Delay-Constrained Capacity For Broadcast Erasure Channels: A Linear-Coding-Based Study

Chih-Chun Wang, Email: chihw@purdue.edu
School of Electrical and Computer Engineering, Purdue University, USA

***Abstract*—Supporting delay-sensitive traffic is critical to the next-generation communication network. This work studies the 1-to-2 broadcast *packet erasure channels* with causal ACKnowledgement (ACK), which is motivated by practical downlink access point networks. While the corresponding delay-constrained Shannon capacity remains an open problem (no existing analysis tools can be directly applied), this work focuses on linear codes and proposes three new definitions of delay-constrained throughput based on different *outage metrics:* the file-based, the rank-based, and the packet-based ones. It then fully characterizes the corresponding linear coding capacity regions for *relatively-short-delay flows* — flows for which the delay requirement is no larger than the interval of file arrivals.**

I. Introduction

More and more delay-sensitive traffic are carried over wireline and wireless networks, e.g., the control traffic of smart grids, airbone drone control, online gaming, etc. It is thus imperative to understand the network capacity C under different *end-to-end delay budget* D, i.e., the tradeoff curve $C(\mathsf{D})$ between the capacity and the delay-budget. For illustration, assume that there is only 1 flow in the network. It is intuitive that for extremely small delay budget D_{tiny}, no scheme can do better than the traditional store-&-forward (a.k.a. routing) policy since any additional processing, e.g., network coding (NC), will inevitably incur additional delay, not allowed under the strict delay budget D_{tiny}. As a result, store-&-forward is capacity-achieving when D is small. On the other extreme, when delay is of no concern $\mathsf{D} = \infty$, the max-flow/min-cut theorem shows that the capacity $C(\infty)$ can again be achieved by forwarding along the *maximum flow* of the network. Since store-&-forward is optimal under both extremes D_{tiny} and $\mathsf{D} = \infty$, a common belief was "store-&-forward can achieve $C(\mathsf{D})$ for all D values when there is only one flow in the network."

Recent results [1], [2] disprove the above statement and show that even for the simplest single unicast setting, for some network instances with moderate D values, NC can strictly improve the delay-constrained throughput of the best store-&-forward policy by as much as 100%. Such a finding prompts the following important question:

> *Given an allowable end-to-end delay budget, how to optimally process the information (within the delay budget) to support delay-sensitive traffic beyond what was previously thought possible under the industry-standard store-&-forward policies?*

This work was supported in parts by NSF grants: ECCS-1407603 and CCF-1422997.

Results along this direction will provide a valuable guidance in practice as system designers can then compare the capacity $C(\mathsf{D})$ versus what is achievable by routing, particularly in the context of complexity overhead of additional processing.

This work aims at answering the above fundamental question in the setting of 1-to-2 broadcast packet erasure channels with causal ACKnowledgment (ACK). Such a setting is motivated by downlink access point networks for which each packet transmission may be seriously corrupted, thus receiving "erasure." Causal ACK is sent from the destinations back to the access point after each packet transmission. Our goal is to characterize how much throughput improvement NC can have under a stringent delay-constrained setting.

Having random erasure means that sometimes it is simply impossible to delivery the messages within the delay constraint, the so-called *outage events.* Therefore, the delay-constrained capacity has to be redefined in a way similar to the *outage capacity* for slow fading channels. Focusing exclusively on linear NC, this work proposes three new definitions of delay-constrained throughput based on different *outage metrics:* the file-based, the rank-based, and the packet-based ones. It then fully characterizes the corresponding *linear NC capacity* for *relatively-short-delay flows* — flows for which the delay requirement is less than the interval of file arrivals.

II. Problem Formulation

A. Delay-constrained flows

Assume slotted transmission and consider two coexisting traffic flows. For flow k, $k = 1, 2$, we assume *periodic file arrival* with period prd_k. Namely, the m-th *file* of flow k, $\forall m \geq 1$, will arrive in the beginning of time slot

$$t^{[k]}_{\text{arr}}(m) \triangleq \mathsf{offset}_k + (m-1)\cdot \mathsf{prd}_k \quad (1)$$

where offset_k is the time-offset of flow k. Each file may have multiple packets and we use $L^{[k]}_m$ to denote the number of packets of the m-th flow-k file. For ease of exposition, we assume $L^{[k]}_m$ is i.i.d. Poisson[1] with average $E(L^{[k]}_m) = \lambda_k$.

The flows are delay-constrained. Namely, the m-th file of flow k will "expire" and become useless at time slot

$$t^{[k]}_{\text{exp}}(m) \triangleq \mathsf{offset}_k + (m-1)\cdot \mathsf{prd}_k + \mathsf{D}_k \quad (2)$$

[1]Even though our analysis can be applied to non-Poisson distributions with no modification, the setting of this work depends on the underlying file size distribution. Namely, whether the system can support average throughput λ_k may depend on the distribution of $L^{[k]}_m$. One possible way to circumvent the dependence on the file size distribution is to use an adversarial model and consider the worst-case performance, which is beyond the scope of this work.

where D_k is the delay requirement for flow k. In sum, the packet arrival process can be specified by $(\lambda_k, \mathsf{offset}_k, \mathsf{prd}_k, \mathsf{D}_k)$. When $\mathsf{prd}_k = 1$, our packet arrival model includes the i.i.d. arrival process as a special case.

Definition 1: A flow is of *relatively short delay* if $\mathsf{D}_k \leq \mathsf{prd}_k$. Namely, the delay requirement is very strict such that all the current packets will expire before the next file arrives.

B. The packet erasure channel and the coding schemes

We consider a 1-to-2 broadcast packet erasure channel. For each time t, source s transmits a packet X_t (recall that each file has $L_m^{[k]}$ packets). With probability p_k destination d_k will receive X_t successfully and otherwise it will receive erasure. We assume that the success events are independent over time and between destinations. For example, the probability that at least one of $\{d_1, d_2\}$ receives the packet is $p_{1\vee 2} \triangleq 1-(1-p_1)(1-p_2)$. We also denote the reception status vector $\mathbf{Z}_t$ by

$$\mathbf{Z}_t = (1_{\{X_t \text{ is rcv'd by } d_1\}}, 1_{\{X_t \text{ is rcv'd by } d_2\}})$$

This work assumes exclusively *linear network coding*. Namely, at time t, source s chooses a coding vector $\mathbf{c}_t$ and uses it to linearly combine *all the packets that have arrived (including packets of both flows).* We assume that causal ACK to be sent to s through an error-free control channel. Therefore, $\mathbf{c}_t$ is a function of $[\mathbf{Z}]_1^{t-1} \triangleq \{\mathbf{Z}_1, \cdots, \mathbf{Z}_{t-1}\}$.

At time slot $t_{\exp}^{[k]}(m)$, destination d_k can use the packets it has successfully received in time slot $t \in [1, t_{\exp}^{[k]}(m)-1]$ plus the reception status $[\mathbf{Z}]_1^{t_{\exp}^{[k]}(m)-1}$ to decode the $L_m^{[k]}$ packets of the m-th flow-k file. Again, here we assume that causal ACK is available to d_k through an error-free control channel.

The above definitions can be rigorously formulated using the language of finite field $\mathsf{GF}(q)$. Since our results hold for arbitrary $\mathsf{GF}(q)$, we assume $\mathsf{GF}(q)$ is fixed in this work.

Definition 2: The arrival rates (λ_1, λ_2) are feasible under *file-outage* probability (ϵ_1, ϵ_2) if there exists an NC scheme and a constant t_0 such that for all $T > t_0$ and all k,

$$\frac{\sum_{m=1}^{m_T^{[k]}} \mathsf{E}\{1_{\{\text{the } m\text{-th flow-}k \text{ file can be decoded in time}\}}\}}{m_T^{[k]}} > 1-\epsilon_k \quad (3)$$

where $m_T^{[k]} \triangleq \max\{m : t_{\exp}^{[k]}(m) \leq T\}$ is the number of flow-k packets that are scheduled to be decoded by time T. The capacity $(\lambda_1^*, \lambda_2^*)$ is the supremum over all feasible rates.

Definition 3: Similarly we can define the capacity under *rank-outage* and *packet-outage* events by replacing (3) with

$$\frac{\sum_{m=1}^{m_T^{[k]}} \mathsf{E}\{R_m^{[k]}\}}{m_T^{[k]} \cdot \lambda_k} > 1-\epsilon_k \quad \text{and} \quad \frac{\sum_{m=1}^{m_T^{[k]}} \mathsf{E}\{M_m^{[k]}\}}{m_T^{[k]} \cdot \lambda_k} > 1-\epsilon_k,$$

respectively, where the rank variable $R_m^{[k]}$ is the number of *innovative* combinations (of the $L_m^{[k]}$ packets) that can be decoded; and the variable $M_m^{[k]}$ is the number of original message packets (out of the $L_m^{[k]}$ packets) that can be decoded.

The package-outage capacity is closely related to the concept of *instantly decodable NC* schemes [3], for which a received packet allows immediate decoding of one more original message packet (not just an innovative combination).

C. Example: When there is only 1 flow

To illustrate our definitions, we consider the degenerate case in which there is only 1 flow with *relatively short delay* requirement, described by $(\lambda, \mathsf{offset}, \mathsf{prd}, \mathsf{D})$ with $\mathsf{D} \leq \mathsf{prd}$. By simple counting arguments, arrival rate λ is feasible under file-outage rate ϵ if and only if

$$P(L \leq S) = \sum_{l=0}^{\mathsf{D}} \sum_{s=l}^{\mathsf{D}} \frac{\lambda^l e^{-\lambda}}{l!} \cdot \binom{\mathsf{D}}{s} p^s (1-p)^{\mathsf{D}-s} > 1-\epsilon$$

where L is the random packet size, p is the packet delivery rate, and S is the random number of success within D trials.

For example, when $\mathsf{D} = \mathsf{prd} = 10$, $p = 0.5$, and $\epsilon = 5\%$, the delay-constrained capacity is $\lambda^* \approx 1.945$. The normalized delay-constrained capacity $\frac{\lambda^*}{\mathsf{D}} \approx 0.1945$ is significantly less than the channel success probability $p = 0.5 = C(\infty)$, the Shannon capacity when the delay is of no concern $\mathsf{D} = \infty$.

D. Open questions and comparison to existing results

One can see that the above counting argument does not hold when there are 2 flows since any capacity computation also needs to take into account all possible ways of designing the underlying NC solution. In some sense, our results greatly simplify the NC design space so that one can, for the first time in the literature, use computers to answer the fundamental delay-capacity tradeoff question, see Section I.

Even when limited to store-&-forward policies, it is non-trivial to find the optimal routing rate when there are $K \geq 2$ flows, since one still has to take into account all possible ways of choosing which flow to send at each time slot. Recently [4] characterizes the K-flow delay-constrained *routing capacity* while assuming: (i) each file consists of only 1 packet ($L_m^{[k]} = 1$), (ii) all flows are of *relatively short delay*; and (iii) all K arrival processes share the same $(\mathsf{offset}, \mathsf{prd}, \mathsf{D})$ values. In comparison, our work focuses on *linear NC capacity* with only 2 flows. We still require (ii) but relax both (i) and (iii).

Results in [5] and the references therein study decoding delay minimization for *single multicast* traffic. The goal is to reduce the decoding delay while still maintaining the optimal single-multicast capacity. In contrast, our work focuses exclusively on 2-unicast and we study throughput maximization under any given end-to-end delay constraint.

III. Main Results

We now describe the main results. Consider a 5-dimensional non-negative vector $\mathbf{q} \triangleq (q^{[1]}, q^{[2]}, q_{\text{o.h.}}^{[1]}, q_{\text{o.h.}}^{[2]}, q_{\text{mix}}) \geq \mathbf{0}$. Define

$$\mathcal{Q} \triangleq \{\mathbf{q} \geq \mathbf{0} : q^{[k]} + q_{\text{o.h.}}^{[k]} + q_{\text{mix}} \leq \mathsf{D}_k + 1, \forall k = 1,2\}$$

as the collection of all $\mathbf{q}$ satisfying the two given inequalities. Define $\mathsf{Prd} \triangleq \text{l.c.m.}(\mathsf{prd}_1, \mathsf{prd}_2)$ as the least common multiple of prd_k. Also define $\mathcal{A}$ as a finite set of 8 elements:

$$\mathcal{A} \triangleq \{\mathsf{uc1}, \mathsf{uc2}, \mathsf{xor}, \mathsf{dx1}, \mathsf{dx2}, \mathsf{idle}, \mathsf{pm}, \mathsf{rc}\}.$$

It will be clearer in the later discussion why we denote the 5 queues in $\mathbf{q}$ and the 8 elements in $\mathcal{A}$ with special names rather than just simply naming them as q_1 to q_5 and a_1 to a_8.

Proposition 1 (File-Outage Capacity): Given any p_k and $(\lambda_k, \mathsf{offset}_k, \mathsf{prd}_k, \mathsf{D}_k)$ with $\mathsf{D}_k \leq \mathsf{prd}_k$, we can explicitly construct $2+\mathsf{Prd}$ constant matrices: Φ_{feas}, Φ_{tx}, and $\{\Phi_{\text{ae}}(h) : h \in [0, \mathsf{Prd}-1]\}$, and $2{\cdot}\mathsf{Prd}$ constant vectors $\{\vec{\phi}^{[1]}_{\text{out}}(h), \vec{\phi}^{[2]}_{\text{out}}(h) : h \in [0, \mathsf{Prd}-1]\}$ such that:

Rates (λ_1, λ_2) are feasible under *file outage* prob. (ϵ_1, ϵ_2) if and only if there exist $2\cdot\mathsf{Prd}$ vectors $\vec{x}(h)$ and $\vec{y}(h)$, $\forall h \in [0, \mathsf{Prd}-1]$, such that each $\vec{x}(h) = \{x_{\mathbf{q},a}(h) : \mathbf{q} \in \mathcal{Q}, a \in \mathcal{A}\}$ is of dimension $8|\mathcal{Q}|$; each $\vec{y}(h) = \{y_{\mathbf{q}}(h) : \mathbf{q} \in \mathcal{Q}\}$ is of dimension $|\mathcal{Q}|$; and they satisfy the following inequalities:

Group 1: Probability vector conditions. $\forall h \in [0, \mathsf{Prd}-1]$

$$\vec{x}(h) \geq \mathbf{0} \text{ and } \sum_{\forall \mathbf{q},a} x_{\mathbf{q},a}(h) = 1; \tag{4}$$

$$\text{and} \quad y_{\mathbf{q}}(h) = \sum_{\forall a \in \mathcal{A}} x_{\mathbf{q},a}(h),\ \forall \mathbf{q} \in \mathcal{Q}. \tag{5}$$

Namely, $\vec{x}(h)$ is a *joint distribution* on a random vector $(\mathbf{q}, a)$ and $\vec{y}(h)$ is the corresponding *marginal distribution* on $\mathbf{q}$.

Group 2: Feasibility conditions. $\forall h \in [0, \mathsf{Prd}-1]$

$$\Phi_{\text{feas}} \cdot \vec{x}(h) = \mathbf{0}. \tag{6}$$

Namely, each $\vec{x}(h)$ has to satisfy some feasibility conditions.

Group 3: Transition probability conditions. $\forall h \in [0, \mathsf{Prd}-1]$

$$\vec{y}((h+1) \bmod \mathsf{Prd}) = \Phi_{\text{ae}}(h) \cdot \Phi_{\text{tx}} \cdot \vec{x}(h). \tag{7}$$

When viewed as a Markov chain, (7) specifies the *transition probability* from $\vec{x}(h)$ to $\vec{y}(h{+}1)$. The modulo operation "$(h+1) \bmod \mathsf{Prd}$" enforces a *tail-biting* condition on the Markov chain.

Group 4: Outage probability conditions. $\forall k \in \{1, 2\}$

$$1 - \epsilon_k < \sum_{h=0}^{\mathsf{Prd}-1} \left(\vec{\phi}^{[k]}_{\text{out}}(h)\right)^{\mathrm{T}} \cdot \vec{x}(h) \tag{8}$$

I.e., the outage probability is linearly related to the probability vectors $\vec{x}(h)$.

Proposition 2 (Rank-outage and packet-outage capacity): (i) The rank-outage and the packet-outage capacity regions are always equal; (ii) They can also be described using (4)–(8), for which the matrices Φ_{feas}, Φ_{tx}, and $\{\Phi_{\text{ae}}(h)\}$ are identical as in Proposition 1 and only the vectors $\vec{\phi}^{[k]}_{\text{out}}$ need to be changed.

Due to limited space, we will focus only on explaining the main ideas of Proposition 1. Proposition 2-(ii) can be proven by very similar steps. Proposition 2-(i) can be proven by showing that any scheme that achieves the rank-outage capacity can be converted to an *instantly decodable* NC scheme that achieves the same rate.

The proof of Proposition 1 consists of two steps: *Step 1:* We consider a special class of NC schemes and prove that one can find an NC scheme in that class that supports capacity (λ_1, λ_2) with file-outage probability (ϵ_1, ϵ_2) if and only if (4)–(8) hold. Namely, (4)–(8) characterize the best possible throughput in that class. *Step 2:* We prove that the best throughput in that class is indeed the linear NC capacity by showing that any linear NC scheme can be converted to an instance in that class with the same throughput. We now explain the detailed ideas.

A. Capacity characterization of a special class of schemes

Instead of directly describing the class of schemes of interest, we first describe a simpler but suboptimal class of achievability schemes and use it to illustrate the main ideas.

Consider four virtual queues: $Q^{[k]}$, $Q^{[k]}_{\text{o.h.}}$, $k = 1, 2$. Whenever a new file of flow-k arrives, the corresponding packets are stored in $Q^{[k]}$, a queue containing those flow-k packets that have never been transmitted.

Suppose s sends an uncoded flow-1 packet and it is received by destination d_1, then it will be successfully delivered and we remove it from $Q^{[1]}$. If such a packet is not heard by d_1 but *overheard* by d_2, then we remove such packet from $Q^{[1]}$ and store it in $Q^{[1]}_{\text{o.h.}}$, which contains those flow-1 packets that have not been delivered but have been overheard by d_2. Virtual queue $Q^{[2]}_{\text{o.h.}}$ is defined symmetrically.

Suppose we have two packets $X \in Q^{[1]}_{\text{o.h.}}$ and $Y \in Q^{[2]}_{\text{o.h.}}$, then source can transmit a linear sum $[X+Y]$, which benefits both destinations since X is overheard by d_2 and Y is overheard by d_1. Thus far, we have described 3 coding operations that can be used by s: (i) Send an uncoded packet from $Q^{[k]}$, $k = 1, 2$. We call these two coding operations $\mathsf{uc1}$ and $\mathsf{uc2}$, respectively; (ii) Send a linear sum $[X+Y]$ by combining packets from $Q^{[1]}_{\text{o.h.}}$ and $Q^{[2]}_{\text{o.h.}}$. We call this coding operation xor.

Additionally, source s can also choose to send X directly from $Q^{[1]}_{\text{o.h.}}$ without XOR, or send $Y \in Q^{[2]}_{\text{o.h.}}$ without XOR. We call these two operations *Degenerate XOR*[2] and denote them by $\mathsf{dx1}$ and $\mathsf{dx2}$, respectively. Obviously s can also choose to stay "idle" and not transmitting. In sum, we have four virtual queues $\tilde{\mathbf{Q}} \triangleq (Q^{[1]}, Q^{[2]}, Q^{[1]}_{\text{o.h.}}, Q^{[2]}_{\text{o.h.}})$ and 6 coding operations $\tilde{\mathcal{A}} \triangleq \{\mathsf{uc1}, \mathsf{uc2}, \mathsf{xor}, \mathsf{dx1}, \mathsf{dx2}, \mathsf{idle}\}$ to choose from. These operations are very well documented in [6]–[8].

With the above notation, we describe a class of NC schemes as follows. For any time t we randomly choose among these 6 operations with some probability. We allow the probability of selecting these operations to depend on the current time stamp t and on current queue lengths $\tilde{\mathbf{q}} = (q^{[1]}, q^{[2]}, q^{[1]}_{\text{o.h.}}, q^{[2]}_{\text{o.h.}})$. Namely, the scheme can be described by the conditional probabilities $\{z_{\tilde{a}|\tilde{\mathbf{q}}}(t) : \forall \tilde{a} \in \tilde{\mathcal{A}}, \forall \tilde{\mathbf{q}}, \forall t\}$. We further impose that the selection probability $z_{\tilde{a}|\tilde{\mathbf{q}}}(t)$ to be periodic with period Prd. Namely, $z_{\tilde{a}|\tilde{\mathbf{q}}}(t) = z_{\tilde{a}|\tilde{\mathbf{q}}}(t+\mathsf{Prd})$. By allowing different $\{z_{\tilde{a}|\tilde{\mathbf{q}}}(t)\}$, we have a class of NC schemes.

We now characterize the best delay-constrained throughput for this class of NC schemes. Specifically, we will explicitly construct matrices Φ_{feas}, Φ_{tx}, $\Phi_{\text{ae}}(h)$ and vectors $\phi^{[k]}_{\text{out}}$ such that "a random selection scheme achieves file-outage throughput (λ_1, λ_2) if and only if we can find $\vec{x}(h)$ and $\vec{y}(h)$ vectors that satisfy (a similar version of) (4)–(8)."

[2]These are called *degenerate-XOR* since they are designed for the cases in which $Q^{[3-k]}_{\text{o.h.}}$ is empty but there are some *leftover* packets in $Q^{[k]}_{\text{o.h.}}$ that still need to be delivered but with no packet to combine.

The only if direction ⇒: Consider any given $\{z_{\tilde{a}|\tilde{\mathbf{q}}}(t) : \forall \tilde{a}, \tilde{\mathbf{q}}, t\}$ that achieves the file-outage throughput (λ_1, λ_2). Since $\{z_{\tilde{a}|\tilde{\mathbf{q}}}(t)\}$ has period Prd, the least common multiple of the arrival/departure patterns of the two flows, in the long run the *distribution* of the queue lengths at each time t will reach a *steady-state*. Using the given $\{z_{\tilde{a}|\tilde{\mathbf{q}}}(t) : \forall \tilde{a}, \tilde{\mathbf{q}}, t\}$, we compute the steady state distribution and denote it by $\vec{y}(t) = \{y_{\tilde{\mathbf{q}}}(t) : \forall \tilde{\mathbf{q}}\}$, which is also periodic with period Prd.

After computing the distribution $\{y_{\tilde{\mathbf{q}}}(t)\}$ on $\tilde{\mathbf{q}}$, we compute the joint distribution on $(\tilde{\mathbf{q}}, \tilde{a})$ by $x_{\tilde{\mathbf{q}},\tilde{a}}(t) = y_{\tilde{\mathbf{q}}}(t) z_{\tilde{a}|\tilde{\mathbf{q}}}(t)$. Obviously the resulting vectors $\vec{y}(t)$ and $\vec{x}(t)$ are probability distributions that satisfy (4) and (5).

We now notice that the computed $\vec{x}(t)$ must satisfy:

$$x_{\tilde{\mathbf{q}},\tilde{a}}(t) = 0 \quad \text{if} \quad \tilde{a} = \mathsf{uc1} \text{ and } q^{[1]} = 0; \tag{9}$$

$$x_{\tilde{\mathbf{q}},\tilde{a}}(t) = 0 \quad \text{if} \quad \tilde{a} = \mathsf{uc2} \text{ and } q^{[2]} = 0; \tag{10}$$

$$x_{\tilde{\mathbf{q}},\tilde{a}}(t) = 0 \quad \text{if} \quad \tilde{a} = \mathsf{xor} \text{ and } q^{[1]}_{\text{o.h.}} q^{[2]}_{\text{o.h.}} = 0; \tag{11}$$

$$x_{\tilde{\mathbf{q}},\tilde{a}}(t) = 0 \quad \text{if} \quad \tilde{a} = \mathsf{dx1} \text{ and } q^{[1]}_{\text{o.h.}} = 0; \tag{12}$$

$$x_{\tilde{\mathbf{q}},\tilde{a}}(t) = 0 \quad \text{if} \quad \tilde{a} = \mathsf{dx2} \text{ and } q^{[2]}_{\text{o.h.}} = 0. \tag{13}$$

For example, to send an uncoded flow-1 packet (coding choice $\tilde{a} = \mathsf{uc1}$) we must have $Q^{[1]}$ being non-empty. Therefore, $z_{\tilde{a}|\tilde{\mathbf{q}}} = 0$ if $\tilde{a} = \mathsf{uc1}$ and $q^{[1]} = 0$, which implies (9). Similarly, to be able to send an XORed packet (coding choice $\tilde{a} = \mathsf{xor}$), both $Q^{[1]}_{\text{o.h.}}$ and $Q^{[2]}_{\text{o.h.}}$ must be non-empty, which implies (11). (9)–(13) are called the *feasibility conditions* and can be written in a matrix form (6).

To explain (7), recall that $\vec{y}(t+1)$ denote the steady-state queue-length distribution in the *beginning* of time $t+1$. We now define the $\vec{w}(t)$ vector as the steady-state queue-length distribution in the *end* of time t. The difference between $\vec{y}(t+1)$ and $\vec{w}(t)$ is that at the beginning of time $t+1$, some old packets will expire and some new packets may arrive. Therefore we have

$$\vec{y}(t+1) = \Phi_{\text{ae}}(t) \cdot \vec{w}(t) \tag{14}$$

where $\Phi_{\text{ae}}(t)$ is the state transition matrix due to packet Arrival and Expiration in the beginning of time $t+1$ (or equivalently in the end of time t). One can explicitly write down $\Phi_{\text{ae}}(t)$ for all t by checking whether there is any new packet arrival/expiration at time $t+1$ using the predefined arrival/expiration time in (1) and (2).

For example, if there is no arrival and expiration in the beginning of time $t+1$, we must have $\vec{y}(t+1) = \vec{w}(t)$. We can then set $\Phi_{\text{ae}}(t) = \mathbf{I}$, the identity matrix. Another example is that suppose at time $t+1$, (i) no new flow-1 packet arrives; (ii) all old flow-1 packets expire; (iii) l new flow-2 packets arrive; and (iv) no old flow-2 packet expires. Then we have

$$\left(w_{(q^{[1]},q^{[2]},q^{[1]}_{\text{o.h.}},q^{[2]}_{\text{o.h.}})}(t)\right) \cdot \left(\frac{(\lambda_2)^l e^{-\lambda_2}}{l!}\right) \longrightarrow y_{(0,q^{[2]}+l,0,q^{[2]}_{\text{o.h.}})}(t+1) \tag{15}$$

where the notation in (15) means that a $\frac{(\lambda_2)^l e^{-\lambda_2}}{l!}$ portion of the value of $w_{(q^{[1]},q^{[2]},q^{[1]}_{\text{o.h.}},q^{[2]}_{\text{o.h.}})}(t)$ will be "transferred" to $y_{(0,q^{[2]}+l,0,q^{[2]}_{\text{o.h.}})}(t+1)$. The reason is as follows: All the flow-1 queues will be reset to 0 since flow-1 packets have expired; $Q^{[2]}$ will increase by l, where l is the number of new flow-2 packets entering the system. Since the file size is Poisson, we have the transition probability[3] being $\frac{(\lambda_2)^l e^{-\lambda_2}}{l!}$.

Using the deterministic arrival/expiration time instants in (1) and (2), we can explicitly write down the $\Phi_{\text{ae}}(t)$ in (14) for all t, which can be further strengthened as

$$\vec{y}((t+1) \bmod \mathsf{Prd}) = \Phi_{\text{ae}}(t) \cdot \vec{w}(t) \tag{16}$$

since the computed $\vec{y}(t)$ has period Prd.

We now claim that the $\vec{w}(t)$ and $\vec{x}(t)$ (both are computed from the given $\{z_{\tilde{a}|\tilde{\mathbf{q}}}(t) : \forall \tilde{a}, \tilde{\mathbf{q}}, t\}$) must satisfy

$$\vec{w}(t) = \Phi_{\text{tx}} \cdot \vec{x}(t) \tag{17}$$

for some Φ_{tx}. The transition matrix Φ_{tx} can be computed by analyzing the packet movement within the queues $Q^{[1]}$, $Q^{[2]}$, $Q^{[1]}_{\text{o.h.}}$ and $Q^{[2]}_{\text{o.h.}}$, which is well documented in [6]–[8] and is thus omitted. For example, if s sends an uncoded packet from $Q^{[2]}$, i.e., $\tilde{a} = \mathsf{uc2}$, then with probability $p_1(1-p_2)$ it will be overheard by d_1 but not by d_2. Therefore the packet will leave $Q^{[2]}$ and enter $Q^{[2]}_{\text{o.h.}}$. Using the same notation as in (15), the probability transfer relationship is thus

$$\left(x_{(q^{[1]},q^{[2]},q^{[1]}_{\text{o.h.}},q^{[2]}_{\text{o.h.}}),\mathsf{uc2}}(t)\right) \cdot p_1(1-p_2) \longrightarrow w_{(q^{[1]},q^{[2]}-1,q^{[1]}_{\text{o.h.}},q^{[2]}_{\text{o.h.}}+1)}(t);$$

Together (16) and (17) lead to (7).

Finally, we consider the outage events. The successful file delivery event happens when in the end of time slot $t = t^{[k]}_{\text{exp}}(m) - 1$, all packets have been successfully delivered and there is no flow-k packet remaining in the queues. As a result, the file delivery probability of flow-k satisfies

$$1 - \epsilon_k < \lim_{M\to\infty} \frac{1}{M} \sum_{m=1}^{M} \sum_{\tilde{\mathbf{q}}: q^{[k]} = q^{[k]}_{\text{o.h.}} = 0} w_{\tilde{\mathbf{q}}}(t^{[k]}_{\text{exp}}(m) - 1)$$

$$= \frac{1}{\mathsf{Prd}/\mathsf{prd}_k} \sum_{m=1}^{\mathsf{Prd}/\mathsf{prd}_k} \sum_{\tilde{\mathbf{q}}: q^{[k]} = q^{[k]}_{\text{o.h.}} = 0} w_{\tilde{\mathbf{q}}}(t^{[k]}_{\text{exp}}(m) - 1) \tag{18}$$

where the last equality follows from that $\vec{w}(t)$ has period Prd. Jointly (18) and (17) can be used to write down (8).

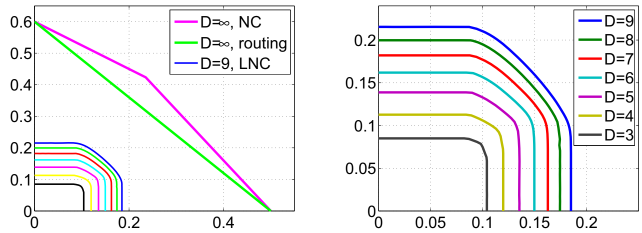
In sum, we have proven that any feasible random-selection scheme can be used to compute $\vec{x}(h)$ and $\vec{y}(h)$ vectors that satisfy (a similar version of) (4)–(8).

The if direction ⇐: We will prove that any $\vec{x}(h)$ and $\vec{y}(h)$ vectors satisfying (4)–(8) can be used to construct $\{z_{\tilde{a}|\tilde{\mathbf{q}}}(t) : \forall \tilde{a}, \tilde{\mathbf{q}}, t\}$ such that the corresponding random-selection scheme achieves file-outage throughput (λ_1, λ_2). This can be achieved by setting

$$z_{\tilde{a}|\tilde{\mathbf{q}}}(t) = \frac{x_{\tilde{\mathbf{q}},\tilde{a}}}{y_{\tilde{\mathbf{q}}}(t)}.$$

Then following similar analysis as in the "only-if" direction, we can use (8) to prove that the outage condition (3) holds.

[3] Since flow-2 can deliver at most D_2 packets within the delay constraint, one can truncate the Poisson distribution and only consider $l \leq \mathsf{D}_2 + 1$ where $l = \mathsf{D}_2 + 1$ represents the case of *excessive arrivals*. The truncation is key to keep the state space finite so that Proposition 1 remains computable.



(a) D being finite vs $\mathsf{D} = \infty$ (b) Magnified version

Fig. 1. Normalized file-outage capacity with $(p_1, p_2) = (0.5, 0.6)$ and $(\epsilon_1, \epsilon_2) = (5\%, 5\%)$. The x- and y-axes are λ_1/D and λ_2/D, respectively.

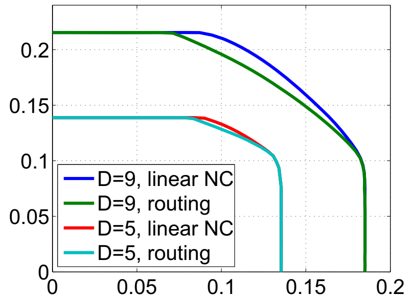


Fig. 2. File-outage capacity of linear NC versus routing for $\mathsf{D} = 5$ and $\mathsf{D} = 9$, respectively. The x- and y-axes are λ_1/D and λ_2/D, respectively.

B. More detailed explanations

The scheme in Section III-A is based on 4 virtual queues $\tilde{\mathbf{Q}} \triangleq (Q^{[1]}, Q^{[2]}, Q^{[1]}_{\text{o.h.}}, Q^{[2]}_{\text{o.h.}})$ and 6 coding operations $\tilde{\mathcal{A}} \triangleq \{\mathsf{uc1}, \mathsf{uc2}, \mathsf{xor}, \mathsf{dx1}, \mathsf{dx2}, \mathsf{idle}\}$ to choose from. In [9], a new network coding scheme is proposed, which strictly generalizes the simple XOR-based scheme by introducing 1 additional virtual queue, called Q_{mix}, and 2 additional coding operations called pm and rc, respectively. Using the scheme in [9] we can now design a similar random selection scheme based on 5 virtual queues $\mathbf{Q} \triangleq (Q^{[1]}, Q^{[2]}, Q^{[1]}_{\text{o.h.}}, Q^{[2]}_{\text{o.h.}}, Q_{\text{mix}})$ and randomly selects one of 8 coding operations $\mathcal{A} \triangleq \{\mathsf{uc1}, \mathsf{uc2}, \mathsf{xor}, \mathsf{dx1}, \mathsf{dx2}, \mathsf{idle}, \mathsf{pm}, \mathsf{rc}\}$ in each time slot. By similar reasonings, (4)–(8) now characterize the best delay-constrained throughput using this new class of NC schemes.

The remaining step is to prove that the best throughput of the above random-selection NC schemes is indeed the linear coding capacity. This can be proven by (i) using the *coding-type framework* [10] to reduce the design space of linear NC in a lossless fashion; and (ii) establishing that given any arbitrary linear NC scheme with a reduced design space, which can possibly be deterministic and non-periodic, one can construct an instance of the above random-selection NC schemes that has the same achievable rates. The key step of the proof is to use the *temporal average* of the arbitrarily given scheme as the random selection probability $\{z_{\tilde{a}|\tilde{\mathbf{q}}}(t)\}$.

IV. A Numerical Example

For any $\mathsf{D} \geq 3$, we set $(\mathsf{offset}_1, \mathsf{prd}_1, \mathsf{D}_1) = (0, \mathsf{D}, \mathsf{D})$ and $(\mathsf{offset}_2, \mathsf{prd}_2, \mathsf{D}_2) = (2, \mathsf{D}, \mathsf{D} - 1)$. I.e., the files of both flows arrive with a common period D but flow-2 has a slightly stricter delay budget and is offset by 2 time slots. Fig. 1 plots the normalized linear NC capacity $\left(\frac{\lambda_1^*}{\mathsf{D}}, \frac{\lambda_2^*}{\mathsf{D}}\right)$ for $\mathsf{D} = 3$ to 9. Fig. 1 shows that larger D leads to larger normalized capacity $\frac{\lambda_k}{\mathsf{D}}$. For $\mathsf{D} = 9$ the delay-constrained sum-rate capacity is 47.8% of the asymptotic NC capacity ($\mathsf{D} = \infty$) in [6].

Using the above setting, one can prove that there is zero linear NC gain when $\mathsf{D} = 3$ since such a small D leaves no room for NC. Following Fig. 1, Fig. 2 compares the linear NC capacity versus routing for $\mathsf{D} = 5$ and 9, respectively. One can see that more relaxed delay requirement (larger D) leads to larger NC gain. [6] can be used to compute the "$\mathsf{D} = \infty$" sum-rate gain of NC over routing, which is 9.80% in this setting. Fig. 2 shows that the sum-rate gain for $\mathsf{D} = 9$ is 4.96%. Namely, *more than half of the asymptotic gain can be realized for* D *as small as 9!*

Note that the optimal linear NC gain reported here (4.96% for $\mathsf{D} = 9$), though small, is *absolute* in the sense that we allow the same delay budget for both NC and routing. Therefore, the throughput gain can be had with zero delay penalty!

V. Conclusion

This work has studied the 1-to-2 broadcast packet erasure channels with causal ACK and characterized the delay-constrained linear network coding (NC) capacity under fixed *relatively short* delay constraints. The results provide the first concrete throughput comparison between linear NC and routing under the same finite delay constraint, a long overdue information necessary for evaluating the practicality of NC.

References

[1] C.-C. Wang and M. Chen, "Sending perishable information: Coding improves delay-constrained throughput even for single unicast," in *Proc. IEEE Int'l Symp. Inform. Theory*. Honolulu, USA, June 2014.

[2] S. Kamath, S. Kannan, P. Viswanath, and C. Chekuri, "Delay-constrained unicast and the triangle-cast problem," in *Proc. IEEE Int'l Symp. Inform. Theory*. Hong Kong, China, June 2015, pp. 804–808.

[3] X. Li, C.-C. Wang, and X. Lin, "Throughput and delay analysis on uncoded and coded wireless broadcast with hard deadline constraints," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 5, May 2011.

[4] I.-H. Hou, V. Borkar, and P. Kumar, "A theory of QoS in wireless," in *Proc. 28th IEEE Conference on Computer Communications (INFOCOM)*, 2009, pp. 486–494.

[5] E. Drinea, C. Fragouli, and L. Keller, "Delay with network coding and feedback," in *Proc. IEEE Int'l Symp. Inform. Theory*. Seoul, Korea, June 2009.

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

[7] C.-C. Wang, "Capacity of 1-to-K broadcast packet erasure channels with channel output feedback," *IEEE Trans. Inf. Theory*, vol. 58, no. 2, pp. 957–988, February 2012.

[8] M. Heindlmaier and S. Bidokhti, "Capacity regions of two-user broadcast erasure channels with feedback and hidden memory," in *Proc. IEEE Int'l Symp. Inform. Theory*. Hong Kong, China, June 2015.

[9] W. Kuo and C.-C. Wang, "Robust and optimal opportunistic scheduling for downlink 2-flow inter-session network coding with varying channel quality," in *Proc. 33rd IEEE Conference on Computer Communications (INFOCOM)*. Toronto, Canada, April 2014.

[10] C.-C. Wang and J. Han, "The capacity region of two-receiver multiple-input broadcast packet erasure channels with channel output feedback," *IEEE Trans. Inf. Theory*, vol. 60, no. 9, pp. 5597–5626, September 2014.