Broadcast erasure channel with feedback: the two multicast case — algorithms and bounds

Efe Onaran*, Marios Gatzianas†‡ and Christina Fragouli†
† School of Computer and Communication Sciences, EPFL, Switzerland.
* Department of Electrical and Electronics Engineering, Bilkent University, Ankara, Turkey.

***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 require knowledge of channel statistics and achieves capacity for $N = 2$ and iid erasures. Since the extension of this algorithm to $N > 2$ is not straightforward, we describe a simple 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 paper 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 broadcast erasure packet erasure channel (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 paper 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.

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 it.
- 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 $N > 2$ is an open problem.
- 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.

Due to space restrictions, the proofs of some results are omitted and presented in [4] instead.

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 [1], [5] (where it is shown that feedback can increase the capacity region) and the case of multiple sources/multiple destinations in a directed acyclic graph [6], 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 our paper cannot be cast into the setting of [6]. Furthermore, the two message sets in our paper are non-degraded, so that we cannot invoke results from relevant literature on degraded messages [7] (most of which does not take feedback into account in the first place).

‡ This work was supported by the ERC Starting Grant Project NOWIRE ERC-2009-StG-240317.

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 [8] 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 [9], which considers broadcast messages with stochastic arrivals. The difference between the last two works and the current paper 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.

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. Hence, all users in $\mathcal{G}_1 \triangleq \{1,\ldots,N\}$ should receive message W_1 while all users in $\mathcal{G}_2 \triangleq \{N+1,\ldots,2N\}$ 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 [6], since each group has a distinct multicast session, and cannot be converted into a setting compatible with [6] 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 (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 (this is a natural extension of the standard definitions in [10] 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 $\boldsymbol{Y}^{l-1} \triangleq (\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_l)$. 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. This justifies the specific selection for the encoding function.
- $2N$ 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 $Y_i^n \triangleq (Y_{i,1}, \ldots, Y_{i,n})$ 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 $\lambda_n(\boldsymbol{W}) = \Pr(\cup_{i\in\mathcal{G}_1}\{g_i(Y_i^n) \neq W_1\} \cup \cup_{i\in\mathcal{G}_2}\{g_i(Y_i^n) \neq W_2\} | \boldsymbol{W})$ while the rate for this code, in information bits per transmitted symbol, is $\boldsymbol{R} = (R_1, R_2)$, where $R_j = (\log_2 M_j)/n$. Then, $\boldsymbol{R}$ is achievable if there exists a sequence of $(\lceil 2^{nR_1} \rceil, \lceil 2^{nR_2} \rceil, n)$ codes such that $\bar{P}_e = \frac{1}{|\mathcal{W}|}\sum_{\boldsymbol{W}\in\mathcal{W}} \lambda_n(\boldsymbol{W}) \to 0$ as $n \to \infty$. 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 has been determined in [11] as follows

$$\mathcal{C}(1) = \left\{ (R_1, R_2) : \max_{\pi\in\mathcal{P}} \left(\frac{R_{\pi(1)}}{1-\epsilon} + \frac{R_{\pi(2)}}{1-\epsilon^2} \right) \leq \log_2 |\mathcal{X}| \right\}, \tag{1}$$

where R_1, R_2 are measured in bits per information symbol, $\mathcal{P}$ is the set of permutations π on $\{1, 2\}$ 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 [11] 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, we now show the following result (full proof is given in [4]; only the algorithm and intuition are presented here).

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 the algorithm `OPT2` described below.

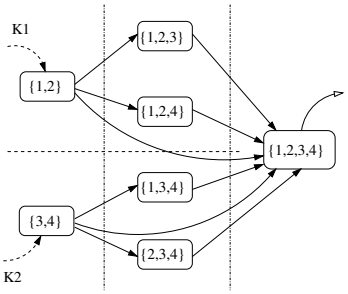

Fig. 1. Queue structure for OPT2. Ovals denote queues, the sets inside the ovals denote the $\mathcal{S}$ corresponding to $Q_{\mathcal{S}}$ and lines with arrows indicate possible index transition under the proposed algorithm.

Capacity achieving 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. III 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. III, 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, 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* [1], [2] will be useful.

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 $\mathbb{F}_q$.

In all 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, 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):

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 of checking the previous conditions and taking suitable actions. The feedback column contains the erasure events (the presence/absence of a bar above a number denotes a successful reception/erasure for that user) while the action column lists the appropriate actions/steps (the number after S. denotes the corresponding step of phase 1). Clearly, different steps may be taken for different users.

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)

Phase 2: each queue $Q_{\mathcal{S}}$ in level 3 is individually combined with $Q_{\mathcal{G}}$ in level 4 (this is still considered as "processing $Q_{\mathcal{S}}$") 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 $\mathcal{S} \cap \tilde{\mathcal{G}}^c_{\mathcal{S}}$. 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 $K_{\mathcal{S}}^i > 0$, then $K_{\mathcal{S}}^i$ is decreased by one.
   b) if $K_{\mathcal{S}}^i = 0$ and $K_{\mathcal{G}}^i > 0$, then $K_{\mathcal{G}}^i$ 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 $K_{\mathcal{S}}^i > 0$, $K_{\mathcal{S}}^i$ is decreased by one and $K_{\mathcal{G}}^i$ is increased by one.
3) if user $\alpha(\mathcal{S})$ received s and $K_{\mathcal{G}}^{\alpha(\mathcal{S})} > 0$, then $K_{\mathcal{G}}^{\alpha(\mathcal{S})}$ 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 $K_{\mathcal{S}_1}^i$ and $K_{\mathcal{G}}^i$ are equal to zero. At this point, a queue $Q_{\mathcal{S}_2}$ is selected such that $i \in \mathcal{S}_2$ and $K_{\mathcal{S}_2}^i > 0$ (so that it is possible to increase $K_{\mathcal{G}}^i$ 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 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.

A. 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.

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 (similarly to [1]), for any t: all packets stored in $Q_{\mathcal{S}}$ at the beginning of slot t are tokens for all $i \in \mathcal{S}$; hence, $K_{\mathcal{S}}^i$ 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 linear combinations from $Q_{\mathcal{S}}$, $K_{\mathcal{S}}^i$ 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 $\mathcal{S} = \{1, 2, 3\}$), 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. The reader is referred to [4] for more details on the intuition behind each step in phases 1–3 as well as the corresponding performance analysis leading to Theorem 1.

IV. A low complexity algorithm for $N > 2$

Generalizing the previous algorithm to higher N is not straightforward since the number of virtual queues, as well as the possible ways of efficiently selecting queues for cross-level combining, increases exponentially. However, we provide next a simple (suboptimal) algorithm, named ALG, that operates on only 3 queues (for arbitrary N) and outperforms a baseline timesharing (TS) scheme. Obviously, if an unbounded number of queues is allowed at the transmitter, better algorithms than ALG can be constructed, as described in [4].

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 $\mathcal{R}_{\text{TS}}$ of the TS scheme is

$$\mathcal{R}_{\text{TS}} = \{(R_1, R_2) \geq \mathbf{0} : R_1 + R_2 \leq (1 - \epsilon) \log_2 |\mathcal{X}|\}, \quad (2)$$

so that we aim in constructing codes which achieve a rate region that is a superset of $\mathcal{R}_{\text{TS}}$. We now propose ALG as a low complexity generalization of the algorithm in Section III.

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 $K_{\mathcal{S}}^i$ 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 $K_{\mathcal{G}_j}^i = |\mathcal{K}_j|$ for each $i \in \mathcal{G}_j$, while $K_{\mathcal{G}}^i = 0$ 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 $K_{\mathcal{S}}^i > 0$, index $K_{\mathcal{S}}^i$ 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 $K_{\mathcal{S}}^i>0$, $K_{\mathcal{S}}^i$ is decreased by 1 while $K_{\mathcal{G}}^i$ 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 $K_{\mathcal{S}}^i=0$ for all $i\in\mathcal{S}$, at which point the algorithm moves to the next queue. The following property can again be proved for any t: at the beginning of slot t, all packets $p\in Q_{\mathcal{S}}$ are tokens for all users $i\in\mathcal{S}$. The interpretation of $K_{\mathcal{S}}^i$ and the feedback-based actions is similar to that of Section III.

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 $\mathcal{G}_2$. 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 $K_{\mathcal{S}}^i$ 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 $K_{\mathcal{G}_1}^i$ 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 $K_{\mathcal{G}_1}^i$ by 1 (and a corresponding increase of $K_{\mathcal{G}}^i$) if s is erased by i but successfully received by *all* users in $\mathcal{G}_2$. Hence,

$$T_{\mathcal{G}_j}^*=\frac{K_j}{1-\epsilon[1-(1-\epsilon)^N]},\quad j=1,2, \tag{3}$$

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

$$K_{\mathcal{G}}^i(\tilde{t}_{\mathcal{G}})=\frac{K_j\epsilon(1-\epsilon)^N}{1-\epsilon[1-(1-\epsilon)^N]},\quad \forall\, i\in\mathcal{G}_j. \tag{4}$$

Simple algebra now leads to the following achievable region $\hat{\mathcal{R}}$ for ALG

$$\hat{\mathcal{R}}=\left\{(R_1,R_2):\max_{\pi\in\mathcal{P}}\left(\frac{R_{\pi(1)}}{1-\epsilon}+\frac{R_{\pi(2)}}{\alpha(\epsilon)(1-\epsilon^2)}\right)\le\log_2|\mathcal{X}|\right\}, \tag{5}$$

where $\alpha(\epsilon)\triangleq 1-\frac{\epsilon}{1+\epsilon}\left[1-(1-\epsilon)^{N-1}\right]=1-O(\epsilon^2)$. Comparing with the achievable region in (2) of the timesharing scheme, we see that $\mathcal{R}_{\text{TS}}$ can also be written in the form of (5) by setting $\alpha_{\text{TS}}(\epsilon)=\frac{1}{1+\epsilon}=1-\frac{\epsilon}{1+\epsilon}$.

Hence, ALG performs better than timesharing, in the sense that $\hat{\mathcal{R}}\supset\mathcal{R}_{\text{TS}}$ (since it holds $\alpha(\epsilon)>\alpha_{\text{TS}}(\epsilon)$), and in fact is asymptotically better as $\epsilon\to 0$ since $\alpha_{\text{TS}}(\epsilon)=1-O(\epsilon)$. However, it is clear that the performance of ALG becomes identical to that of TS as $N\to\infty$, i.e. $\alpha(\epsilon)\to\alpha_{\text{TS}}(\epsilon)$ for all ϵ 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 rates $(R_1,R_2)\in\mathcal{C}(N(n_0))$ it holds $R_1+R_2\le(1-\epsilon)\log_2|\mathcal{X}|+4\epsilon'$.

The Theorem essentially states 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 $Z_i^n\triangleq(Z_{i,l}:1\le l\le n)$ 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 $\boldsymbol{Z}_{\mathcal{I}}^n\triangleq(Z_i^n:i\in\mathcal{I})$, for any $\mathcal{I}\subseteq\mathcal{G}$. For brevity, we write $\boldsymbol{Z}^n$ instead of $\boldsymbol{Z}_{\mathcal{G}}^n$ and define $d(Z_i^n,Z_j^n)\triangleq\sum_{l=1}^n\mathbb{I}[Z_{i,l}=0,Z_{j,l}=1]$ as the number of slots where user j successfully received the transmitted packet and user i erased it. Note that $d(Z_i^n,Z_j^n)\neq d(Z_j^n,Z_i^n)$. For any $i\in\mathcal{G}_1$, we further define $d_i^*\triangleq\min_{j\in\mathcal{G}_2}\ d(Z_i^n,Z_j^n)$ and $j^*(i)\triangleq\arg\min_{j\in\mathcal{G}_2}\ d(Z_i^n,Z_j^n)$, so that $d_i^*,j^*(i)$ are random variables that depend only on Z_i^n and $\boldsymbol{Z}_{\mathcal{G}_2}^n$. 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).

$$\begin{aligned}n(R_1+R_2)=H(W_1,W_2)&=I(W_1,W_2;Y_i^n,\boldsymbol{Z}^n)\\&\quad+H(W_1,W_2|Y_i^n,\boldsymbol{Z}^n).\end{aligned} \tag{6}$$

Using the same argument as in the converse part of Shannon's theorem for feedback capacity of point-to-point channels, we find [4]

$$I(W_1,W_2;Y_i^n,\boldsymbol{Z}^n)\le n(1-\epsilon)\log_2|\mathcal{X}|. \tag{7}$$

Expanding the last entropy term in (6) and using Fano's inequality also yields

$$\begin{aligned}&H(W_1,W_2|Y_i^n,\boldsymbol{Z}^n)\le H(W_1|Y_i^n)+H(W_2|Y_i^n,\boldsymbol{Z}^n)\\&\le 1+\bar{P}_{e,1}nR_1+H(W_2|Y_i^n,\boldsymbol{Z}^n),\end{aligned} \tag{8}$$

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 (8) is given in the following Lemma.

Lemma 1: It holds

$$H(W_2|Y_i^n, \boldsymbol{Z}^n) \leq 1 + \bar{P}_{e,N+1} n R_2 + \mathbb{E}[d_i^*] \log_2|\mathcal{X}|, \quad (9)$$

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

Proof: It holds

$$\begin{aligned} H(W_2|Y_i^n, \boldsymbol{Z}^n) = H(W_2|Y_i^n, \boldsymbol{Z}^n, Y_{j^*(i)}^n) \\ + I(W_2; Y_{j^*(i)}^n | Y_i^n, \boldsymbol{Z}^n). \end{aligned} \quad (10)$$

Since knowledge of $\boldsymbol{Z}^n$ implies knowledge of $j^*(i) \in \mathcal{G}_2$, Fano's inequality allows us to write (10) as

$$H(W_2|Y_i^n, \boldsymbol{Z}^n) \leq 1 + \bar{P}_{e,j^*(i)} n R_2 + H(Y_{j^*(i)}^n | Y_i^n, \boldsymbol{Z}^n), \quad (11)$$

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 (11). We now apply the entropy chain rule to the last term in (11) and expand it as follows:

$$\begin{aligned} &H(Y_{j^*(i)}^n | Y_i^n, \boldsymbol{Z}^n) \\ &= \sum_{t=0}^{n} \sum_{\boldsymbol{z}^n : d_i^* = t} \sum_{l=1}^{n} H(Y_{j^*(i),l} | Y_i^n, \boldsymbol{Z}^n = \boldsymbol{z}^n) \Pr(\boldsymbol{Z}^n = \boldsymbol{z}^n), \end{aligned} \quad (12)$$

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 (12) 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

$$\begin{aligned} &\sum_{l=1}^{n} H(Y_{j^*(i),l} | Y_i^n, \boldsymbol{Z}^n = \boldsymbol{z}^n) \\ &\leq |\{l : z_{i,l} = 0,\ z_{j^*(i),l} = 1\}| \cdot \log_2|\mathcal{X}| = d_i^* \log_2|\mathcal{X}|. \end{aligned} \quad (13)$$

Inserting (13) into (12) yields

$$H(Y_{j^*(i)}^n | Y_i^n, \boldsymbol{Z}^n) \leq \sum_{t=0}^{n} t \Pr(d_i^* = t) \log_2|\mathcal{X}|, \quad (14)$$

which, combined with (11), immediately produces the desired expression. ∎

Lemma 2: It holds $\mathbb{E}[d_i^*] = \sum_{t=1}^{n} \Pr(d_i^* \geq t) \leq n(1 - \epsilon^n)^N$.

Proof: The equality for $\mathbb{E}[d_i^*]$ is a well-known identity for non-negative random while the inequality follows from manipulation of binomial expressions. See [4] for details. ∎

We are finally in position to prove Theorem 2.

Proof of Theorem 2: We bound each term in the RHS of (6) through (7) and (8), also applying Lemmas 1, 2, and divide by n to get

$$\begin{aligned} R_1 + R_2 \leq &(1-\epsilon)\log_2|\mathcal{X}| + \frac{2}{n} + \bar{P}_{e,1} R_1 \\ &+ \bar{P}_{e,N+1} R_2 + \mathrm{e}^{-N\epsilon^n} \log_2(2N+1), \end{aligned} \quad (15)$$

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 $|\mathcal{X}| = q > 2N$ (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 (15). 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 $\lim_{n\to\infty} \left[\mathrm{e}^{-N(n)\epsilon^n} \log_2 N(n)\right] \stackrel{?}{=} 0$. The last condition can be easily proved from the assumption $N(n) = (1/\epsilon^n)\omega(\ln n)$ through standard calculus.

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] E. Onaran, M. Gatzianas, and C. Fragouli, "Coding schemes for broadcast erasure channels with feedback: the two multicast case," EPFL, Tech. Rep., 2013. [Online]. Available: http://infoscience.epfl.ch/record/184032

[5] 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

[6] 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.

[7] 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.

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

[9] 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.

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

[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.