}$. Both $Q_{\mathcal{S}}$ and $K^i_{\mathcal{S}}$ are dynamically updated during the algorithm's operation; the rationale for introducing these entities will be explained later.

Initialization: the packets of set $\mathcal{K}_j$ are placed into queue $Q_{\mathcal{G}_j}$, for $j = 1, 2$, as shown by the dashed arrowed lines of Fig. 2, while all other queues are empty. We also set $K^1_{\{1,2\}} = K^2_{\{1,2\}} = 0$ and $K^3_{\{3,4\}} = K^4_{\{3,4\}} = 0$, while all other indices $K^i_{\mathcal{S}}$ are set to zero. For nomenclature purposes, we define "level l" as the group of queues $Q_{\mathcal{S}}$ with $|\mathcal{S}| = l$.

Encoding: the source/transmitter sequentially processes the queues in each level, in ascending level order (relative order within a given level is unimportant). Hence, there are 3 encoding phases corresponding to the processing of levels 2–4 of Fig. 2, respectively. A common feature to all phases is that the source treats the packets stored in the queues as elements of a finite field $\mathbb{F}_q$ with size q (i.e. $\mathcal{X} = \mathbb{F}_q$) and transmits a linear combination s, over $\mathbb{F}_q$, of *all* packets in the queue currently being processed (potentially combining them with packets in $Q_{\mathcal{G}}$, in certain cases, as will be described). The concept of *token*, introduced in [1], [2], will be useful and is described next.

Definition 1: A transmitted packet s at slot t is a token for user $i \in \mathcal{G}$ iff it can be written as

$$s = \sum_{u \in \mathcal{D}_i} b^{(i)}_s(u)u + c^{(i)}_s,$$

where $\mathcal{D}_i$ is the set of packets intended for user i (i.e. $\mathcal{D}_i = \mathcal{K}_j$ for $i \in \mathcal{G}_j$), and the values of $b^{(i)}_s(u)$, $c^{(i)}_s$ are known to user i at the beginning of slot t. We also define $\boldsymbol{b}^{(i)}_s \triangleq (b^{(i)}_s(u) : u \in \mathcal{D}_i)$.

Definition 2: A set $\mathcal{T}$ of tokens for user i is linearly independent iff the corresponding set of coefficient vectors $\{\boldsymbol{b}^{(i)}_s : s \in \mathcal{T}\}$ is linearly independent over $\mathbb{F}_q$.

It is clear from Definition 1 that "tokeness" is a time-varying property, in the sense that a packet s may not necessarily be a token for user i at the time of its transmission but may become so after successful reception by i (since it trivially holds $s = c^{(i)}_s$). Furthermore, tokeness is absorbing, i.e. once a packet becomes token for user i, it remains so forever. It also follows from the definition that tokeness is preserved under linear combinations, as described in the following proposition.

Proposition 1: Let $\mathcal{P}$ be a set of packets such that each packet $p \in \mathcal{P}$ is a token for *all* users $i \in \mathcal{S}$. Then, *any* linear combination $\sum_{p \in \mathcal{P}} a(p)p$ is also a token for all users in $\mathcal{S}$.

In all subsequent cases, we denote with $\mathcal{O}$ the set of users which successfully receive a packet. The exact value of $\mathcal{O}$ is conveyed to the source through feedback from the users.

Phase 1: the source individually processes each queue $Q_{\mathcal{S}}$, where $|\mathcal{S}| = 2$ (i.e. $Q_{\mathcal{G}_1}$, $Q_{\mathcal{G}_2}$), and transmits a linear combination $s = \sum_{p \in Q_{\mathcal{S}}} a_s(p)p$, where $a_s(p)$ are selected randomly and uniformly in $\mathbb{F}_q$ (this rule for generating $a_s(p)$ will also apply to all subsequent phases). The generator of $a_s(p)$ is also available at the users so that they always know the values of $a_s(p)$ for a transmitted packet s even if they don't successfully receive s. After getting user feedback and learning $\mathcal{O}$, the source takes the following actions, or steps (the actions are not mutually exclusive so that all conditions should be checked and steps 1,2 can both be performed in a single transmission):

TABLE I
DEMONSTRATION (PARTIAL) OF EXECUTING OPT2.

Phase 1. Processing $Q_{\mathcal{S}}$: $\mathcal{S} = \{1, 2\}$	
Feedback	Actions w.r.t. users 1, 2 if $K^1_{\{1,2\}} > 0$, $K^2_{\{1,2\}} > 0$
$1, \bar{2}, \bar{3}, \bar{4}$	$K^1_{\{1,2\}}--$; (S.1 for user 1)
$\bar{1}, 2, \bar{3}, \bar{4}$	$K^2_{\{1,2\}}--$; (S.1 for user 2)
$\bar{1}, \bar{2}, 3, \bar{4}$	$K^1_{\{1,2\}}--$, $K^1_{\{1,2,3\}}++$; (S.2 for user 1) $K^2_{\{1,2\}}--$, $K^2_{\{1,2,3\}}++$; (S.2 for user 2)
$\bar{1}, 2, 3, 4$	$K^1_{\{1,2\}}--$, $K^1_{\mathcal{G}}++$; (S.2 for user 1) $K^2_{\{1,2\}}--$; (S.1 for user 2)
$\bar{1}, \bar{2}, \bar{3}, \bar{4}$	retransmit (S.3)
Phase 2. Proc. $Q_{\mathcal{S}}$ with $Q_{\mathcal{G}}$: $\mathcal{S} = \{1, 2, 3\}$ ($\tilde{G}_{\mathcal{S}} = \{1, 2\}$, $\alpha(\mathcal{S}) = 3$)	
Feedback	Action w.r.t. users 1, 2, 3

$1, \bar{2}, 3, 4$	if ($K^1_{\{1,2,3\}} > 0$) then $K^1_{\{1,2,3\}}--$; (S.1a for user 1)
	else if ($K^1_{\mathcal{G}} > 0$) then $K^1_{\mathcal{G}}--$; (S.1b for user 1)
	if ($K^3_{\mathcal{G}} > 0$) then $K^3_{\mathcal{G}}--$; (S.3 for user 3)
	if ($K^2_{\{1,2,3\}} > 0$) then $K^2_{\{1,2,3\}}--$, $K^2_{\mathcal{G}}++$; (S.2 for user 1)

$\bar{1}, \bar{2}, 3, \bar{4}$	if ($K^3_{\mathcal{G}} > 0$) then $K^3_{\mathcal{G}}--$; (S.3 for user 3)
	else retransmit; (S.4)

1) for each $i \in \mathcal{S} \cap \mathcal{O}$ with $K^i_{\mathcal{S}} > 0$, decrease $K^i_{\mathcal{S}}$ by one.
2) if s is erased by at least one user in $\mathcal{S}$ (i.e. $\mathcal{S} \cap \mathcal{O}^c \neq \varnothing$, where ^{c} denotes set complement w.r.t. $\mathcal{G}$) *and* received by at least one user outside $\mathcal{S}$ (i.e. $\mathcal{O} \cap \mathcal{S}^c \neq \varnothing$), then packet s is added to queue $Q_{\mathcal{S} \cup \mathcal{O}}$ and for each $i \in \mathcal{S} \cap \mathcal{O}^c$ with $K^i_{\mathcal{S}} > 0$ the source performs the following actions: $K^i_{\mathcal{S}}$ is decreased by one while $K^i_{\mathcal{S} \cup \mathcal{O}}$ is increased by one.
3) if none of the above conditions are satisfied, s is retransmitted without generating new coefficients $a_s(p)$.

Queue $Q_{\mathcal{S}}$ is processed until it holds $K^i_{\mathcal{S}} = 0$ for *all* $i \in \mathcal{S}$. Phase 1 is complete when both level 2 queues have been processed as described above. Table I contains a (non-exhaustive) list of examples, written in C-style pseudocode, of checking the previous conditions and taking suitable actions. The feedback column contains the erasure events (the absence/presence of a bar above a number denotes a successful reception/erasure for that user) while the action column lists the appropriate actions (or steps). The number after S. denotes the corresponding step of phase 1. Clearly, different steps may be taken for different users.

Phase 2: each queue $Q_{\mathcal{S}}$ in level 3 is individually combined with $Q_{\mathcal{G}}$ in level 4 (this is still counted as "processing $Q_{\mathcal{S}}$", although two different queues are considered) by transmitting a packet $s = \sum_{p \in Q_{\mathcal{S}} \cup Q_{\mathcal{G}}} a_s(p)p$. Notice that, by construction of the queues, for each index set $\mathcal{S}$ in a level 3 queue, exactly one of $\mathcal{S} \supseteq \mathcal{G}_1$, $\mathcal{S} \supseteq \mathcal{G}_2$ holds. Define $\tilde{\mathcal{G}}_{\mathcal{S}}$ to be either $\mathcal{G}_1$ or $\mathcal{G}_2$, depending on which of the above conditions is true for a given $\mathcal{S}$ and denote with $\alpha(\mathcal{S})$ the member of the singleton set $\mathcal{S} \cap \tilde{\mathcal{G}}^c_{\mathcal{S}}$. For instance, for $\mathcal{S} = \{1, 2, 3\}$, it holds $\tilde{\mathcal{G}}_{\mathcal{S}} = \{1, 2\}$ and $\alpha(\mathcal{S}) = \{3\}$. The following actions are now performed (again, all cases must be considered):

1) for each $i \in \tilde{\mathcal{G}}_{\mathcal{S}} \cap \mathcal{O}$:
   a) if $K^i_{\mathcal{S}} > 0$, then $K^i_{\mathcal{S}}$ is decreased by one.
   b) if $K^i_{\mathcal{S}} = 0$ and $K^i_{\mathcal{G}} > 0$, then $K^i_{\mathcal{G}}$ is decreased by one.
2) if s is erased by at least one user in $\tilde{\mathcal{G}}_{\mathcal{S}}$ *and* received by user $\alpha(\mathcal{S})$, then s is added to $Q_{\mathcal{G}}$ and for each $i \in \tilde{\mathcal{G}}_{\mathcal{S}} \cap \mathcal{O}^c$ with $K^i_{\mathcal{S}} > 0$, $K^i_{\mathcal{S}}$ is decreased by one and $K^i_{\mathcal{G}}$ is increased by one.
3) if user $\alpha(\mathcal{S})$ received s and $K^{\alpha(\mathcal{S})}_{\mathcal{G}} > 0$, then $K^{\alpha(\mathcal{S})}_{\mathcal{G}}$ is decreased by one.
4) if none of the previous conditions is satisfied, s is retransmitted.

Table I also contains some examples of applying various steps of phase 2. In contrast to phase 1, a queue need not be processed in contiguous slots, i.e. it is possible to process $Q_{\{1,2,3\}}$ for some slots, switch to processing $Q_{\{1,2,4\}}$ and then revert to $Q_{\{1,2,3\}}$. The switch from a queue $Q_{\mathcal{S}_1}$ to another queue $Q_{\mathcal{S}_2}$ in level 3 is performed when, for some $i \in \mathcal{S}_1$, both $K^i_{\mathcal{S}_1}$ and $K^i_{\mathcal{G}}$ are equal to zero. At this point, a queue $Q_{\mathcal{S}_2}$ is selected such that $i \in \mathcal{S}_2$ and $K^i_{\mathcal{S}_2} > 0$ (so that it is possible to increase $K^i_{\mathcal{G}}$ due to step 2 of phase 2). No switch is made if no such $\mathcal{S}_2$ exists.

Each level 3 queue $Q_{\mathcal{S}}$ is processed until it holds $K_{\mathcal{S}}^i = 0$ for all $i \in \mathcal{S}$, and phase 2 is complete when all level 3 queues have been processed.

Phase 3: only $Q_{\mathcal{G}}$ is processed and the transmitted packet s has the form $s = \sum_{p \in Q_{\mathcal{G}}} a_s(p)p$. After the transmitter gets feedback and learns $\mathcal{O}$, it performs the following: for each $i \in \mathcal{O}$ with $K_{\mathcal{G}}^i > 0$, $K_{\mathcal{G}}^i$ is decreased by 1. This phase is complete when it holds $K_{\mathcal{G}}^i = 0$ for all $i \in \mathcal{G}$.

Decoding: a standard random network coding argument, similar to the one used in [1], shows that, for a sufficiently large field size q (namely, $q > 2N$), the random coefficients $a_s(p)$ for each transmission can be selected such that each user $i \in \mathcal{G}_j$ has received, with high probability, K_j linearly independent tokens s by the end of the algorithm. Since $\boldsymbol{b}_s^{(i)}$, $c_s^{(i)}$ are known to i, each user can solve the resulting linear system and decode its intended packets.

B. Intuition behind the algorithm

Inspired by [2], the algorithm operates on the following premise: the packets should be combined in such a way that the transmitted packet s allows any user i that receives it to *either* create, if possible, a new equation for its unknown packets (which is linearly independent w.r.t. previously created equations by i) *or* gain new side information which can be exploited in the future. In other words, the algorithm becomes more rate-efficient if the transmitted packet always offers some benefit to as many users as possible. The virtual queues are used to keep track of overhearing (i.e. which user received which packets), which is helpful in choosing which packets to combine. In fact, the following property can be proved, via induction on time (see [1] for a similar proof in a different setting).

Property 1: Under `OPT2`, the following is true for any t: all packets stored in $Q_{\mathcal{S}}$ at the beginning of slot t are tokens for all $i \in \mathcal{S}$. Furthermore, the transmitted linear combination s at slot t can be selected such that $\boldsymbol{b}_s^{(i)}$ is linearly independent with respect to the set $\{\boldsymbol{b}_{s'}^{(i)} : s' \text{ has been received by } i \text{ prior to } t\}$ for all $i \in \mathcal{S}$ with $K_{\mathcal{S}}^i > 0$.

Thus, $K_{\mathcal{S}}^i$ should be interpreted as the number of linearly independent equations that user i still needs to create from packets in $Q_{\mathcal{S}}$.

As user $i \in \mathcal{S}$ receives properly constructed linear combinations s from $Q_{\mathcal{S}}$ (in the sense described in Property 1), $K_{\mathcal{S}}^i$ is decreased until it becomes zero, at which point user i has received all available useful information from $Q_{\mathcal{S}}$. If some $K_{\mathcal{S}}^i$ is zero when processing of $Q_{\mathcal{S}}$ begins, cross-level combining should be used, as described in phase 2; this is necessary to avoid inefficiency since, in case $Q_{\mathcal{S}}$ is processed by itself and the transmitted packet is only received by a user $i \in \mathcal{S}$ which already has $K_{\mathcal{S}}^i = 0$ (e.g. user $i = 3$ for $\mathcal{S} = \{1, 2, 3\}$ always has $K_{\mathcal{S}}^i = 0$), this transmission offers no benefit to i. Cross-level combining and step 3 of phase 2 imply that the latter case can still provide a benefit to user i as long as $K_{\mathcal{G}}^i > 0$. Hence, even with cross-level combining, an efficient (in terms of rate) algorithm should guarantee that *not* all $K_{\mathcal{G}}^i$ indices, for $i \in \mathcal{G}$, become zero while there is still some non-zero $K_{\mathcal{S}}^j$ index in level 3.

Hence, the various feedback-based actions of phase 1 should be interpreted as follows: step 1 captures the fact that, from Property 1, any user $i \in \mathcal{S}$ with $K_{\mathcal{S}}^i > 0$ that receives the transmitted s gains a new useful token, so that $K_{\mathcal{S}}^i$ should be reduced accordingly. Step 2 captures the fact that a packet s that is not received by user $i \in \mathcal{S}$ but is received by at least one user outside $\mathcal{S}$ has become a token for at least one new user (in fact, it has become a token for all users in $\mathcal{O} \cap \mathcal{S}^c$) so that, using Proposition 1 and Property 1, it has become a token for all users in set $\mathcal{S} \cup \mathcal{O}$ and can be moved to a higher level queue. Hence, any user $i \in \mathcal{S}$ with $K_{\mathcal{S}}^i > 0$ that did not receive s can potentially get the missing information, in the future, from a linear combination coming out of $Q_{\mathcal{S} \cup \mathcal{O}}$ instead of $Q_{\mathcal{S}}$; the indices $K_{\mathcal{S}}^i$, $K_{\mathcal{S} \cup \mathcal{O}}^i$ are then modified accordingly to capture this deferment. Finally, step 3 of phase 1 merely states that a packet is retransmitted if none of the above conditions hold, in which case the current transmission offered no benefit to any users.

The actions of phase 3 are interpreted similarly to phase 1, the main difference being that there is no equivalent action to step 2 of phase 1, since the set is processed in phase 3 is maximal (i.e. for $\mathcal{S} = \mathcal{G}$, the set outside $\mathcal{S}$ is empty). The actions of phase 2 are also interpreted similarly to phase 1, with the important addition of step 3 in phase 2, which ensures that a packet that is only received by $\alpha(\mathcal{S})$ provides a benefit to this user.

C. Performance analysis of `OPT2`

Following [2], we compute the average number of slots $T^*_{\mathcal{S}}$ required by `OPT2` to process $Q_{\mathcal{S}}$, so that `OPT2` achieves all rates (R_1, R_2) with $R_j = (\log_2 q)K_j / \sum_{\mathcal{S}} T^*_{\mathcal{S}}$ (units are information bits per transmitted symbol)[1], $j = 1, 2$, where the $\mathcal{S}$ summation is performed over all index sets shown in Fig. 2. Denoting with $T^*_{i,\mathcal{S}}$ the average number of slots required, during the processing of $Q_{\mathcal{S}}$, for $K^i_{\mathcal{S}}$ to become 0, it clearly holds $T^*_{\mathcal{S}} = \max_{i\in\mathcal{S}} T^*_{i,\mathcal{S}}$. To compute $T^*_{i,\mathcal{S}}$, it suffices to compute the value of $K^i_{\mathcal{S}}$ when processing of $Q_{\mathcal{S}}$ begins. This can be easily performed by sequentially analyzing each of the 3 phases. We denote with $\hat{t}_k$ the time instant when phase k is complete

In phase 1, $K^i_{\mathcal{G}_1}$ is *not* decreased by one if the transmitted packet is erased by user i *as well as* both users in $\mathcal{G}_2$, so that

$$\begin{aligned} T^*_{i_1,\mathcal{G}_1} &= \frac{K_1}{1-\epsilon^3} \quad \forall\, i_1 \in \mathcal{G}_1, \\ T^*_{i_2,\mathcal{G}_1} &= \frac{K_2}{1-\epsilon^3} \quad \forall\, i_2 \in \mathcal{G}_2, \end{aligned} \tag{2}$$

while the actions performed in step 2 of phase 1 imply that

$$\begin{aligned} K^i_{\mathcal{S}}(\hat{t}_1) &= T^*_{i,\mathcal{G}_{\mathcal{S}}}\epsilon^2(1-\epsilon) \quad \forall\, \mathcal{S}:\ |\mathcal{S}| = 3,\ i \in \mathcal{S}, \\ K^i_{\mathcal{G}}(\hat{t}_1) &= T^*_{i,\mathcal{G}_j}\epsilon(1-\epsilon)^2 \quad \forall\, i \in \mathcal{G}. \end{aligned} \tag{3}$$

In phase 2, $K^i_{\mathcal{S}}$, where $|\mathcal{S}| = 3$, is not decreased by one if the transmitted packet is erased by user i *as well as* $\alpha(\mathcal{S})$ so that

$$T^*_{i,\mathcal{S}} = \frac{K^i_{\mathcal{S}}(\hat{t}_1)}{1-\epsilon^2} \quad \forall\, \mathcal{S}:\ |\mathcal{S}| = 3,\ i \in \mathcal{S}. \tag{4}$$

The actions of phase 2 allow $K^i_{\mathcal{G}}$ to either decrease (steps 1b, 3) or increase (step 2), so that considering the cumulative effect of these actions leads to

$$K^i_{\mathcal{G}}(\hat{t}_2) = \left[K^i_{\mathcal{G}}(\hat{t}_1) + \sum_{\substack{\mathcal{S}:\mathcal{S}\supseteq\mathcal{G}_j \\ |\mathcal{S}|=3}} \left[T^*_{i,\mathcal{S}}\epsilon(1-\epsilon) - (T^*_{\mathcal{S}} - T^*_{i,\mathcal{S}})(1-\epsilon) \right] - T^*_{\mathcal{G}^c_j}(1-\epsilon) \right]^+, \tag{5}$$

where $[x]^+ \triangleq \max(x, 0)$ and $i \in \mathcal{G}_j$. Notice that, due to the iid symmetry in (3), it holds $T^*_{\mathcal{S}} = T^*_{i,\mathcal{S}}$, for all $i \in \mathcal{S}$ and $|\mathcal{S}| = 3$, so that the second term of the summation over $\mathcal{S}$ in (5) is equal to zero. Finally, for phase 3, it holds $T^*_{i,\mathcal{G}} = \frac{K^i_{\mathcal{G}}(\hat{t}_2)}{1-\epsilon}$, since $K^i_{\mathcal{G}}$ is decreased by one only when the transmitted packet is received by i.

Performing the algebra in the above expressions yields

$$\begin{aligned} K^1_{\mathcal{G}}(\hat{t}_2) = K^2_{\mathcal{G}}(\hat{t}_2) &= \frac{\epsilon}{(1+\epsilon)(1+\epsilon+\epsilon^2)} \left[(1+\epsilon^2)K_1 - \epsilon K_2\right]^+, \\ K^3_{\mathcal{G}}(\hat{t}_2) = K^4_{\mathcal{G}}(\hat{t}_2) &= \frac{\epsilon}{(1+\epsilon)(1+\epsilon+\epsilon^2)} \left[(1+\epsilon^2)K_2 - \epsilon K_1\right]^+, \end{aligned} \tag{6}$$

and since $1+\epsilon^2 > \epsilon$, for any $0 < \epsilon < 1$, we conclude that at least one of $K^1_{\mathcal{G}}(\hat{t}_2)$, $K^3_{\mathcal{G}}(\hat{t}_2)$ is strictly positive, so that the $K^i_{\mathcal{G}}$ indices do not become all zero before the corresponding level 3 indices and the inefficient case mentioned in Section III-B is avoided. Finally, for phase 3 it holds

$$T^*_{i,\mathcal{G}} = \frac{K^i_{\mathcal{G}}(\hat{t}_2)}{1-\epsilon} \quad \forall\, i \in \mathcal{G}. \tag{7}$$

We now have all ingredients for the proof of Theorem 1.

Proof of Theorem 1: From the previous expressions, it is a matter of simple algebra (made easier with the help of Maple) to compute $\sum_{\mathcal{S}} T^*_{\mathcal{S}}$ and R_1, R_2 as functions of K_1, K_2, ϵ and eliminate K_1, K_2 to show that `OPT2` achieves the region $\mathcal{C}(1)$. ■

The previous analysis and proof of Theorem 1 admit a straightforward extension to the case of spatially independent but not identical channels, although the algebra becomes quite tedious to the point that it hides some of the intuition behind `OPT2`. The reader is referred to the Appendix for more details.

[1]a rigorous information-theoretic analysis, according to the definitions of Section II, can be performed using a truncation argument for a finite blocklength n and, obviously, leads to the same result (see [1] for such an analysis in a slightly different setting).

IV. A LOW COMPLEXITY ALGORITHM FOR $N > 2$

Optimally generalizing the previous algorithm to higher N is not straightforward since the number of virtual queues, as well as the possible ways of selecting queues for cross-level combining, increases exponentially. However, we provide next a simple (suboptimal) algorithm for arbitrary N and evaluate its performance compared to a timesharing (TS) scheme, which is used as a benchmark tool and described as follows.

TS scheme: the source first communicates message W_1 to all users in group $\mathcal{G}_1$, using any code (say, a standard network coding based scheme [3]) that achieves the multicast cut-set bound for $\mathcal{G}_1$ *only*. The source then communicates message W_2 to all users in $\mathcal{G}_2$, using an identical approach to achieve the cut-set bound for $\mathcal{G}_2$ *only*.

The achievable region $\mathcal{R}_{\texttt{TS}}$ of the TS scheme is

$$\mathcal{R}_{\texttt{TS}} = \{(R_1, R_2) \geq \mathbf{0} : R_1 + R_2 \leq (1-\epsilon)\log_2|\mathcal{X}|\}, \tag{8}$$

so that we aim in constructing codes which achieve a rate region that is a superset of $\mathcal{R}_{\texttt{TS}}$. Note that $\mathcal{R}_{\texttt{TS}}$ can also be written in the form

$$\mathcal{R}_{\texttt{TS}} = \left\{(R_1, R_2) \geq \mathbf{0} : \max\left(\frac{R_1}{1-\epsilon} + \frac{R_2}{\alpha_{\texttt{TS}}(\epsilon)(1-\epsilon^2)}, \frac{R_2}{1-\epsilon} + \frac{R_1}{\alpha_{\texttt{TS}}(\epsilon)(1-\epsilon^2)}\right) \leq \log_2|\mathcal{X}|\right\}, \tag{9}$$

where $\alpha_{\texttt{TS}}(\epsilon) = \frac{1}{1+\epsilon}$. Clearly, any algorithm that employs some form of feedback-based session mixing is expected to perform better than TS. In fact, one can propose the following straightforward extension, called EXTN, of OPT2 to arbitrary N.

EXTN (applicable for arbitrary N):

- construct virtual queues $Q_{\mathcal{S}}$, for all $\mathcal{S} \subseteq \mathcal{G}$ that satisfy exactly one of the relations $\mathcal{S} \supseteq \mathcal{G}_1$ and $\mathcal{S} \supseteq \mathcal{G}_2$, and associate indices $K^i_{\mathcal{S}}$, for $i \in \mathcal{S}$, to each $Q_{\mathcal{S}}$. Hence, the virtual queues will lie at levels $N, N+1, \ldots, 2N$. The interpretation of the queues and indices is the same as in OPT2.
- initialize the virtual queues and indices exactly as in OPT2.
- individually process each of the queues $Q_{\mathcal{S}}$ in levels N through $2N-2$ (i.e. all levels except for the two highest) in the same manner as OPT2, i.e. transmit linear combinations of all packets in $Q_{\mathcal{S}}$ and apply the same feedback based actions as in phase 1 of OPT2.
- for the queues in level $2N-1$ (and only for these), perform cross level combining with queue $Q_{\mathcal{G}}$ and apply the actions of phase 2 of OPT2. Finally, $Q_{\mathcal{G}}$ is processed by itself, analogously to phase 3 of OPT2.

The performance analysis for EXTN is similar to that of OPT2 (albeit more tedious, due to the exponential increase of the number of queues) and is based on computing the values of $K^i_{\mathcal{S}}$ at various stages of the algorithm and using them to compute the number of slots $T^*_{\mathcal{S}}$ needed to process $Q_{\mathcal{S}}$. Surprisingly, it turns out that the achievable region of EXTN can also be written in the form of (9), where $\alpha_{\texttt{TS}}(\epsilon)$ is replaced by a term $\alpha_{\texttt{EXTN}}(N,\epsilon)$ that depends on both N and ϵ. For $N = 3, 4, 5$, these terms are as follows

$$\begin{aligned}
\alpha_{\texttt{EXTN}}(3,\epsilon) &= \frac{(1+\epsilon^2)(1+\epsilon+\epsilon^2)}{1+\epsilon+2\epsilon^2+2\epsilon^3+\epsilon^4} \\
\alpha_{\texttt{EXTN}}(4,\epsilon) &= \frac{(1+\epsilon+\epsilon^2)(1+\epsilon^2)(1+\epsilon+\epsilon^2+\epsilon^3+\epsilon^4)}{1+2\epsilon+4\epsilon^2+8\epsilon^3+7\epsilon^4+6\epsilon^5+5\epsilon^6+5\epsilon^7+\epsilon^8} \\
\alpha_{\texttt{EXTN}}(5,\epsilon) &= \frac{(1+\epsilon+\epsilon^2)(1+\epsilon^2)(1+\epsilon+\epsilon^2+\epsilon^3+\epsilon^4)(1-\epsilon+\epsilon^2)}{1+\epsilon+3\epsilon^2+9\epsilon^3-3\epsilon^4+13\epsilon^5+3\epsilon^6+12\epsilon^7-5\epsilon^8+7\epsilon^9+\epsilon^{10}}
\end{aligned} \tag{10}$$

with the expressions becoming even more cumbersome for $N \geq 6$. A plot of these terms is given in Fig. 3, which also contains the timesharing (TS) performance for comparison purposes.

Clearly, EXTN outperforms TS for all values of ϵ and is in fact asymptotically better than TS, for any finite N, as $\epsilon \to 0$ in the sense that $\alpha_{\texttt{TS}}(\epsilon) = 1 - O(\epsilon)$ and $\alpha_{\texttt{EXTN}}(N,\epsilon) = 1 - O(\epsilon^2)$, as evidenced by the different slopes in Fig. 3. At the same time, it is apparent that this performance has been achieved at the cost of significant encoding complexity (a metric of which is the number of virtual queues used). The latter statement motivates the search for a low encoding complexity algorithm that outperforms TS and still retains the $1 - O(\epsilon^2)$ performance. Since typical values for ϵ are less than 0.1, this algorithm will be a viable alternative to EXTN.

We now propose the following network coding based algorithm (hereafter referred to as ALG), which can be seen as a low complexity generalization of OPT2.

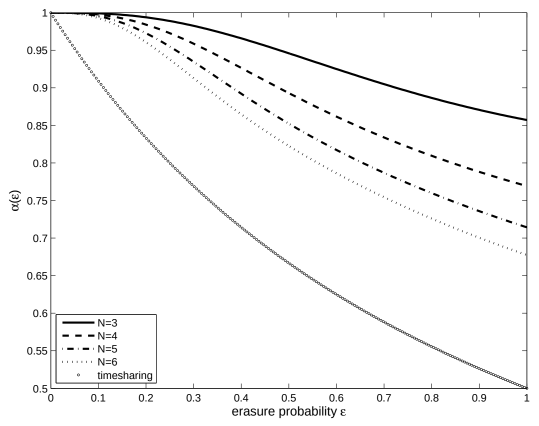

Fig. 3. EXTN performance.

Basic data structures: the transmitter maintains three virtual queues $Q_{\mathcal{G}_1}$, $Q_{\mathcal{G}_2}$, $Q_{\mathcal{G}}$ as well as non-negative integer indices $K^i_{\mathcal{S}}$ for $\mathcal{S} \in \{\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}\}$ and all $i \in \mathcal{S}$.

Initialization: the packets of set $\mathcal{K}_j$ are placed into queue $Q_{\mathcal{G}_j}$ for $j = 1, 2$, respectively, while $Q_{\mathcal{G}}$ is empty. We also set $K^i_{\mathcal{G}_j} = |\mathcal{K}_j|$ for each $i \in \mathcal{G}_j$, while $K^i_{\mathcal{G}} = 0$ for all $i \in \mathcal{G}$.

Encoding: the transmitter sequentially processes queues $Q_{\mathcal{S}}$, for $\mathcal{S} \in \{\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}\}$, in *that order*, by treating each packet as an element of field $\mathbb{F}_q$ and transmitting a linear combination $s = \sum_{p \in Q_{\mathcal{S}}} a_s(p)p$, where $a_s(p)$ are chosen randomly and uniformly in $\mathbb{F}_q$. Denote with $\mathcal{O}$ the set of users that successfully received s. Once the transmitter learns $\mathcal{O}$ through the received feedback, it performs the following actions (the actions are not mutually exclusive so all conditions should be checked):

1) for each $i \in \mathcal{S} \cap \mathcal{O}$ with $K^i_{\mathcal{S}} > 0$, index $K^i_{\mathcal{S}}$ is decreased by 1.
2) if s is erased by at least one user in $\mathcal{S}$ and received by *all* users in set $\mathcal{G} - \mathcal{S}$ (i.e. $\mathcal{O} \supseteq (\mathcal{G} - \mathcal{S})$), then packet s is added to queue $Q_{\mathcal{G}}$ and for each $i \in \mathcal{S} \cap \mathcal{O}^c$ with $K^i_{\mathcal{S}} > 0$, $K^i_{\mathcal{S}}$ is decreased by 1 while $K^i_{\mathcal{G}}$ is increased by 1.
3) if none of the above conditions are satisfied, then s is retransmitted.

Queue $Q_{\mathcal{S}}$ is processed until it holds $K^i_{\mathcal{S}} = 0$ for all $i \in \mathcal{S}$, at which point the algorithm moves to the next queue. The following property can also be proved for any t: at the beginning of slot t, all packets $p \in Q_{\mathcal{S}}$ are tokens for all users $i \in \mathcal{S}$.

Decoding: repeating the argument for $N = 2$ in Section III verbatim, it can be shown that each user $i \in \mathcal{G}$ has received K_j linearly independent tokens, with high probability, by the end of the algorithm and can solve for its unknown packets.

A. Performance analysis

Examining the 3 types of feedback-based actions in the encoding of ALG, it is clear that ALG discards a lot of side information (which explains its suboptimal nature), since, during the processing of $Q_{\mathcal{G}_1}$, it moves a packet to $Q_{\mathcal{G}}$ only if it is seen by *all* users in $\mathcal{G}_2$. This is akin to an "all or nothing" approach, which does not exploit any of the cases where the packet is received by a subset of $\mathcal{G}_2$. We now show that this crude approach still leads to better performance than TS.

As in Section III, we compute the average number of slots $T^*_{\mathcal{S}}$ required to process queue $Q_{\mathcal{S}}$, for $\mathcal{S} \in \{\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}\}$, so that the achievable rate, in information bits per transmission, is $R_j = (K_j \log_2 q)/T^*$, where $T^* = T^*_{\mathcal{G}_1} + T^*_{\mathcal{G}_2} + T^*_{\mathcal{G}}$. Denoting with $T^*_{i,\mathcal{S}}$ the (average) number of slots required, under the application of ALG, for $K^i_{\mathcal{S}}$ to become 0, it clearly follows that $T^*_{\mathcal{S}} = \max_{i \in \mathcal{S}} T^*_{i,\mathcal{S}}$. Some thought reveals that, during the processing of $Q_{\mathcal{G}_1}$, index $K^i_{\mathcal{G}_1}$ is *not* decreased if the transmitted packet s is erased by i as well at least one user in $\mathcal{G}_2$. Similarly, s is moved from $Q_{\mathcal{G}_1}$ to $Q_{\mathcal{G}}$ resulting in a decrease of $K^i_{\mathcal{G}_1}$ by 1 (and a corresponding increase of $K^i_{\mathcal{G}}$) if s is erased by i but successfully received by *all* users in $\mathcal{G}_2$. Hence,

$$T^*_{\mathcal{G}_j} = \frac{K_j}{1 - \epsilon[1 - (1 - \epsilon)^N]}, \quad j = 1, 2, \tag{11}$$

while the values of indices $K^i_{\mathcal{G}}$ at the beginning of processing $Q_{\mathcal{G}}$ (denote this time instant as $\tilde{t}_{\mathcal{G}}$) are given by

$$K^i_{\mathcal{G}}(\tilde{t}_{\mathcal{G}}) = \frac{K_j \epsilon (1-\epsilon)^N}{1-\epsilon[1-(1-\epsilon)^N]}, \quad \forall\, i \in \mathcal{G}_j. \tag{12}$$

Simple algebra now leads to the following achievable region for ALG

$$\mathcal{R}_{\texttt{ALG}} = \left\{ (R_1, R_2) \geq \mathbf{0} : \max\left(\frac{R_1}{1-\epsilon} + \frac{R_2}{\alpha_{\texttt{ALG}}(\epsilon)(1-\epsilon^2)}, \frac{R_1}{\alpha_{\texttt{ALG}}(\epsilon)(1-\epsilon^2)} + \frac{R_2}{1-\epsilon} \right) \leq \log_2|\mathcal{X}| \right\}, \tag{13}$$

where

$$\alpha_{\texttt{ALG}}(\epsilon) \triangleq 1 - \frac{\epsilon}{1+\epsilon}\left[1-(1-\epsilon)^{N-1}\right] = 1 - O(\epsilon^2). \tag{14}$$

Hence, ALG performs better than timesharing, in the sense that $\mathcal{R}_{\texttt{ALG}} \supset \mathcal{R}_{\texttt{TS}}$ (since it holds $\alpha(\epsilon) > \alpha_{\texttt{TS}}(\epsilon)$), and in fact is asymptotically better as $\epsilon \to 0$. However, a quick glance at (14) indicates that the performance of ALG becomes identical to that of TS as $N \to \infty$. A natural question now is whether this property is a result of selecting a “crude” algorithm in the first place, or whether there is a deeper result behind this. This is examined next and a partial answer is provided for a special relation between N and n.

V. Asymptotic performance as $N \to \infty$

For the reader’s convenience, we immediately state the main asymptotic result of this Section, which will be proved after some intermediate results have been established first.

Theorem 2: If N is allowed to increase as a function of n such that $N(n) = (1/\epsilon)^n w(n)$, where $w(n) = \omega(\ln n)$ (i.e. $w(n)/\ln n \to \infty$ as $n \to \infty$), then, for any $\epsilon' > 0$, there exists a sufficiently large n_0 such that for all achievable rates $(R_1, R_2) \in \mathcal{C}(N(n_0))$ it holds $R_1 + R_2 \leq (1-\epsilon)\log_2|\mathcal{X}| + 4\epsilon'$.
The Theorem essentially asserts that if N can grow with n in a certain way, timesharing essentially provides the best possible sum-rate, asymptotically as $n \to \infty$. However, it does *not* assert that timesharing is optimal as $N \to \infty$ regardless of n.

The following notation will be useful in proving the results that lead to Theorem 2. Let $Z^n_i \triangleq (Z_{i,l} : 1 \leq l \leq n)$ be the feedback sequence of user i at the end of n time slots, where $Z_{i,l} = 0$ ($Z_{i,l} = 1$) indicates that an erasure (successful reception) occurred for user i at slot l, respectively. We also denote $\boldsymbol{Z}^n_{\mathcal{I}} \triangleq (Z^n_i : i \in \mathcal{I})$, for any $\mathcal{I} \subseteq \mathcal{G}$. For brevity, we write $\boldsymbol{Z}^n$ instead of $\boldsymbol{Z}^n_{\mathcal{G}}$ and define $d(Z^n_i, Z^n_j) \triangleq \sum_{l=1}^n \mathbb{I}[Z_{i,l} = 0, Z_{j,l} = 1]$ as the number of slots where user j successfully received the transmitted packet and user i erased it. Note that $d(Z^n_i, Z^n_j) \neq d(Z^n_j, Z^n_i)$. For any $i \in \mathcal{G}_1$, we further define

$$\begin{aligned} d^*_i &\triangleq \min_{j \in \mathcal{G}_2}\; d(Z^n_i, Z^n_j), \\ j^*(i) &\triangleq \arg\min_{j \in \mathcal{G}_2}\; d(Z^n_i, Z^n_j), \end{aligned} \tag{15}$$

so that $d^*_i, j^*(i)$ are random variables that depend only on Z^n_i and $\boldsymbol{Z}^n_{\mathcal{G}_2}$. We now pick an arbitrary $i \in \mathcal{G}_1$, whence the following expression follows for *any* achievable rates R_1, R_2 (under an *arbitrary* coding scheme, according to the definitions in Section II).

$$n(R_1 + R_2) = H(W_1, W_2) = I(W_1, W_2; Y^n_i, \boldsymbol{Z}^n) + H(W_1, W_2 | Y^n_i, \boldsymbol{Z}^n). \tag{16}$$

Using the same argument as in the converse part of Shannon's theorem for feedback capacity of point-to-point channels, we find

$$\begin{aligned}
I(W_1, W_2; Y_i^n, \boldsymbol{Z}^n) &= \sum_{l=1}^{n} I(Y_{i,l}\boldsymbol{Z}_l; W_1, W_2 | Y_i^{l-1}, \boldsymbol{Z}^{l-1}) \\
&\le \sum_{l=1}^{n} \Big[H(Y_{i,l}, \boldsymbol{Z}_l) - H(Y_{i,l}, \boldsymbol{Z}_l | W_1, W_2, Y_i^{l-1}, \boldsymbol{Z}^{l-1}) \Big] \\
&\overset{(a)}{=} \sum_{l=1}^{n} \Big[H(Y_{i,l}, \boldsymbol{Z}_l) - H(Y_{i,l}, \boldsymbol{Z}_l | W_1, W_2, \boldsymbol{Y}^{l-1}, X_l) \Big] \\
&\overset{(b)}{=} \sum_{l=1}^{n} [H(Y_{i,l}, \boldsymbol{Z}_l) - H(Y_{i,l}, \boldsymbol{Z}_l | X_l)] \\
&= \sum_{l=1}^{n} [H(Y_{i,l}|\boldsymbol{Z}_l) + H(\boldsymbol{Z}_l) - H(\boldsymbol{Z}_l|X_l) - H(Y_{i,l}|X_l, \boldsymbol{Z}_l)] \\
&\overset{(c)}{=} \sum_{l=1}^{n} H(Y_{i,l}|\boldsymbol{Z}_l) = \sum_{l=1}^{n} H(Y_{i,l}|Z_{i,l}=1)\Pr(Z_{i,l}=1) \le n(1-\epsilon)\log_2|\mathcal{X}|
\end{aligned} \tag{17}$$

where (a) is due to the fact that knowledge of $(\boldsymbol{Z}^{l-1}, W_1, W_2)$ implies knowledge of $(\boldsymbol{Y}^{l-1}, X_l)$, based on the encoding function in Section II, (b) is due to the memoryless property and (c) is due to the independence of $\boldsymbol{Z}_l, X_l$, the fact that knowledge of $(X_l, \boldsymbol{Z}_l)$ implies knowledge of $Y_{i,l}$, and the fact that $Z_{i,l} = 0$ implies $Y_{i,l} = E$.

Expanding the last entropy term in (16) and using Fano's inequality also yields

$$H(W_1, W_2|Y_i^n, \boldsymbol{Z}^n) \le H(W_1|Y_i^n) + H(W_2|Y_i^n, \boldsymbol{Z}^n) \le 1 + \bar{P}_{e,1}nR_1 + H(W_2|Y_i^n, \boldsymbol{Z}^n), \tag{18}$$

where we used the decoding function $\hat{W}_i = g_i(Y_i^n)$ and defined $\bar{P}_{e,1} \triangleq \Pr(\hat{W}_i \neq W_1)$. An upper bound for the last conditional entropy term in (18) is given in the following Lemma.

Lemma 1: It holds

$$H(W_2|Y_i^n, \boldsymbol{Z}^n) \le 1 + \bar{P}_{e,N+1}nR_2 + \mathbb{E}[d_i^*]\log_2|\mathcal{X}|, \tag{19}$$

where $\bar{P}_{e,N+1} \triangleq \Pr(\hat{W}_{N+1} \neq W_2)$.

Proof: It holds

$$H(W_2|Y_i^n, \boldsymbol{Z}^n) = H(W_2|Y_i^n, \boldsymbol{Z}^n, Y_{j^*(i)}^n) + I(W_2; Y_{j^*(i)}^n | Y_i^n, \boldsymbol{Z}^n). \tag{20}$$

Since knowledge of $\boldsymbol{Z}^n$ implies knowledge of $j^*(i) \in \mathcal{G}_2$, Fano's inequality allows us to write (20) as

$$H(W_2|Y_i^n, \boldsymbol{Z}^n) \le 1 + \bar{P}_{e,j^*(i)}nR_2 + H(Y_{j^*(i)}^n|Y_i^n, \boldsymbol{Z}^n), \tag{21}$$

where $\bar{P}_{e,j^*(i)} = \Pr(\hat{W}_{j^*(i)} \neq W_2 | j^*(i))$. Since erasures among users are iid and the users cannot cooperate during decoding, we can further assume, without loss of generality, that all users in $\mathcal{G}_2$ have the same decoding function and probability of error (i.e. no user has a benefit or disadvantage over the others). This implies that $\bar{P}_{e,j^*(i)} = \bar{P}_{e,N+1} = \Pr(\hat{W}_{N+1} \neq W_2)$ so that $\bar{P}_{e,N+1}$ can be used in (21). We now apply the entropy chain rule to the last term in (21) and expand it as follows:

$$H(Y_{j^*(i)}^n|Y_i^n, \boldsymbol{Z}^n) = \sum_{t=0}^{n} \sum_{\boldsymbol{z}^n : d_i^* = t} \sum_{l=1}^{n} H(Y_{j^*(i),l}|Y_i^n, \boldsymbol{Z}^n = \boldsymbol{z}^n)\Pr(\boldsymbol{Z}^n = \boldsymbol{z}^n) \tag{22}$$

and make the following crucial observation: knowledge of $\boldsymbol{z}^n$ implies knowledge of $j^*(i)$ and for any slot l such that $z_{j^*(i),l} = 0$ it holds $Y_{j^*(i),l} = E$ so that the conditional entropy in (22) is 0 for this l. Additionally, if $z_{i,l} = z_{j^*(i),l} = 1$, then $Y_{j^*(i),l} = Y_{i,l}$ so that the conditional entropy is again 0.

Hence, the only uncertainty for $H_{j^*(i),l}$ exists when $z_{i,l} = 0$ and $z_{j^*(i),l} = 1$, whence we conclude that, for all $\boldsymbol{z}^n$ such that $d_i^* = t$, it holds

$$\sum_{l=1}^{n} H(Y_{j^*(i),l}|Y_i^n, \boldsymbol{Z}^n = \boldsymbol{z}^n) \le |\{l : z_{i,l} = 0,\ z_{j^*(i),l} = 1\}| \cdot \log_2|\mathcal{X}| = d_i^* \log_2|\mathcal{X}|. \tag{23}$$

Inserting (23) into (22) yields

$$H(Y_{j^*(i)}^n | Y_i^n, \boldsymbol{Z}^n) \leq \sum_{t=0}^{n} t \Pr(d_i^* = t) \log_2 |\mathcal{X}|, \tag{24}$$

which, combined with (21). immediately produces the desired expression. ■

An upper bound for $\mathbb{E}[d_i^*]$ is provided in the next Lemma.

Lemma 2: It holds

$$\mathbb{E}[d_i^*] = \sum_{t=1}^{n} \Pr(d_i^* \geq t) \leq n(1-\epsilon^n)^N. \tag{25}$$

Proof: The first equality in (25) is a well-known identity for non-negative random variables so we restrict our attention to the inequality in (25). It is helpful to define, for any user $i \in \mathcal{G}_1$, the set $\overline{Z_i^n}(t) \triangleq \{z^n : d(Z_i^n, z^n) \geq t\}$, which implies, for some fixed $j_0 \in \mathcal{G}_2$,

$$\Pr(d_i^* \geq t) = \Pr\left(\cap_{j \in \mathcal{G}_2} \left\{Z_j^n \in \overline{Z_i^n}(t)\right\}\right) = \sum_{z_i^n} \left(\Pr(Z_{j_0}^n \in \overline{Z_i^n}(t) | Z_i^n = z_i^n)\right)^N \Pr(Z_i^n = z_i^n), \tag{26}$$

where we used the fact that feedback sequences $Z_{j_1}^n$, $Z_{j_2}^n$ are independent for $j_1 \neq j_2$, with $j_1, j_2 \in \mathcal{G}_2$, and identically distributed.

Denoting with $E(z_i^n)$ the number of erasures for user i in the feedback sequence z_i^n, (26) can be written as

$$\Pr(d_i^* \geq t) = \sum_{k=0}^{n} \sum_{z_i^n : E(z_i^n) = k} \left(\Pr(Z_{j_0}^n \in \overline{Z_i^n}(t) | Z_i^n = z_i^n)\right)^N \Pr(Z_i^n = z_i^n), \tag{27}$$

and the definition of $\overline{Z_i^n}(t)$ allows us to bound the conditional probability in (27) as

$$\Pr(Z_{j_0}^n \in \overline{Z_i^n}(t) | Z_i^n = z_i^n) \leq \sum_{\rho=t}^{k} \binom{k}{\rho} (1-\epsilon)^\rho \epsilon^{k-\rho} \tag{28}$$

for any z_i^n such that $E(z_i^n) = k$ since, by definition of $\overline{Z_i^n}(t)$, any $Z_{j_0}^n \in \overline{Z_i^n}(t)$ must satisfy $Z_{j_0,l} = 1,\ Z_{i,l} = 0$ for at least ρ slots, where $t \leq \rho \leq k \leq n$.

The probability in (28) is upper bounded by setting $t = 1$ in the lower summation index, which implies through the binomial theorem that

$$\Pr(Z_{j_0}^n \in \overline{Z_i^n}(t) | Z_i^n = z_i^n) \leq 1 - \epsilon^k \leq 1 - \epsilon^n \tag{29}$$

Inserting (29) into (27) yields

$$\Pr(d_i^* \geq t) \leq \sum_{k=t}^{n} (1-\epsilon^n)^N \binom{n}{k} \epsilon^k (1-\epsilon)^{n-k} \leq (1-\epsilon^n)^N \tag{30}$$

since the k summation expresses a probability. Eq. (25) now follows immediately. ■

We are finally in position to prove Theorem 2.

Proof of Theorem 2: We bound each term in the RHS of (16) through (17) and (18), also applying Lemmas 1, 2, and divide by n to get

$$\begin{aligned} R_1 + R_2 &\leq (1-\epsilon)\log_2|\mathcal{X}| + \frac{2}{n} + \bar{P}_{e,1}R_1 + \bar{P}_{e,N+1}R_2 + (1-\epsilon^n)^N \log_2|\mathcal{X}| \\ &\leq (1-\epsilon)\log_2|\mathcal{X}| + \frac{2}{n} + \bar{P}_{e,1}R_1 + \bar{P}_{e,N+1}R_2 + \mathrm{e}^{-N\epsilon^n} \log_2(2N+1), \end{aligned} \tag{31}$$

for *any* R_1, R_2, where we used the fact that $(1-\xi)^N \leq \mathrm{e}^{-N\xi}$ (which follows from the well-known inequality $\ln x \leq x - 1$) and it must hold $|\mathcal{X}| > 2N$ (hence, we set $|\mathcal{X}| = 2N+1$) for the random linear network coding scheme to allow correct decoding with h.p. Due to the symmetry created by the iid erasures, instead of $\bar{P}_{e,N+1}$, which is the probability of error for user $N+1$, we could use in its place $\bar{P}_{e,j}$, for any *fixed* $j \in \mathcal{G}_2$, in (31). In this sense, we treat user $N+1$ as the "first" user in $\mathcal{G}_2$, which implies that $\bar{P}_{e,N+1}$ can be upper bounded (say, assuming optimal MAP decoding) by a quantity that depends on n but *not* N.

Hence, for any $(R_1, R_2) \in \mathcal{C}^{out}$ and any $\epsilon' > 0$, there exists some $n_0(\epsilon')$ such that $\max(2/n, \bar{P}_{e,1}R_1, \bar{P}_{e,N+1}R_2) < \epsilon'$ for all $n > n_0(\epsilon')$ *and* all N. Since $\mathcal{C}^{out} \supseteq \mathcal{C}(N(n_0(\epsilon')))$, the statement in the previous sentence also holds for all $(R_1, R_2) \in \mathcal{C}(N(n_0(\epsilon')))$. It now suffices to prove that, for this ϵ', and for any $N > N(n_0(\epsilon'))$ it holds $\mathrm{e}^{-N\epsilon^n} \log_2(2N+1) < \epsilon'$, which is equivalent to showing that $\lim_{n\to\infty} \left[\mathrm{e}^{-N(n)\epsilon^n} \log_2 N(n)\right] \overset{?}{=} 0$. The last limit can also be written as

$$\lim_{n\to\infty} \mathrm{e}^{-w(n)} \left(-n\log_2 \epsilon + \log_2 w(n)\right) \overset{?}{=} 0. \tag{32}$$

The latter condition can be easily proved from the assumption $N(n) = (1/\epsilon^n)\omega(\ln n)$ (which implies $w(n) = \omega(\ln n)$) through standard calculus and l'Hôpital rule. ■

VI. Conclusions

In this document, we considered the setting of 2 distinct multicast sessions (each destined for N different users) over a broadcast packet erasure channel with feedback and described a virtual queue-based capacity achieving scheme for $N = 2$, using insight from [1], [2]. After demonstrating the inherent complexity/performance tradeoff for $N > 2$, we proposed a low complexity coding scheme, for arbitrary N, that only operates on 3 queues and still outperforms timesharing for any finite N. Our findings have led us to conjecture that timesharing becomes asymptotically optimal as $N \to \infty$. We were able to provide a partial affirmative result for this conjecture by considering a special relation between N, n. However, the conjecture remains an open problem if $N \to \infty$ independently of n. The latter case may be pursued in the future.

Appendix

A. *Performance analysis of* `OPT2` *for independent non-identical channels*

We first need to slightly generalize the notation. We retain the notation and meaning of $\mathcal{G}_1$, $\mathcal{G}_2$, $\mathcal{G}$, $T^*_{i,\mathcal{S}}$, $T^*_{\mathcal{S}}$, $K^i_{\mathcal{S}}$, $\alpha(\mathcal{S})$ for $N = 2$, as well as the time instants $\hat{t}_1$, $\hat{t}_2$ corresponding to the end of phase 1, 2, respectively. We further denote with ϵ_i the probability that a transmitted symbol is erased by user i, while $\epsilon_{\mathcal{H}}$ denotes the probability that a symbol is erased by *all* users in set $\mathcal{H}$, i.e. $\epsilon_{\mathcal{H}} = \prod_{i\in\mathcal{H}} \epsilon_i$ (due to the independence assumption). Without loss of generality, we assume that $\epsilon_1 \geq \epsilon_2$ and $\epsilon_3 \geq \epsilon_4$. These two inequalities imply that the outer capacity bound $\mathcal{C}(1)$ in (1) can be written as

$$\begin{aligned}\mathcal{C}^{out} = \mathcal{C}(1) &= \left\{(R_1, R_2) \geq \mathbf{0} : \max_{i\in\mathcal{G}_1, j\in\mathcal{G}_2} \max\left(\frac{R_1}{1-\epsilon_i} + \frac{R_2}{1-\epsilon_{\{i,j\}}}, \frac{R_1}{1-\epsilon_{\{i,j\}}} + \frac{R_2}{1-\epsilon_j}\right) \leq 1\right\} \\ &= \left\{(R_1, R_2) \geq \mathbf{0} : \max\left(\frac{R_1}{1-\epsilon_1} + \frac{R_2}{1-\epsilon_{\{1,3\}}}, \frac{R_1}{1-\epsilon_3} + \frac{R_2}{1-\epsilon_{\{1,3\}}}\right) \leq 1\right\}.\end{aligned} \tag{33}$$

Following along similar lines as in Section III-C, the trick is to show that, at the end of phase 2, at least one of the $K^i_{\mathcal{G}}$ indices is non-zero. Specifically, we will prove the equivalent statement

$$K^1_{\mathcal{G}}(\hat{t}_2) + K^3_{\mathcal{G}}(\hat{t}_2) > 0 \quad \forall\, \epsilon_1, \epsilon_3 \in (0,1), \tag{34}$$

which implies that at least one of $K^1_{\mathcal{G}}(\hat{t}_2)$, $K^3_{\mathcal{G}}(\hat{t}_2)$ is positive. We concentrate on the case $0 < \epsilon_1, \epsilon_3 < 1$, since the cases where at least one of ϵ_1, ϵ_3 is either 0 or 1 lead to trivial results.

We now compute the average number of slots required for the processing of each queue $Q_{\mathcal{S}}$ by modifying the expressions in Section III-C to account for the non-identical erasure probabilities. Starting with $Q_{\{1,2\}}$, it is clear that index $K^i_{\{1,2\}}$, for $i \in \{1,2\}$, is not decreased only if the transmitted symbol is erased by user i as well as *both* users 3,4. Hence, it holds

$$\begin{aligned} T^*_{1,\{1,2\}} &= \frac{K_1}{1-\epsilon_{\{1,3,4\}}} = \frac{K_1}{1-\epsilon_1\epsilon_{\{3,4\}}}, \\ T^*_{2,\{1,2\}} &= \frac{K_1}{1-\epsilon_{\{2,3,4\}}} = \frac{K_1}{1-\epsilon_2\epsilon_{\{3,4\}}}, \end{aligned} \tag{35}$$

and, since $\epsilon_1 \geq \epsilon_2$, we conclude that

$$T^*_{\{1,2\}} = \max_{i\in\{1,2\}} T^*_{i,\{1,2\}} = \frac{K_1}{1-\epsilon_1\epsilon_{\{3,4\}}}. \tag{36}$$

An analogous treatment for $Q_{\{3,4\}}$ yields

$$\begin{aligned} T^*_{3,\{3,4\}} &= \frac{K_2}{1-\epsilon_{\{1,2,3\}}} = \frac{K_2}{1-\epsilon_3\epsilon_{\{1,2\}}}, \\ T^*_{4,\{3,4\}} &= \frac{K_2}{1-\epsilon_{\{1,2,4\}}} = \frac{K_2}{1-\epsilon_4\epsilon_{\{1,2\}}}, \\ T^*_{\{3,4\}} &= \max_{j\in\{3,4\}} T^*_{j,\{3,4\}} = \frac{K_2}{1-\epsilon_3\epsilon_{\{1,2\}}}, \end{aligned} \tag{37}$$

the last expression following from the assumption $\epsilon_3 \geq \epsilon_4$.

The actions performed in step 2 of phase 1 lead to the increase of indices $K^i_{\mathcal{S}}$, for $|\mathcal{S}| \geq 3$. For example, $K^1_{\{1,2,3\}}$ increases whenever a symbol is erased by users 1,4 but received by user 3. A careful examination of the conditions in step 2 leads to the following expressions

$$\begin{aligned} K^1_{\{1,2,3\}}(\hat{t}_1) &= T^*_{1,\{1,2\}} p_{\{3\},\{1,4\}} = T^*_{1,\{1,2\}}\epsilon_{14}(1-\epsilon_3), \\ K^2_{\{1,2,3\}}(\hat{t}_1) &= T^*_{2,\{1,2\}} p_{\{3\},\{2,4\}} = T^*_{2,\{1,2\}}\epsilon_{24}(1-\epsilon_3), \end{aligned} \tag{38}$$

$$\begin{aligned} K^1_{\{1,2,4\}}(\hat{t}_1) &= T^*_{1,\{1,2\}} p_{\{4\},\{1,3\}} = T^*_{1,\{1,2\}}\epsilon_{13}(1-\epsilon_4), \\ K^2_{\{1,2,4\}}(\hat{t}_1) &= T^*_{2,\{1,2\}} p_{\{4\},\{2,3\}} = T^*_{2,\{1,2\}}\epsilon_{23}(1-\epsilon_4), \end{aligned} \tag{39}$$

$$\begin{aligned} K^1_{\{1,2,3,4\}}(\hat{t}_1) &= T^*_{1,\{1,2\}} p_{\{3,4\},\{1\}} = T^*_{1,\{1,2\}}\epsilon_1(1-\epsilon_3)(1-\epsilon_4), \\ K^2_{\{1,2,3,4\}}(\hat{t}_1) &= T^*_{2,\{1,2\}} p_{\{3,4\},\{2\}} = T^*_{2,\{1,2\}}\epsilon_2(1-\epsilon_3)(1-\epsilon_4), \end{aligned} \tag{40}$$

$$\begin{aligned} K^3_{\{1,3,4\}}(\hat{t}_1) &= T^*_{3,\{3,4\}} p_{\{1\},\{2,3\}} = T^*_{3,\{3,4\}}\epsilon_{23}(1-\epsilon_1), \\ K^4_{\{1,3,4\}}(\hat{t}_1) &= T^*_{4,\{3,4\}} p_{\{1\},\{2,4\}} = T^*_{4,\{3,4\}}\epsilon_{24}(1-\epsilon_1), \end{aligned} \tag{41}$$

$$\begin{aligned} K^3_{\{2,3,4\}}(\hat{t}_1) &= T^*_{3,\{3,4\}} p_{\{2\},\{1,3\}} = T^*_{3,\{3,4\}}\epsilon_{13}(1-\epsilon_2), \\ K^4_{\{2,3,4\}}(\hat{t}_1) &= T^*_{4,\{3,4\}} p_{\{2\},\{1,4\}} = T^*_{4,\{3,4\}}\epsilon_{14}(1-\epsilon_2), \end{aligned} \tag{42}$$

$$\begin{aligned} K^3_{\{1,2,3,4\}}(\hat{t}_1) &= T^*_{3,\{3,4\}} p_{\{1,2\},\{3\}} = T^*_{3,\{3,4\}}\epsilon_3(1-\epsilon_1)(1-\epsilon_2), \\ K^4_{\{1,2,3,4\}}(\hat{t}_1) &= T^*_{4,\{3,4\}} p_{\{1,2\},\{4\}} = T^*_{4,\{3,4\}}\epsilon_4(1-\epsilon_1)(1-\epsilon_2). \end{aligned} \tag{43}$$

In phase 2, $K^i_{\mathcal{S}}$, with $|\mathcal{S}| = 3$, is not decreased by one only if the transmitted packet is erased by users i and $\alpha(\mathcal{S})$. Hence, it holds

$$\begin{aligned} T^*_{1,\{1,2,3\}} &= \frac{K^1_{\{1,2,3\}}(\hat{t}_1)}{1-\epsilon_{14}} = \frac{K_1}{1-\epsilon_1\epsilon_{34}}\,\frac{\epsilon_1\epsilon_4(1-\epsilon_3)}{1-\epsilon_1\epsilon_4}, \\ T^*_{2,\{1,2,3\}} &= \frac{K^2_{\{1,2,3\}}(\hat{t}_1)}{1-\epsilon_{24}} = \frac{K_1}{1-\epsilon_2\epsilon_{34}}\,\frac{\epsilon_2\epsilon_4(1-\epsilon_3)}{1-\epsilon_2\epsilon_4}, \\ T^*_{\{1,2,3\}} &= \max_{i\in\{1,2\}} T^*_{1,\{1,2,3\}} = T^*_{1,\{1,2,3\}}, \end{aligned} \tag{44}$$

where the last expression follows from the assumption $\epsilon_1 \geq \epsilon_2$ and the fact that the function $\frac{x}{(1-ax)(1-bx)}$ is strictly increasing for $x > 0$ and $0 < a, b < 1$. For the other queues, similar reasoning yields

$$\begin{aligned} T^*_{1,\{1,2,3\}} &= \frac{K^1_{\{1,2,3\}}(\hat{t}_1)}{1-\epsilon_{14}} = \frac{K_1}{1-\epsilon_1\epsilon_{34}}\,\frac{\epsilon_1\epsilon_4(1-\epsilon_3)}{1-\epsilon_1\epsilon_4}, \\ T^*_{2,\{1,2,3\}} &= \frac{K^2_{\{1,2,3\}}(\hat{t}_1)}{1-\epsilon_{24}} = \frac{K_1}{1-\epsilon_2\epsilon_{34}}\,\frac{\epsilon_2\epsilon_4(1-\epsilon_3)}{1-\epsilon_2\epsilon_4}, \\ T^*_{\{1,2,3\}} &= \max_{i\in\{1,2\}} T^*_{1,\{1,2,3\}} = T^*_{1,\{1,2,3\}}. \end{aligned} \tag{45}$$

$$\begin{aligned} T^*_{1,\{1,2,4\}} &= \frac{K^1_{\{1,2,4\}}(\hat{t}_1)}{1-\epsilon_{13}} = \frac{K_1}{1-\epsilon_1\epsilon_{34}} \frac{\epsilon_1\epsilon_3(1-\epsilon_4)}{1-\epsilon_1\epsilon_3}, \\ T^*_{2,\{1,2,4\}} &= \frac{K^2_{\{1,2,4\}}(\hat{t}_1)}{1-\epsilon_{23}} = \frac{K_1}{1-\epsilon_2\epsilon_{34}} \frac{\epsilon_2\epsilon_3(1-\epsilon_4)}{1-\epsilon_2\epsilon_3}, \\ T^*_{\{1,2,4\}} &= \max_{i\in\{1,2\}} T^*_{1,\{1,2,4\}} = T^*_{1,\{1,2,4\}}. \end{aligned} \tag{46}$$

$$\begin{aligned} T^*_{3,\{1,3,4\}} &= \frac{K^3_{\{1,3,4\}}(\hat{t}_1)}{1-\epsilon_{23}} = \frac{K_2}{1-\epsilon_{12}\epsilon_3} \frac{\epsilon_2\epsilon_3(1-\epsilon_1)}{1-\epsilon_2\epsilon_3}, \\ T^*_{4,\{1,3,4\}} &= \frac{K^4_{\{1,3,4\}}(\hat{t}_1)}{1-\epsilon_{24}} = \frac{K_2}{1-\epsilon_{12}\epsilon_4} \frac{\epsilon_2\epsilon_4(1-\epsilon_1)}{1-\epsilon_2\epsilon_4}, \\ T^*_{\{1,3,4\}} &= \max_{i\in\{3,4\}} T^*_{i,\{1,3,4\}} = T^*_{3,\{1,3,4\}}. \end{aligned} \tag{47}$$

$$\begin{aligned} T^*_{3,\{2,3,4\}} &= \frac{K^3_{\{2,3,4\}}(\hat{t}_1)}{1-\epsilon_{13}} = \frac{K_2}{1-\epsilon_{12}\epsilon_3} \frac{\epsilon_1\epsilon_3(1-\epsilon_2)}{1-\epsilon_1\epsilon_3}, \\ T^*_{4,\{2,3,4\}} &= \frac{K^4_{\{2,3,4\}}(\hat{t}_1)}{1-\epsilon_{14}} = \frac{K_2}{1-\epsilon_{12}\epsilon_4} \frac{\epsilon_1\epsilon_4(1-\epsilon_2)}{1-\epsilon_1\epsilon_4}, \\ T^*_{\{2,3,4\}} &= \max_{i\in\{3,4\}} T^*_{i,\{2,3,4\}} = T^*_{3,\{1,3,4\}}. \end{aligned} \tag{48}$$

The packet movements performed in phase 2 imply that, at the end of phase 2 (time instant $\hat{t}_2$) it holds $K^i_{\{1,2,3,4\}} = [\tilde{K}^i_{\{1,2,3,4\}}]^+$, for $i \in \{1,2,3,4\}$, where

$$\begin{aligned} \tilde{K}^1_{\{1,2,3,4\}} = K^1_{\{1,2,3,4\}}(\hat{t}_1) &+ T^*_{1,\{1,2,3\}}(1-\epsilon_4)\epsilon_1 - \left(T^*_{\{1,2,3\}} - T^*_{1,\{1,2,3\}}\right)(1-\epsilon_1) \\ &+ T^*_{1,\{1,2,4\}}(1-\epsilon_3)\epsilon_1 - \left(T^*_{\{1,2,4\}} - T^*_{1,\{1,2,4\}}\right)(1-\epsilon_1) - T^*_{\{1,3,4\}}(1-\epsilon_1), \end{aligned} \tag{49}$$

$$\begin{aligned} \tilde{K}^2_{\{1,2,3,4\}} = K^2_{\{1,2,3,4\}}(\hat{t}_1) &+ T^*_{2,\{1,2,3\}}(1-\epsilon_4)\epsilon_2 - \left(T^*_{\{1,2,3\}} - T^*_{2,\{1,2,3\}}\right)(1-\epsilon_2) \\ &+ T^*_{2,\{1,2,4\}}(1-\epsilon_3)\epsilon_2 - \left(T^*_{\{1,2,4\}} - T^*_{2,\{1,2,4\}}\right)(1-\epsilon_2) - T^*_{\{2,3,4\}}(1-\epsilon_2), \end{aligned} \tag{50}$$

$$\begin{aligned} \tilde{K}^3_{\{1,2,3,4\}} = K^3_{\{1,2,3,4\}}(\hat{t}_1) &+ T^*_{3,\{1,3,4\}}(1-\epsilon_2)\epsilon_3 - \left(T^*_{\{1,3,4\}} - T^*_{3,\{1,3,4\}}\right)(1-\epsilon_3) \\ &+ T^*_{3,\{2,3,4\}}(1-\epsilon_1)\epsilon_3 - \left(T^*_{\{2,3,4\}} - T^*_{3,\{2,3,4\}}\right)(1-\epsilon_3) - T^*_{\{1,2,3\}}(1-\epsilon_3), \end{aligned} \tag{51}$$

$$\begin{aligned} \tilde{K}^4_{\{1,2,3,4\}} = K^4_{\{1,2,3,4\}}(\hat{t}_1) &+ T^*_{4,\{1,3,4\}}(1-\epsilon_2)\epsilon_4 - \left(T^*_{\{1,3,4\}} - T^*_{4,\{1,3,4\}}\right)(1-\epsilon_4) \\ &+ T^*_{4,\{2,3,4\}}(1-\epsilon_1)\epsilon_4 - \left(T^*_{\{2,3,4\}} - T^*_{4,\{2,3,4\}}\right)(1-\epsilon_4) - T^*_{\{1,2,4\}}(1-\epsilon_3). \end{aligned} \tag{52}$$

The above expressions are the generalization of (5) to the case of non-identical erasure probabilities.

At this point, it is most convenient to perform the algebra using a symbol manipulation program such as Maple to compute

$$\begin{aligned} \tilde{K}^1_{\{1,2,3,4\}} + \tilde{K}^3_{\{1,2,3,4\}} &= \frac{\epsilon_1(1-\epsilon_3)}{(1-\epsilon_{14}\epsilon_3)(1-\epsilon_{14})}(1-2\epsilon_4+\epsilon_3\epsilon_4)K_1 \\ &+ \frac{\epsilon_3(1-\epsilon_1)}{(1-\epsilon_{23}\epsilon_1)(1-\epsilon_{23})}(1-2\epsilon_2+\epsilon_1\epsilon_2)K_2 + \text{ non-negative terms.} \end{aligned} \tag{53}$$

The assumptions $\epsilon_1 \geq \epsilon_2$, $\epsilon_3 \geq \epsilon_4$ imply $1-2\epsilon_4+\epsilon_3\epsilon_4 \geq (1-\epsilon_4)^2$ and $1-2\epsilon_2+\epsilon_1\epsilon_2 \geq (1-\epsilon_2)^2$ which, combined with the fact $0 < \epsilon_1 < 1$, leads us to conclude that

$$\tilde{K}^1_{\{1,2,3,4\}} + \tilde{K}^3_{\{1,2,3,4\}} > 0 \Leftrightarrow \text{at least one of } K^1_{\{1,2,3,4\}}(\hat{t}_2),\, K^3_{\{1,2,3,4\}}(\hat{t}_2) > 0. \tag{54}$$

It is now a matter of simple algebra (performed by Maple) to compute $\sum_{\mathcal{S}} T^*_{\mathcal{S}}$ as functions of K_1, K_2 (and channel statistics) and eliminate the K_1, K_2 terms to show that OPT2 achieves the following region

$$\begin{aligned}\mathcal{R}_{\texttt{OPT2}} = \Bigg\{(R_1,R_2) \geq \mathbf{0} : \max\Bigg(&\frac{R_1}{1-\epsilon_1} + \frac{R_2}{1-\epsilon_{\{1,3\}}}, \frac{R_1}{1-\epsilon_{\{1,3\}}} + \frac{R_2}{1-\epsilon_3},\\ &\frac{AR_1}{1-\epsilon_2} + \frac{R_2}{1-\epsilon_{\{2,3\}}}, \frac{R_1}{1-\epsilon_{\{1,4\}}} + \frac{BR_2}{1-\epsilon_4}\Bigg) \leq 1\Bigg\},\end{aligned} \tag{55}$$

where

$$\begin{aligned} A &= 1 + \frac{\epsilon_{\{3,4\}}(1-\epsilon_2)(\epsilon_1-\epsilon_2)}{(1-\epsilon_1\epsilon_{\{3,4\}})(1-\epsilon_2\epsilon_{\{3,4\}})},\\ B &= 1 + \frac{\epsilon_{\{1,2\}}(1-\epsilon_4)(\epsilon_3-\epsilon_4)}{(1-\epsilon_3\epsilon_{\{1,2\}})(1-\epsilon_4\epsilon_{\{1,2\}})}.\end{aligned} \tag{56}$$

A simple geometrical argument regarding lines on a plane reveals that, for $\epsilon_1 \geq \epsilon_2$ and $\epsilon_3 \geq \epsilon_4$, the terms in the second line of (55) are dominated by the terms in the first line (so that $\mathcal{R}_{\texttt{OPT2}} = \mathcal{C}(1)$, which is the desired optimality result) if it holds

$$\begin{aligned} \frac{1-\epsilon_2}{A} &\overset{?}{\geq} 1-\epsilon_1,\\ \frac{1-\epsilon_4}{B} &\overset{?}{\geq} 1-\epsilon_3.\end{aligned} \tag{57}$$

The last two inequalities are easily verified through Maple to be true.

References

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

[2] M. Gatzianas, S. Saeedi, and C. Fragouli, "Feedback-based coding algorithms for broadcast erasure channels with degraded message sets," in *Proc. International Symposium on Network Coding (NetCod)*, June 2012.

[3] C. Fragouli and E. Soljanin, *Network coding fundamentals*. NOW Publishers, 2007.

[4] C.-C. Wang, "Capacity of 1–to-K broadcast packet erasure channels with channel output feedback," in *Proc. 48th Annual Allerton Conference*, October 2010. [Online]. Available: http://arxiv.org/abs/1010.2436v1

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

[6] J. Körner and K. Marton, "General broadcast channels with degraded message sets," *IEEE Trans. Inform. Theory*, vol. 23, no. 1, pp. 60–64, January 1977.

[7] C.-C. Wang, "Capacity region of two symmetric nearby erasure channels with channel state feedback," in *Proc. Information Theory Workshop (ITW)*, September 2012.

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

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

[10] 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.