On Throughput-Delay Trade-offs in Streaming Communication

Gauri Joshi, Student Member, IEEE, Yuval Kochman, Member, IEEE, Gregory W. Wornell, Fellow, IEEE

Index Terms

erasure codes, streaming, in-order packet delivery, throughput-delay trade-off

This work was presented in part at the International Symposium on Information Theory, July 2012, the INFOCOM workshop on Communication and Networking Techniques for Contemporary Video, Apr 2014, and the International Symposium on Network Coding, Jun 2014.

G. Joshi and G. W. Wornell are with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA. Y. Kochman is with the School of Computer Science and Engineering, the Hebrew University of Jerusalem, Israel. (E-mail: gauri@mit.edu, yuvalko@cs.huji.ac.il, gww@mit.edu)
## CONTENTS

### I Introduction

I-A Motivation ........................................... 3  
I-B Previous Work ......................................... 4  
I-C Our Contributions ....................................... 5  
I-D Organization ........................................... 6  

### II Preliminaries

II-A System Model .......................................... 6  
II-B Throughput and Delay Metrics .......................... 7  

### III Point-to-point Streaming

III-A Immediate Feedback ................................... 9  
III-B No Feedback .......................................... 9  
III-C General Block-wise Feedback .......................... 12  

### IV Multicast Streaming

IV-A Problem Setup ......................................... 19  
IV-B Optimal Performance for One of the Users .......... 21  
IV-C General Throughput-delay Trade-offs ................. 27  

### V Concluding Remarks

V-A Major Implications ..................................... 31  
V-B Future Perspectives ..................................... 33  

References ................................................. 33
I. INTRODUCTION

A. Motivation

A recent report [1] shows that 62% of the Internet traffic in North America comes from real-time streaming applications. Unlike traditional file transfer where only total delay matters, streaming imposes delay constraints on each individual packet. Further, many applications require in-order playback of packets at the receiver. Packets received out of order are buffered until the missing packets in the sequence are successfully decoded. In audio and video applications some packets can be dropped without affecting the streaming quality. However, other applications such as remote desktop, and collaborative tools such as Dropbox [2] and Google Docs [3] have strict order constraints on packets, where packets represent instructions that need to be executed in order at the receiver.

Thus, there is a need to develop transmission schemes that can ensure in-order packet delivery to the user, with efficient use of available bandwidth. To ensure that packets are decoded in order, the transmission scheme must give higher priority to older packets that were delayed, or received in error. However, repeating old packets instead of transmitting new packets results in a loss in the overall rate of packet delivery to the user, i.e., the throughput. Thus there is a fundamental trade-off between throughput and in-order decoding delay.

The throughput loss incurred to achieve low in-order decoding delay can be significantly reduced if the source receives feedback about packet losses, and thus can adapt its future transmission strategy to strike the right balance between old and new packets. We study this interplay between feedback and the throughput-delay trade-off. This analysis can help design transmission schemes that achieve the best throughput-delay trade-off with a limited amount of feedback.

Further, if there are more than one users that request a common stream of packets, then there is an additional inter-dependence between the throughput-delay trade-offs. Since the users decode different sets of packets depending of their channel erasures, a packet that is innovative and in-order for one user, may be redundant for another. Thus, the source must strike a balance of giving priority to each of the users. We present a framework to analyze such multicast streaming scenarios.
B. Previous Work

When there is immediate and error-free feedback, it is well understood that a simple Automatic-repeat-request (ARQ) scheme is both throughput and delay optimal. But only a few papers in literature have analyzed streaming codes with delayed or no feedback. Fountain codes [4] are capacity-achieving erasure codes, but they are not suitable for streaming because the decoding delay is proportional to the size of the data. Streaming codes without feedback for constrained channels such as adversarial and cyclic burst erasure channels were first proposed in [5], and also extensively explored in [6], [7]. The thesis [5] also proposed codes for more general erasure models and analyzed their decoding delay. These codes are based upon sending linear combinations of source packets; indeed, it can be shown that there is no loss in restricting the codes to be linear. However, decoding delay does not capture in order packet delivery, which is required for streaming applications. This aspect is captured in the delay metrics in [8] and [9], which consider that packets are played in-order at the receiver. The authors in [8] analyze the throughput-delay trade-off for uncoded packet transmission over a channel with long feedback delay. In [9] we propose coding schemes that minimize playback delay in point-to-point streaming for the no feedback and immediate feedback cases. However, the case of block-wise feedback to the source remains to be explored.

When the source needs to stream the data to multiple users, even the immediate feedback case becomes non-trivial. The use of network coding in multicast packet transmission has been studied in [10]–[15]. The authors in [10] use as a delay metric the number of coded packets that are successfully received, but do not allow immediate decoding of a source packet. For two users, the paper shows that a greedy coding scheme is throughput-optimal and guarantees immediate decoding in every slot. However, optimality of this scheme has not been proved for three or more users. In [11], the authors analyze decoding delay with the greedy coding scheme in the two user case. However, both these delay metrics do not capture the aspect of in-order packet delivery.

In-order packet delivery for multicast with immediate feedback is considered in [12]–[14]. These works consider that packets are generated by a Poisson process and are greedily added to all future coded combinations. Another related work is [15] which also considers Poisson
packet generation in a two-user multicast scenario and derives the stability condition for having finite delivery delay. However, in practice the source can use feedback about past erasures to decide which packets to add to the coded combinations, instead of just greedy coding over all generated packets. In this paper we consider this more realistic model of packet transmission.

C. Our Contributions

For the point-to-point streaming scenario, also presented in [16], we consider this unexplored problem of how to effectively utilize feedback received by the source to ensure in-order packet delivery to the user. We consider block-wise feedback, where the source receives information about past channel states at periodic intervals. In contrast to playback delay considered in [8] and [9], we propose a more versatile delay metric called the in-order decoding exponent. This metric captures the burstiness in the in-order decoding of packet for applications that require packets in-order, but do not necessarily play them at a constant rate.

In the limiting case of immediate feedback, we can use ARQ and achieve the optimal throughput and delay simultaneously. But as the feedback delay increases, we have to compromise on at least one of these metrics. Our analysis shows that for the same throughput, having more frequent block-wise feedback significantly improves the in-order decoding exponent. This conclusion is reminiscent of [17] which studied the effect of feedback on error exponents. We present a spectrum of coding schemes spanning the throughput-delay trade-off, and prove that they give the best trade-off within a broad class of schemes for the no feedback, and small feedback delay cases.

We extend the analysis the trade-off between throughput and in-order delivery for point-to-point streaming to the multicast scenario. We focus of two user case with immediate feedback to the source. Since each user may decode a different set of packets depending of the channel erasures, the in-order packet required by one user may be redundant to the other user. Thus, giving priority to one user can cause a throughput loss to the other. We analyze this interplay between the throughput-delay trade-offs of the two users using a Markov chain model for in-order packet decoding. In [20] we analyzed the throughput-delay trade-off with the delay metric being the probability of in-order decoding, instead of the decoding exponent considered in this work.
D. Organization

The rest of the paper is organized as follows. In Section III we analyze the effect of block-wise feedback on the trade-off between the throughput and the in-order decoding exponent in point-to-point streaming. We first consider the extreme cases of immediate feedback and no feedback in Section III-A and Section III-B respectively. In Section III-C we propose coding schemes for the general case of block-wise feedback after every \( d \) slots and analyze their throughput-delay trade-offs.

In Section IV we present a framework for analyzing in-order packet delivery in multicast streaming. In Section IV-B we find the best coding scheme for a user that is piggybacking on a primary user that is always given higher priority. In Section IV-C we propose general coding schemes that allow us to tune the level of priority given to each user and hence achieve different points on its throughput-delay trade-off.

II. PRELIMINARIES

A. System Model

We consider a point-to-point packet streaming scenario where the source has a large stream of packets \( s_1, s_2, \cdots, s_n \). The encoder creates a coded packet \( y_n = f(s_1, s_2, \cdots, s_n) \) in each slot \( n \) and transmits it over the channel. The encoding function \( f \) is known to the receiver. For example, if \( y_n \) is a linear combination of the source packets, the coefficients are included in the transmitted packet so that the receiver can use them to decode the source packets from the coded combination. Without loss of generality, we can assume that \( y_n \) is a linear combination of the source packets.

We consider an i.i.d. packet erasure channel where every transmitted packet is correctly received with probability \( p \), and otherwise received in error and discarded. An erasure channel is a good model when encoded packets have a set of checksum bits that can be used to verify with high probability whether the received packet is error-free.

The receiver application requires the stream of packets to be in order. Packets received out of order are buffered until the missing packets in the sequence are decoded. We assume that the buffer is large enough to store all the out-of-order packets. Every time the earliest undecoded packet is decoded, a burst of in-order decoded packets is delivered to the application. For example,
suppose that \( s_1 \) has been delivered and \( s_3, s_4, s_6 \) are decoded and waiting in the buffer. If \( s_2 \) is decoded in the next slot, then \( s_2, s_3 \) and \( s_4 \) are delivered to the application.

Due to this in-order property, the transmitter can stop including \( s_k \) in coded packets when it knows that the receiver can decode \( s_k \) once all \( s_i \) for \( i < k \) are decoded. We refer such packets as “seen” packets. The notion of “seen” is defined formally as follows.

**Definition 1 (Seen Packets).** A packet \( s_k \) is marked as “seen” by the transmitter when it knows that a coded combination that only includes \( s_k \) and packets \( s_i \) for \( 1 \leq i \leq k \) is received successfully.

We consider that the source receives block-wise feedback about channel erasures after every \( d \) slots. Thus, before transmitting in slot \( kd + 1 \), for all integers \( k \geq 1 \), the source knows about the erasures in slots \( (k - 1)d + 1 \) to \( kd \). It can use this information to adapt its transmission strategy in slot \( kd + 1 \). Block-wise feedback can be used to model a half-duplex communication channel where after every \( d \) slots of packet transmission, the channel is reserved for the receiver to send feedback about the status of decoding. Note that the feedback can be used to estimate \( p \), the probability of success of the erasure channel, when it is unknown to the source.

**B. Throughput and Delay Metrics**

We consider two metrics to measure the quality of streaming, the throughput \( \tau \) and in-order decoding exponent \( \lambda \). First we define the notion of innovative coded packets.

**Definition 2.** A coded packet is said to be “innovative” if it is linear independent with respect to the coded packets received until that time.

The throughput is the rate at which “innovative” coded packets are received and is formally defined as follows.

**Definition 3 (Throughput).** If \( I_n \) is the number of innovative packets received until time \( n \), the throughput \( \tau \) is,

\[
\lim_{n \to \infty} \frac{I_n}{n} \quad \text{in probability.} \tag{1}
\]
The bandwidth required is proportional to $1/\tau$. The receiver application may require a minimum level of throughput. For example, if applications with playback require $\tau$ to be greater than the playback rate. The throughput captures the overall rate at which packets go through the channel, irrespective of the order. The in-order decoding aspect is captured by a metric called the in-order decoding exponent $\lambda$ which is defined as follows.

**Definition 4 (In-order Decoding Exponent).** Let $T$ be the time between two successive instants of in-order decoding of one or more packets at the receiver. Then the in-order decoding exponent $\lambda$ is

$$\lambda \triangleq -\lim_{n \to \infty} \frac{\log \Pr(T > n)}{n} \text{ in probability}.$$  \hspace{1cm} (2)

The relation (2) can also be stated as $\Pr(T > n) \approx e^{-n\lambda}$ where $\approx$ stands for asymptotic equality defined in [18, Page 63]. The in-order decoding exponent captures the burstiness in packet decoding. We choose this metric because key quality-of-service indicators can be expressed in terms of it. For example, if the streaming application plays packets at a constant rate, and there are $b$ in-order decoded packets in the receiver buffer, then the probability of an interruption in playback is proportional to $e^{-\lambda b}$. We can show that the expected playback delay asymptotically grows as $1/\gamma \cdot \log n$, where $\gamma \geq \lambda$.

We first consider the point-to-point streaming in Section III and analyze how the trade-off between $\tau$ and $\lambda$ is affected by the block-wise feedback delay $d$. In Section IV we analyze the trade-off for multicast streaming with immediate feedback.

Another possible metric is $\mathbb{E}[T]$ which is the expected time between two in-order packet delivery instants. It captures the average delay between in-order decoding instants, whereas $\lambda$ (defined in (2)) measures the tail exponent. Although we focus on maximizing $\lambda$ in this paper, the coding schemes that we propose ensure finite $\mathbb{E}[T]$. For the point-to-point streaming case considered in Section III we can evaluate bound $\mathbb{E}[T]$ to within an interval $d$ slots. For the multicast streaming with immediate feedback in Section IV we can exactly evaluate $\mathbb{E}[T]$. Further, the proposed coding schemes are optimal in both $\lambda$ and $\mathbb{E}[T]$. The optimality in $\mathbb{E}[T]$ is shown in [20] which studies the throughput-delay trade-off with $\mathbb{E}[T]$ as the delay metric.
III. POINT-TO-POINT STREAMING

A. Immediate Feedback

In the immediate feedback \((d = 1)\) case, the source has complete knowledge of past erasures before transmitting each packet. We can show that a simple automatic-repeat-request (ARQ) scheme is optimal in both \(\tau\) and \(\lambda\). In this scheme, the source transmits the lowest index unseen packet, and repeats it until the packet successfully goes through the channel.

Since a new packet is received in every successful slot, the throughput \(\tau = p\), the success probability of the erasure channel. The ARQ scheme is throughput-optimal because the throughput \(\tau = p\) is equal to the information-theoretic capacity of the erasure channel \([18]\). Moreover, it also gives the optimal the in-order decoding exponent \(\lambda\) because one in-order packet is decoded in every successful slot. To find \(\lambda\), first observe that the tail distribution of the time \(T\), the interval between successive in-order decoding instants is,

\[
\Pr(T > n) = (1 - p)^n
\]

Substituting this in Definition \([4]\) we get the exponent \(\lambda = -\log(1 - p)\). Thus, the trade-off for the immediate feedback case is \((\tau, \lambda) = (p, -\log(1 - p))\).

From this analysis of the immediate feedback case we can find limits on the range of achievable \((\tau, \lambda)\) for any feedback delay \(d\). Since a scheme with immediate feedback can always simulate one with delayed feedback, the throughput and delay metrics \((\tau, \lambda)\) achievable for any feedback delay \(d\) must lie in the region \(0 \leq \tau \leq p\), and \(0 \leq \lambda \leq -\log(1 - p)\).

B. No Feedback

Now we consider the other extreme case \((d = \infty)\), corresponding to when there is no feedback to the source. We propose a coding scheme that gives the best \((\tau, \lambda)\) trade-off among the class of full-rank codes, defined as follows.

**Definition 5 (Full-rank Codes).** In slot \(n\) we transmit a linear combination of all packets \(s_1\) to \(s_{V[n]}\), where the coefficients are chosen from a large enough field such that the coded combinations are independent with high probability. We refer to \(V[n]\) as the transmit index in slot \(n\).
Conjecture 1. Given transmit index $V[n]$, there is no loss of generality in including all packets $s_1$ to $s_{V[n]}$.

We believe this conjecture is true because the packets are required in-order at the receiver. Thus, every packet $s_j, j < V[n]$ is required before packet $s_{V[n]}$ and there is no advantage in excluding $s_j$ from the combination. Hence we believe that there is no loss of generality in restricting our attention to full-rank codes. A direct approach to verifying this conjecture would involve checking all possible channel erasure patterns.

Theorem 1. The optimal throughput-delay trade-off among full-rank codes is $(\tau, \lambda) = (r, D(r||p))$ for all $0 \leq r \leq p$. It is achieved by the coding scheme with $V[n] = \lceil rn \rceil$ for all $n$.

The term $D(r||p)$ is the binary information divergence function, which is defined for $0 < p, r < 1$ as

$$D(r||p) = r \log \frac{r}{p} + (1 - r) \log \frac{1 - r}{1 - p}, \quad (4)$$

where $0 \log 0$ is assumed to be $0$. As $r \to 0$, $D(r||p)$ converges to $-\log(1 - p)$, which is the best possible $\lambda$ as given in Section III-A.

Proof: We first show that the scheme with transmit index $V[n] = \lceil rn \rceil$ in time slot $n$ achieves the trade-off $(\tau, \lambda) = (r, D(r||p))$. Then we prove the converse by showing that no other full-rank scheme gives a better trade-off.

Achievability Proof: Consider the scheme with transmit index $V[n] = \lceil rn \rceil$, where $r$ represents the rate of adding new packets to the transmitted stream. The rate of adding packets is below the capacity of the erasure channel. Thus it is easy to see that the throughput $\tau = r$. Let $E[n]$ be the number of combinations, or equations received until time $n$. It follows the binomial distribution with parameter $p$. All packets $s_1 \cdots s_{V[n]}$ are decoded when $E[n] \geq V[n]$. Define the event $G_n = \{ E[j] < V[j] \text{ for all } 1 \leq j \leq n \}$, that there is no packet decoding until slot $n$. The tail distribution of time $T$ between successive in-order decoding instants is,

$$\Pr(T > n) = \sum_{k=0}^{[nr]-1} \Pr(E[n] = k) \Pr(G_n|E[n] = k),$$

$$= \sum_{k=0}^{[nr]-1} \binom{n}{k} p^k (1 - p)^{n-k} \Pr(G_n|E[n] = k),$$
where $\Pr(G_n | E[n] = k) = 1 - k/n$ as given by the Generalized Ballot theorem in [19, Chapter 4]. Hence it is sub-exponential and does not affect the exponent of $\Pr(T > n)$ and we have

$$\Pr(T > n) \doteq \sum_{k=0}^{[nr]-1} \binom{n}{k} p^k (1 - p)^{n-k},$$  \hspace{1cm} (5)$$

$$\doteq \binom{n}{[nr]-1} p^{[nr]-1} (1 - p)^{n-[nr]+1},$$  \hspace{1cm} (6)$$

$$\doteq e^{-nD(r||p)},$$  \hspace{1cm} (7)$$

where in (5) we take the asymptotic equality $\doteq$ to find the exponent of $\Pr(T > n)$, and remove the $\Pr(G_n | E[n] = k)$ term because it is sub-exponential. In (6), we only retain the $k = [nr] - 1$ term from the summation because for $r \leq p$, that term asymptotically dominates other terms. Finally, we use the Stirlings approximation $\binom{n}{k} \approx e^{nH(k/n)}$ to obtain (7).

**Converse Proof:** First we show that the transmit index $V[n]$ of the optimal full-rank scheme should be non-decreasing in $n$. Given any scheme, we can permute the order of transmitting the coded packets such that $V[n]$ is non-decreasing in $n$. This does not affect the throughput $\tau$, but it can improve the in-order decoding exponent $\lambda$ because decoding can occur sooner when the initial coded packets include fewer source packets.

We now show that it is optimal to have $V[n] = \lceil rn \rceil$, where we add new packets to the transmitted stream at a constant rate $r$. Suppose a full-rank scheme uses rate $r_i$ for $n_i$ slots for
all $1 \leq i \leq L$, such that $\sum_{i=0}^{L} n_i = n$ and $\sum_{i=1}^{L} n_i r_i = nr$. Then, the tail distribution of time $T$ between successive in-order decoding instants is,

$$\Pr(T > n) = \sum_{k=0}^{\lceil \sum_{i=1}^{L} n_ir_i \rceil - 1} \Pr(E[n] = k) \Pr(G_n|E[n] = k), \quad (8)$$

$$= \sum_{k=0}^{\lceil nr \rceil - 1} \binom{n}{k} p^k (1-p)^{n-k}, \quad (9)$$

$$= e^{-nD(r\|p)}. \quad (10)$$

Varying the rate of adding packets affects the term $\Pr(G_n|E[n] = k)$ in (8), but it is still $\omega(1/n)$ and we can eliminate it when we take the asymptotic equality in (9). As a result, the in-order delay exponent is same as that if we had a constant rate $r$ of adding new packets to the transmitted stream. Hence we have proved that no other full-rank scheme can achieve a better $(\tau, \lambda)$ trade-off than $V[n] = \lceil nr \rceil$ for all $n$.

Fig. 1 shows the $(\tau, \lambda)$ trade-off for the immediate feedback and no feedback cases, with success probability $p = 0.6$. The optimal trade-off with any feedback delay $d$ lies in between these two extreme cases.

C. General Block-wise Feedback

In Section III-A and Section III-B we considered the extreme cases of immediate feedback $(d = 1)$ and no feedback $(d = \infty)$ respectively. We now analyze the $(\tau, \lambda)$ trade-off with general block-wise feedback delay of $d$ slots. We restrict our attention to a class of coding schemes called time-invariant schemes, which are defined as follows.

**Definition 6** (Time-invariant schemes). A time-invariant scheme is represented by a vector $x = [x_1, \cdots, x_d]$ where $x_i$, for $1 \leq i \leq d$, are non-negative integers such that $\sum_i x_i = d$. In each block we transmit $x_i$ independent linear combinations of the $i$ lowest-index unseen packets in the stream.

The above class of schemes is referred to as time-invariant because the vector $x$ is fixed across all blocks. Note that there is also no loss of generality in restricting the length of the vector $x$ to $d$. This is because each block can provide only up to $d$ innovative coded packets, and hence
there is no advantage in adding more than \( d \) unseen packets to the stream in a given block. Observe that as \( d \to \infty \), the class of time-invariant schemes are equivalent to full-rank codes defined in Definition 5.

**Conjecture 2.** To find the best \((\tau, \lambda)\) trade-off, there is no loss of generality in focusing on time-invariant schemes.

We believe this conjecture is true because, it can be shown that any full-rank code can be expressed as a randomized combination of time-invariant schemes. Thus, if Conjecture 1 is true, it follows that there is no loss of generality in focusing on time-invariant schemes.

Given a vector \( \mathbf{x} \), define \( p_d \), as the probability of decoding the first unseen packet during the block, and \( S_d \) as the number of innovative coded packets that are received during that block. We can express \( \tau_{\mathbf{x}} \) and \( \lambda_{\mathbf{x}} \) in terms of \( p_d \) and \( S_d \) as,

\[
(\tau_{\mathbf{x}}, \lambda_{\mathbf{x}}) = \left( \frac{\mathbb{E}[S_d]}{d}, -\frac{1}{d} \log(1 - p_d) \right),
\]

where we get throughput \( \tau_{\mathbf{x}} \) by normalizing the \( \mathbb{E}[S_d] \) by the number of slots in the slots. We can show that the probability \( \Pr(T > kd) \) of no in-order packet being decoded in \( k \) blocks is equal \((1 - p_d)^k\). Substituting this in (2) we get \( \lambda_{\mathbf{x}} \).

**Example 1.** Consider the time-invariant scheme \( \mathbf{x} = [1, 0, 3, 0] \) where block size \( d = 4 \). That is, we transmit 1 combination of the first unseen packet, and 3 combinations of the first 3 unseen packets. Fig. 2 illustrates this scheme for one channel realization. The probability \( p_d \) and \( \mathbb{E}[S_d] \) are,

\[
p_d = p + (1 - p) \binom{3}{1} p^3 (1 - p)^0 = p + (1 - p)p^3,
\]

\[
\mathbb{E}[S_d] = \sum_{i=1}^{3} i \cdot \binom{4}{i} p^i (1 - p)^{4-i} + 3p^4 = 4p - p^4,
\]

where in (13), we get \( i \) innovative packets if there are \( i \) successful slots for \( 1 \leq i \leq 3 \). But if all 4 slots are successful we get only 3 innovative packets. We can substitute (12) and (13) in (11) to get the \((\tau, \lambda)\) trade-off.
Fig. 2: Illustration of the time-invariant scheme $x = [1, 0, 3, 0]$ with block size $d = 4$. Each bubble represents a coded combination, and the numbers inside it are the indices of the source packets included in that combination. The check and cross marks denote successful and erased slots respectively. The packets that are “seen” in each block are not included in the coded packets in future blocks.

**Remark 1.** Time-invariant schemes with different $x$ can be equivalent in terms of the $(\tau, \lambda)$. In particular, given $x_1 \geq 1$, if any $x_i = 0$, and $x_{i+1} = w \geq 1$, then the scheme is equivalent to setting $x_i = 1$ and $x_{i+1} = w - 1$, keeping all other elements of $x$ the same. This is because the number of independent linear combinations in the block, and the probability of decoding the first unseen is preserved by this transformation. For example, $x = [1, 1, 2, 0]$ gives the same $(\tau, \lambda)$ as $x = [1, 0, 3, 0]$.

In Section III-A we saw that with immediate feedback, we can achieve $(\tau, \lambda) = (p, -\log(1 - p))$. However, with block-wise feedback we can achieve optimal $\tau$ (or $\lambda$) only at the cost of sacrificing the optimality of the other metric. We now find the best achievable $\tau$ (or $\lambda$) with optimal $\lambda$ (or $\tau$).

**Claim 1 (Cost of Optimal Exponent $\lambda$).** With block-wise feedback after every $d$ slots, and in-order decoding exponent $\lambda = -\log(1 - p)$, the best achievable throughput $\tau = (1 - (1 - p)^d)/d$.

**Proof:** If we want to achieve $\lambda = -\log(1 - p)$, we require $p_d$ in (11) to be equal to $1 - (1 - p)^d$. The only scheme that can achieve this is $x = [d, 0, \cdots, 0]$, where we transmit $d$ copies of the first unseen packet. The number of innovative packets $S_d$ received in every block is 1 with probability $1 - (1 - p)^d$, and zero otherwise. Hence, the best achievable throughput is $\tau = (1 - (1 - p)^d)/d$ with optimal $\lambda = -\log(1 - p)$. 

\[ \Box \]
This result gives us insight on how much bandwidth (which is proportional to $1/\tau$) is needed for a highly delay-sensitive application that needs $\lambda$ to be as large as possible.

**Claim 2** (Cost of Optimal Throughput $\tau$). With block-wise feedback after every $d$ slots, and throughput $\tau = p$, the best achievable in-order decoding exponent is $\lambda = -\log(1 - p)/d$.

**Proof:** If we want to achieve $\tau = p$, we need to guarantee an innovation packet in every successful slot. The only time invariant scheme that ensures this is $x = [1, 0, \cdots, 0, d-1]$, or its equivalent vectors $\mathbf{x}$ as given by Remark 1. With $\mathbf{x} = [1, 0, \cdots, 0, d-1]$, the probability of decoding the first unseen packet is $p_d = p$. Substituting this in (11) we get $\lambda = -\log(1 - p)/d$, the best achievable $\lambda$ when $\tau = p$.

Tying back to Fig. 1, Claim 1 and Claim 2 correspond to moving leftwards and downwards along the dashed lines from the optimal trade-off $(p, -\log(1 - p))$. From Claim 1 and Claim 2 we see that both $\tau$ and $\lambda$ are $\Theta(1/d)$, keeping the other metric optimal.

For any given throughput $\tau$, our aim is to find the coding scheme that maximizes $\lambda$. We first prove that any convex combination of achievable points $(\tau, \lambda)$ can be achieved.

**Lemma 1** (Combining of Time-invariant Schemes). By randomizing between time-invariant schemes $\mathbf{x}^{(i)}$ for $1 \leq i \leq B$, we can achieve the throughput-delay trade-off given by any convex combination of the points $(\tau_{\mathbf{x}^{(i)}}, \lambda_{\mathbf{x}^{(i)}})$.

**Proof:** Here we prove the result for $B = 2$, that is randomizing between two schemes. It can be extended to general $B$ using induction. Given two time-invariant schemes $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ that achieve the throughput-delay trade-offs $(\tau_{\mathbf{x}^{(1)}}, \lambda_{\mathbf{x}^{(1)}})$ and $(\tau_{\mathbf{x}^{(2)}}, \lambda_{\mathbf{x}^{(2)}})$ respectively, consider a randomized strategy where, in each block we use the scheme $\mathbf{x}^{(1)}$ with probability $\mu$ and scheme $\mathbf{x}^{(2)}$ otherwise. Then, it is easy to see that the throughput on the new scheme is $\tau = \mu \tau_{\mathbf{x}^{(1)}} + (1 - \mu) \tau_{\mathbf{x}^{(2)}}$.

Now we prove the in-order decoding exponent $\lambda$ is also a convex combinations of $\lambda_{\mathbf{x}^{(1)}}$ and $\lambda_{\mathbf{x}^{(2)}}$. Let $p_{d_1}$ and $p_{d_2}$ be the probabilities of decoding the first unseen packet in a block using scheme $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ respectively. Suppose in an interval with $k$ blocks, we use scheme $\mathbf{x}^{(1)}$ for $h$ blocks, and scheme $\mathbf{x}^{(2)}$ in the remaining blocks, we have

$$\Pr(T > kd) = (1 - p_{d_1})^h (1 - p_{d_2})^{k-h}.$$ (14)
Using this we can evaluate \( \lambda \) as,

\[
\lambda = \lambda_{x^{(1)}} \lim_{k \to \infty} \frac{h}{k} + \lambda_{x^{(2)}} \lim_{k \to \infty} \frac{k-h}{k}
\]

(15)

\[
= \mu \lambda_{x^{(1)}} + (1-\mu) \lambda_{x^{(2)}}
\]

(16)

where we get (15) using (11). As \( k \to \infty \), by the weak law of large numbers, the fraction \( h/k \) converges to \( \mu \).

The main implication of Lemma 1 is that, to find the best \((\tau, \lambda)\) trade-off, we only have to find the points \((\tau_x, \lambda_x)\) that lie on the convex envelope of the achievable region spanned by all possible \( x \).

For general \( d \), it is hard to search for the \((\tau_x, \lambda_x)\) that lie on the optimal trade-off. We propose a set of time-invariant schemes that are easy to analyze and give a good \((\tau, \lambda)\) trade-off. In Theorem 2 we give the \((\tau, \lambda)\) trade-off for the proposed codes and show that for \( d = 2 \) and \( d = 3 \), it is the best trade-off among all time-invariant schemes.

**Definition 7 (Proposed Codes for general \( d \)).** For general \( d \), we propose using the time-invariant schemes with \( x_1 = a \) and \( x_{d-a+1} = d-a \), for \( a = 1, \cdots d \).

In other words, in every block of \( d \) slots, we transmit the first unseen packet \( a \) times, followed by \( d-a \) combinations of the first \( d-a+1 \) unseen packets. These schemes span the \((\tau, \lambda)\) trade-off as \( a \) varies from 1 to \( d \), with a higher value of \( a \) corresponding to higher \( \lambda \) and lower \( \tau \). In particular, observe that the \( a = d \) and \( a = 1 \) codes correspond to codes given in the proofs of Claim 1 and Claim 2.

**Theorem 2 (Throughput-Delay Trade-off for General \( d \)).** The codes proposed in Definition 7 give the trade-off points

\[
(\tau, \lambda) = \left( \frac{1-(1-p)^a + (d-a)p}{d}, -\frac{a}{d} \log(1-p) \right).
\]

(17)

for \( a = 1, \cdots d \). For \( d = 2 \) and \( d = 3 \), the piecewise linear curve joining these points is the best trade-off among all time-invariant schemes.

**Proof:** To find the \((\tau, \lambda)\) trade-off points, we first evaluate \( \mathbb{E}[S_d] \) and \( p_d \). With probability \( 1-(1-p)^a \) we get 1 innovative packet from the first \( a \) slots in a block. The number of innovative
packets received in the remaining \(d - a\) slots is equal to the number of successful slots. Thus, the expected number of innovative coded packets received in the block is

\[
\mathbb{E}[S_d] = 1 - (1 - p)^a + (d - a)p
\]  

(18)

If the first \(a\) slots in the block are erased, the first unseen packet cannot be decoded, even if all the other slots are successful. Hence, we have \(p_d = 1 - (1 - p)^a\). Substituting \(\mathbb{E}[S_d]\) and \(p_d\) in (11), we get the trade-off in (17). By Lemma 1, we can achieve any convex combination of the \((\tau, \lambda)\) points in (17). In Lemma 2 and Lemma 3 below, we prove that for \(d = 2\) and \(d = 3\), the codes proposed in Definition 7 give the best trade-off among all time-invariant schemes.

**Lemma 2.** For \(d = 2\), the codes proposed in Definition 7 give the best \((\tau, \lambda)\) trade-off among all time-invariant schemes.

**Proof:** When \(d = 2\) there are only two possible time-invariant schemes \(x = [2, 0]\) and \([1, 1]\) that give unique \((\tau, \lambda)\). By Remark 1 all other \(x\) are equivalent to one of these vectors in terms \((\tau, \lambda)\). The vectors \(x = [2, 0]\) and \([1, 1]\) correspond to the \(a = 1\) and \(a = 2\) codes proposed in Definition 7. Hence, the line joining their corresponding \((\tau, \lambda)\) points, as shown in Fig. 3, is the best trade-off for \(d = 2\).

**Lemma 3.** For \(d = 3\), the codes proposed in Definition 7 give the best \((\tau, \lambda)\) trade-off among all time-invariant schemes.

**Proof:** When \(d = 3\) there are four time-invariant schemes \(x^{(1)} = [1, 0, 2]\), \(x^{(2)} = [2, 1, 0]\), \(x^{(3)} = [1, 2, 0]\) and \(x^{(4)} = [3, 0, 0]\) that give unique \((\tau, \lambda)\), according to Definition 6 and Remark 1. The vectors \(x^{(1)}\), \(x^{(2)}\) and \(x^{(4)}\) correspond to the codes with \(a = 1, 2, 3\) in Definition 7. The throughput-delay trade-offs \((\tau_{x^{(i)}}, \lambda_{x^{(i)}})\) for \(i = 1, 2, 4\) achieved by these schemes are given by (17). From Claim 1 and Claim 2 we know that \((\tau_{x^{(1)}}, \lambda_{x^{(1)}})\) and \((\tau_{x^{(4)}}, \lambda_{x^{(4)}})\) have to be on the optimal trade-off. By comparing the slopes of the lines joining these points we can show that the point \((\tau_{x^{(2)}}, \lambda_{x^{(2)}})\) lies above the line joining \((\tau_{x^{(1)}}, \lambda_{x^{(1)}})\) and \((\tau_{x^{(4)}}, \lambda_{x^{(4)}})\) for all \(p\). Fig. 3 illustrates this for \(p = 0.6\). For the scheme with \(x^{(3)} = [1, 2, 0]\), we have

\[
(\tau_{x^{(3)}}, \lambda_{x^{(3)}}) = \left(\frac{(3p - p^3)}{3}, -\frac{(\log(1 - p)^2(1 + p))}{3}\right).
\]
Fig. 3: The throughput-delay trade-off of the suggested coding schemes in Definition [7] for $p = 0.6$ and various values of block-wise feedback delay $d$. The trade-off becomes significantly worse as $d$ increases. The point labels on the $d = 2$ and $d = 3$ trade-offs are $x$ vectors of the corresponding codes.

Again, by comparing the slopes of the lines joining $(\tau_{x(i)}, \lambda_{x(i)})$ for $i = 1, \cdots, 4$ we can show that for all $p$, $(\tau_{x(3)}, \lambda_{x(3)})$ lies below the piecewise linear curve joining $(\tau_{x(i)}, \lambda_{x(i)})$ for $i = 1, 2, 4$.

Fig. 3 shows the trade-off given by (17) for different values of $d$. We observe that the trade-off becomes significantly worse as $d$ increases. Thus we can imply that frequent feedback to the source is important in delay-sensitive applications to ensure fast in-order decoding of packets. As $d \to \infty$, and $a = \alpha d$, the trade-off converges to $((1 - \alpha)p, -\alpha \log(1 - p))$ for $0 \leq \alpha \leq 1$, which is the line joining $(0, -\log(1 - p))$ and $(p, 0)$.

In Lemma 2 and Lemma 3 we showed that the codes proposed in Definition [7] give the best trade-off among all time-invariant schemes. Numerical results suggest that even for general $d$ these schemes give a trade-off that is close to the best trade-off among all time-invariant schemes.

Thus, in this section we analyzed how block-wise feedback affects the trade-off between throughput $\tau$ and in-order decoding exponent $\lambda$, which measures the burstiness in-order packet decoding in streaming communication. Our analysis gives us the insight that frequent feedback is crucial for fast in-order packet delivery. Given that feedback comes in blocks of $d$ slots, we present a spectrum of coding schemes that span different points on the $(\tau, \lambda)$ trade-off. Depending
upon the delay-sensitivity and bandwidth limitations of the applications, these codes provide the flexibility to choose a suitable operating point on trade-off.

IV. Multicast Streaming

In this section we analyze the trade-off between throughput $\tau$ and in-order decoding exponent $\lambda$ for a multicast streaming scenario. In [20] we consider the trade-off between $\tau$ and $\sigma$, the expected fraction of slots in which there is in-order packet delivery.

Since each user decodes a different set of packets, the in-order packet required by one user may be redundant to the other user, causing the latter to lose throughput if that packet is transmitted. Thus there is an inter-dependence between throughput and the in-order delivery delay of the two users. We analyze these throughput-delay trade-offs of the two users using a Markov chain model of in-order packet decoding. We propose coding schemes which allow us to tune the level of priority given to each user and hence achieve different points on its throughput-order trade-off.

A. Problem Setup

Consider a source that has to multicast an infinite stream of packets $s_n, n \in \mathbb{N}$ of equal size to $K$ users $U_1, U_2, \cdots, U_K$. We consider an i.i.d. erasure channel to each user such that every transmitted packet is received successfully at user $U_k$ with probability $p_k$, and otherwise received in error and discarded. The erasure events are independent across the users. The theoretical analysis presented in this paper focuses on the two user case. For simplicity of notation in this case, let $a \triangleq p_1p_2$, $b \triangleq p_1(1-p_2)$, $c \triangleq (1-p_1)p_2$ and $d \triangleq (1-p_1)(1-p_2)$, the probabilities of the four possible erasure patterns. The feedback and packet delivery model is same as given in Section II-A. Again we consider the trade-off between the throughput $\tau$ and in-order decoding exponent $\lambda$ defined in Section II-B.

Remark 2. Note that in the case of no feedback case ($d = \infty$) we can directly extend Theorem 1 to show that the optimal throughput-delay trade-off for user $U_k$, $k = 1, 2, \cdots K$ among full-rank codes is $(\tau_k, \lambda_k) = (r, D(r\|p_k))$, if $0 \leq r \leq p_k$. If $r > p_k$ then $\lambda_k = 0$ for user $U_k$. Since we are transmitting a common stream, the rate $r$ of adding new packets to the transmitted packet stream is same for all users.
For the rest of the section we focus on the other extreme case of immediate feedback to the source \((d = 1)\). In this case, the source can use the feedback to give priority to a user which is lagging behind in the in-order packet delivery.

Since the packets have to be delivered in-order to the receiver, ideally every user should be able to decode its next in-order packet, or its “required” packet in every successful slot. The notion of required packets is formally defined as follows.

**Definition 8** (Required packet). The required packet of \(U_i\) is its earliest undecoded packet. Its index is denoted by \(r_i\).

In other words, \(s_{r_i}\) is the first unseen packet of user \(U_i\), where the notion of “seen” packets is given by Definition 1. For example, if packets \(s_1, s_3\) and \(s_4\) have been decoded at user \(U_i\), its required packet \(s_{r_i}\) is \(s_2\).

The best possible trade-off is \((\tau_i, \lambda_i) = (p_i, -\log(1 - p_i))\), and it can be achieved when there is only one user, and the source uses a simple Automatic-repeat-request (ARQ) protocol where it keeps retransmitting the earliest undecoded packet until that packet is decoded. In this paper our objective is to design coding strategies to maximize \(\tau\) and \(\lambda\) for the two user case. For two or more users we can show that it is impossible to achieve the optimal trade-off \((\tau, \lambda) = (p_i, -\log(1 - p_i))\) simultaneously for all users.

We now present code structures that maximize throughput and in-order decoding exponent of the users.

**Claim 3** (Include only Required Packets). In a given slot, it is sufficient for the source to transmit a combination of packets \(s_{r_i}\) for \(i \in \mathcal{I}\) where \(\mathcal{I}\) is some subset of \(\{1, 2, \ldots, K\}\).

**Proof:** Consider a candidate packet \(s_c\) where \(c \neq r_i\) for any \(1 \leq i \leq K\). If \(c < r_i\) for all \(i\), then \(s_c\) has been decoded by all users, and it need not be included in the combination. For all other values of \(c\), there exists a required packet \(s_{r_i}\) for some \(i \in \{1, 2, \ldots, K\}\) that, if included instead of \(s_c\), will allow more users to decode their required packets. Hence, including that packet instead of \(s_c\) gives a higher in-order decoding exponent \(\lambda\). \(\blacksquare\)

**Claim 4** (Include only Decodable Packets). If a coded combination already includes packets \(s_{r_i}\)
with $i \in \mathcal{I}$, and $U_j$, $j \notin I$ has not decoded all $s_{r_i}$ for $i \in \mathcal{I}$, then a scheme that does not include $s_{r_j}$ in the combination gives a better throughput-delay trade-off than a scheme that does.

**Proof:** If $U_j$ has not decoded all $s_{r_i}$ for $i \in \mathcal{I}$, the combination is innovative but does not help decoding an in-order packet, irrespective of whether $s_{r_j}$ is included in the combination. However, if we do not include packet $s_{r_j}$, $U_j$ may be able to decode one of the packets $s_{r_i}$, $i \in \mathcal{I}$, which can save it from out-of-order packet decoding in a future slot. Hence excluding $s_{r_j}$ gives a better throughput-delay trade-off.

**Example 2.** Suppose we have three users $U_1$, $U_2$, and $U_3$. User $U_1$ has decoded packets $s_1$, $s_2$, $s_3$ and $s_5$, user $U_2$ has decoded $s_1$, $s_3$, and $s_4$, and user $U_3$ has decoded $s_1$, $s_2$, and $s_5$. The required packets of the three users are $s_4$, $s_2$ and $s_3$ respectively. By Claim 3, the optimal scheme should transmit a linear combination of one or more of these packets. Suppose we construct combination of $s_4$ and $s_2$ and want to decide whether to include $s_3$ or not. Since user $U_3$ has not decoded $s_4$, we should not include $s_3$ as implied by Claim 4.

The choice of the initial packets in the combination is governed by a priority given to each user in that slot. Claims 3 and 4 imply the following code structure for the two user case.

**Proposition 1** (Code Structure for the Two User Case). Every achievable trade-off between throughput and in-order decoding exponent can be obtained by a coding scheme where the source transmits $s_{r_1}$, $s_{r_2}$ or the exclusive-or, $s_{r_1} \oplus s_{r_2}$ in each slot. It transmits $s_{r_1} \oplus s_{r_2}$ if and only if $r_1 \neq r_2$, and $U_1$ has decoded $s_{r_2}$ or $U_2$ has decoded $s_{r_1}$.

In the rest of this section we analyze the two user case and focus on coding schemes as given by Proposition 1.

**B. Optimal Performance for One of the Users**

In this section we consider that the source always gives priority to one user, called the primary user. We determine the best achievable throughput-smoothness trade-off for a secondary user that is “piggybacking” on such a primary user.

Without loss of generality, suppose that $U_1$ is the primary user, and $U_2$ is the secondary user. Recall that ensuring optimal performance for $U_1$ implies achieving $(\tau_1, \lambda_1) = (p_1, -\log(1-p_1))$. 

<table>
<thead>
<tr>
<th>Time</th>
<th>Sent</th>
<th>$U_1$</th>
<th>$U_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$s_1$</td>
<td>$s_1$</td>
<td>$\times$</td>
</tr>
<tr>
<td>2</td>
<td>$s_2$</td>
<td>$\times$</td>
<td>$s_2$</td>
</tr>
<tr>
<td>3</td>
<td>$s_1 \oplus s_2$</td>
<td>$s_2$</td>
<td>$s_1$</td>
</tr>
<tr>
<td>4</td>
<td>$s_3$</td>
<td>$s_3$</td>
<td>$\times$</td>
</tr>
<tr>
<td>5</td>
<td>$s_4$</td>
<td>$s_4$</td>
<td>$s_4$</td>
</tr>
</tbody>
</table>

Fig. 4: Illustration of the optimal coding scheme when the source always give priority to user $U_1$. The third and fourth columns show the packets decoded at the two users. Cross marks indicate erased slots for the corresponding user.

While ensuring this, the best throughput-delay trade-off for user $U_2$ is achieved by the coding scheme given by Claim 5 below.

**Claim 5 (Optimal Coding Scheme).** A coding scheme where the source transmits $s_{r_1} \oplus s_{r_2}$ if $U_2$ has already decoded $s_{r_1}$, and otherwise transmits $s_{r_1}$, gives the best achievable $(\tau_2, \lambda_2)$ trade-off while ensuring optimal $(\tau_1, \lambda_1)$.

*Proof:* Since $U_1$ is the primary user, the source must include its required packet $s_{r_1}$ in every coded combination. By Proposition 1 if the source transmits $s_{r_1} \oplus s_{r_2}$ if $U_2$ has already decoded $s_{r_1}$, and transmits $s_{r_1}$ otherwise, we get the best achievable throughput-delay trade-off for $U_2$.

Fig. 4 illustrates this scheme for one channel realization.

Packet decoding at the two users with the scheme given by Claim 5 can be modeled by the Markov chain shown in Fig. 5. The state index $i$ can be expressed in terms of the number of gaps in decoding of the users, defined as follows.

**Definition 9 (Number of Gaps in Decoding).** The number of gaps in $U_i$’s decoding is the number of undecoded packets of $U_i$ with indices less than $r_{\text{max}} = \max_i r_i$.

In other words, the number of gaps is the amount by which a user $U_i$ lags behind the user that is leading the in-order packet decoding. The state index $i$, for $i \geq -1$ is equal to the number
Fig. 5: Markov chain model of packet decoding with the coding scheme given by Claim 5, where $U_1$ is the primary user. The state index $i$ represents the number of gaps in decoding of $U_2$ minus that for $U_1$. The states $i'$ are the advantage states where $U_2$ gets a chance to decode its required packet.

of gaps in decoding at $U_2$, minus that for $U_1$. Since the source gives priority to $U_1$, it always has zero gaps in decoding, except when there is a $c = p_2(1 - p_1)$ probability erasure in state 0, which causes the system goes to state $-1$. The states $i'$ for $i \geq 1$ are called “advantage” states and are defined as follows.

Definition 10 (Advantage State). The system is in an advantage state when $r_1 \neq r_2$, and $U_2$ has decoded $s_{r_1}$ but $U_1$ has not.

By Claim 5, the source transmits $s_{r_1} \oplus s_{r_2}$ when the system is in an advantage state $i'$, and it transmits $s_{r_1}$ when the system is in state $i$ for $i \geq -1$. We now describe the state transitions of this Markov chain. First observe that with probability $d = (1 - p_1)(1 - p_2)$, both users experience erasures and the system transitions from any state to itself. When the system is in state $-1$, the source transmits $s_{r_1}$. Since $s_{r_1}$ has been already decoded by $U_2$, the probability $c = p_2(1 - p_1)$ erasure also keeps the system in the same state. If the channel is successful for $U_1$, which occurs with probability $p_1 = a + b$, it fills its decoding gap and the system goes to state 0.

The source transmits $s_{r_1}$ in any state $i$, $i \geq 1$. With probability $a = p_1 p_2$, both users decode $s_{r_1}$, and hence the state index $i$ remains the same. With probability $b = p_1 (1 - p_2)$, $U_1$ receives $s_{r_1}$ but $U_2$ does not, causing a transition to state $i + 1$. With probability $c = (1 - p_1)p_2$, $U_2$ receives $s_{r_1}$ and $U_1$ experiences an erasure due to which the system moves to the advantage state $i'$. 
When the system is an advantage state, having decoded $s_{r_1}$ gives $U_2$ an advantage because it can use $s_{r_1} \oplus s_{r_2}$ transmitted in the next slot to decode $s_{r_2}$. From state $i'$, with probability $a$, $U_1$ decodes $s_{r_1}$ and $U_2$ decodes $s_{r_2}$, and the state transitions to $i - 1$. With probability $c$, $U_2$ decodes $s_{r_2}$, but $U_1$ does not decode $s_{r_1}$. Thus, the system goes to state $(i - 1)'$, except when $i = 1$, where it goes to state 0.

We now solve for the steady-state distribution of this Markov chain. Let $\pi_i$ and $\pi_i'$ be the steady-state probabilities of states $i$ for $i \geq -1$ and advantages states $i'$ for all $i \geq 0$ respectively. The steady-state transition equations are given by

\begin{align}
(1 - a - d)\pi_i &= b(\pi_{i-1} + \pi_i') + a\pi_{i+1} \quad \text{for } i \geq 1, \\
(1 - d)\pi_i' &= c(\pi_i + \pi_{i+1}) \quad \text{for } i \geq 1, \\
(1 - c - d)\pi_{-1} &= c(\pi_0 + \pi_1'), \\
(1 - a - d)\pi_0 &= a\pi_1' + (a + b)\pi_{-1}.
\end{align}

By rearranging the terms in (19)-(22), we get the following recurrence relation,

$$
\pi_i = \frac{(1 - a - d)}{c} \pi_{i-1} - \frac{b}{c} \pi_{i-2} \quad \text{for } i \geq 2.
$$

(23)

Solving the recurrence in (23) and simplifying (19)-(22) further, we can express $\pi_i, \pi_i'$ for $i \geq 2$ in terms of $\pi_1$ as follows,

\begin{align}
\frac{\pi_i}{\pi_{i-1}} &= \frac{b}{c}, \\
\frac{\pi_i'}{\pi_i} &= \frac{c}{a + c}.
\end{align}

(24) (25)

From (24) we see that the Markov chain will be positive-recurrent and a unique steady-state distribution exists only if $b < c$, which is equivalent to $p_1 < p_2$. If $p_1 \geq p_2$, the expected recurrence time to state 0, that is the time taken for $U_2$ to catch up with $U_1$ is infinity. When the Markov chain is positive recurrent, we can use (24) and (25) to solve for all the steady state probabilities.

**Claim 6 (Trade-off for the Piggybacking user).** When the source always gives priority to user $U_1$ such that it achieves the optimal trade-off $(\tau_1, \lambda_1) = (p_1, -\log(1 - p_1))$, the scheme in Claim 5
gives the best achievable \((\tau_2, \lambda_2)\) trade-off for piggybacking user \(U_2\),

\begin{align}
\tau_2 &= \min(p_1, p_2) \\
\lambda_2 &= -\log \left( \max \left( \frac{1 - c + d + \sqrt{(1 - c + d)^2 + 4(bc + cd - d)}}{2}, 1 - p_1 \right) \right). 
\end{align}

**Proof:** Since we always give priority to the primary user \(U_1\), we have \((\tau_1, \lambda_1) = (p_1, -\log(1 - p_1))\). When \(p_1 < p_2\), we can express the throughput \(\tau_2\) in terms of the steady state probabilities of the Markov chain in Fig. 5. User \(U_2\) experiences a throughput loss when it is in state \(-1\) and the next slot is successful. Thus, when \(p_2 > p_1\),

\begin{align}
\tau_2 &= p_2(1 - \pi_{-1}), \\
&= p_2 \left( 1 - \frac{c - b}{a + c} \right) = p_1.
\end{align}

If \(p_1 \geq p_2\), the system drifts infinitely to the right side and \(\pi_{-1}\) is zero. Thus

\[\tau_2 = p_2,
\]

which is equal to the capacity of the erasure channel to \(U_2\).

To determine \(\lambda_2\), first observe that \(U_2\) decodes an in-order packet when the system is in state 0 or states \(i'\), for \(i \geq 1\), and the next slot is successful. As given by Definition 4, the in-order decoding exponent \(\lambda_2\) is the asymptotic decay rate of \(\Pr(T > t)\), the probability that no in-order packet is decoded by \(U_2\) for \(t\) consecutive slots. To determine \(\lambda_2\), we add an absorbing state \(F\) to the Markov chain as shown in Fig. 6 such that the system transitions to \(F\) when an in-order packet is decoded by \(U_2\).

In Fig. 6 all the states \(i\) and \(i'\) for \(i \geq 1\) are fused into states \(I\) and \(I'\) because this does not affect the probability distribution of the time to reach the absorbing state \(F\). The in-order decoding exponent \(\lambda_2\) is equal to the rate of convergence of this Markov chain to its steady state, which is known to be (see [21, Chapter 4]) \(\lambda_2 = -\log \xi_2\) where \(\xi_2\) is the second largest
Fig. 6: Markov model used to determine the in-order decoding exponent $\lambda_2$ of user $U_2$. The absorbing state $F$ is reached when an in-order packet is decoded by $U_2$. The exponent of the distribution of the time taken to reach this state is $\lambda_2$.

eigenvalue of the state transition matrix of the Markov chain,

$$A = \begin{pmatrix}
    d & b & 0 & 0 & a+c \\
    0 & a+b+d & c & 0 & 0 \\
    0 & b & d & 0 & a+c \\
    a+b & 0 & 0 & c+d & 0 \\
    0 & 0 & 0 & 0 & 1
\end{pmatrix}. \quad (31)$$

Solving for the second largest eigen-value of $A$, we can show that

$$\xi_2 = \max \left( 1 - p_1, \frac{1 - c + d + \sqrt{(1 - c + d)^2 + 4(bc + cd - d)}}{2} \right). \quad (32)$$

Hence the in-order decoding exponent $\lambda_2 = -\log \xi_2$ is as given by (27).

In Fig. 7 we plot the throughput $\tau_2$, and the in-order decoding exponent $\lambda_2$ as $p_2$ varies from 0 to 1, for different values of $p_1$. When $p_2 < p_1$, both $\tau_2$ and $\lambda_2$ increase with $p_2$. But for all $p_2 \geq p_1$, the throughput is fixed at $p_1$ and $\lambda_2$ increases, which gives the vertical line at the end of each curve. From Fig. 7 we can infer that the secondary user $U_2$ achieves high $\lambda_2$ only in the regime $p_1 < p_2$, when it has a better channel than the primary user.
C. General Throughput-delay Trade-offs

For the general case, we propose coding schemes that can be combined to tune the priority given to each of the two users and achieve different points on their throughput-delay trade-offs.

Let $r_{\text{max}} = \max(r_1, r_2)$ and $r_{\text{min}} = \min(r_1, r_2)$, where $r_1$ and $r_2$ are the indices of the required packets of the two users. We refer to the user with the higher index $r_i$ as the leader(s) and the other user as the lagger. Thus, $U_1$ is the leader and $U_2$ is the lagger when $r_1 > r_2$. If $r_1 = r_2$, without loss of generality we consider $U_1$ as the leader.

**Definition 11** (Priority-$(q_1, q_2)$ Codes). If the lagger $U_i$ has not decoded packet $s_{r_{\text{max}}}$, the source transmits $s_{r_{\text{min}}}$ with probability $q_i$ and $s_{r_{\text{max}}}$ otherwise. If the lagger has decoded $s_{r_{\text{max}}}$, the source transmits $s_{r_{\text{max}}} \oplus s_{r_{\text{min}}}$.

Note that the code given in Claim 5, where the source always gives priority to user $U_1$ is a special case of priority-$(q_1, q_2)$ codes with $(q_1, q_2) = (1, 0)$. Another special case is $(q_1, q_2) = (0, 0)$ which is a greedy coding scheme that always favors the user which is ahead in in-order decoding. The greedy coding scheme ensures throughput optimality to both users, i.e. $\tau_1 = p_1$ and $\tau_2 = p_2$.

The Markov model of packet decoding with a priority-$(q_1, q_2)$ code is as shown in Fig. 8, which is a two-sided version of the Markov chain in Fig. 5. Same as in Fig. 5, the index $i$ of
Fig. 8: Markov chain model of packet decoding with the priority- \((q_1, q_2)\) coding scheme given by Definition [11]. The state index \(i\) represents the number of gaps in decoding if \(U_2\) compared to \(U_1\) and \(q_i\) is the probability of giving priority to the \(U_i\) when it is the lagger, by transmitting its required packet \(s_{r_i}\), and

a state \(i\) of the Markov chain is the number of gaps in decoding of \(U_2\) minus that for \(U_1\). User \(U_1\) is the leader when the system is in state \(i \geq 1\) and \(U_2\) is the leader when \(i \leq -1\), and both users are leaders when \(i = 0\). The system is in the advantage state \(i'\) if packet is decoded by the lagger but not the leader.

For simplicity of representation we define the notation \(\bar{d} \triangleq 1 - d\), \(\bar{q}_1 \triangleq 1 - q_1\) and \(\bar{q}_2 \triangleq 1 - q_2\). The state-transition equations of the Markov chain are as follows.

\[
(d - a)\pi_0 = d(\pi_1 + \pi_{-1}) + q_1(a + b)\pi'_{-1} + q_2(a + c)\pi_1 \tag{33}
\]

\[
(d - \bar{q}_2a - q_2b)\pi_i = q_2(a + c)\pi_{i+1} + \bar{q}_2b\pi_{i-1} + b\pi'_i + a\pi'_{i+1} \quad \text{for } i \geq 2 \tag{34}
\]

\[
(d - \bar{q}_1a - q_1c)\pi_i = q_1(a + b)\pi_{i+1} + \bar{q}_1c\pi_{i+1} + c\pi'_i + a\pi'_{i-1} \quad \text{for } i \leq -2 \tag{35}
\]

\[
(d - \bar{q}_2a - q_2b)\pi_1 = q_2(a + c)\pi_2 + b(\pi'_1 + \pi_0 + \pi'_1) + a\pi_2' \tag{36}
\]

\[
(d - \bar{q}_1a - q_1c)\pi_{-1} = q_1(a + b)\pi_0 + c(\pi'_1 + \pi_0 + \pi'_1) + a\pi'_{-2} \tag{37}
\]

\[
\bar{d}\pi'_i = \bar{q}_2c\pi_i + c\pi'_{i+1} \quad \text{for } i \geq 1 \tag{38}
\]

\[
\bar{d}\pi'_i = \bar{q}_2c\pi_i + b\pi'_{i+1} \quad \text{for } i \leq -1 \tag{39}
\]

Rearranging the terms, we get the following recurrence in the steady-state probabilities on the
right-side of the chain,
\[ \alpha_3 \pi_{i+3} + \alpha_2 \pi_{i+2} + \alpha_1 \pi_{i+1} + \alpha_0 \pi_i = 0 \quad \text{for } i \geq 1 \quad (40) \]

where,
\[ \alpha_3 = c(a + c)q_2 \quad (41) \]
\[ \alpha_2 = -cd + bcq_2 - (a + c)q_2 \bar{d} \quad (42) \]
\[ \alpha_1 = \bar{d}(\bar{d} - bq_2 - a\bar{q}_2) \quad (43) \]
\[ \alpha_0 = -d\bar{b}\bar{q}_2 \quad (44) \]

The characteristic equation of this recurrence has the roots \( 1, \rho \) and \( \rho' \). We can show that both \( \rho \) and \( \rho' \) are positive and at least one of them is greater than 1. The expression for the smaller root is,
\[ \rho = -\frac{\alpha_3 + \alpha_2}{2\alpha_3} - \frac{\sqrt{(\alpha_3 + \alpha_2)^2 + 4\alpha_3\alpha_0}}{2\alpha_3} \quad (45) \]

The Markov chain is positive recurrent and a unique steady state distribution exists when \( \rho < 1 \) which is equivalent to,
\[ 2\sqrt{(\alpha_3 + \alpha_2)^2 + 4\alpha_3\alpha_0} > -(1 + 4\alpha_3\alpha_0) \quad (46) \]

Thus, when \( \rho < 1 \), the steady-state probabilities \( \pi_i \) and \( \pi'_i \) for \( i \geq 1 \) are related by the recurrences,
\[ \frac{\pi_{i+1}}{\pi_i} = \rho \quad \text{and} \quad \frac{\pi'_i}{\pi_i} = \frac{c(1 - q_2)}{\bar{d} - c\rho} \quad (47) \]

Similarly, for the left side of the chain we have the recurrences,
\[ \frac{\pi_{i+1}}{\pi_i} = \mu \quad \text{and} \quad \frac{\pi'_i}{\pi_i} = \frac{b(1 - q_1)}{\bar{d} - b\mu} \quad (48) \]

where
\[ \mu = -\frac{\beta_3 + \beta_2}{2\beta_3} - \frac{\sqrt{(\beta_3 + \beta_2)^2 + 4\beta_3\beta_0}}{2\beta_3} \quad (49) \]

with the expressions \( \beta_k \) for \( k = 0, 1, 2, 3 \) being the same as \( \alpha_k \) with \( b \) and \( c \) interchanged, and \( q_2 \) replaced by \( q_1 \). We can use these recurrences we can express all steady-state probabilities \( \pi_i \) and \( \pi'_i \) for \( i \geq 1 \) in terms of \( \pi_1 \), and the steady-state probabilities \( \pi_i \) and \( \pi'_i \) for \( i \leq -1 \) in
terms of $\pi_{-1}$. Then using the states transition equation (33), and the fact that all the steady state probabilities sum to 1, we can solve for all the steady-state probabilities of the Markov chain.

**Claim 7** ($(\tau, \lambda)$ Trade-offs with Priority-$(q_1, q_2)$ codes). Using the priority-$(q_1, q_2)$ codes given by Definition 11, the two users achieve the throughput-delay trade-offs as given by

$$\tau_2 = p_2 \left(1 - \frac{q_1 \pi_{-1}}{1 - \mu}\right)$$  \hspace{1cm} (50)

$$\lambda_2 = -\log \max \left(d + q_1 c + \bar{q}_1 b, \frac{2d + \bar{q}_2 a + b + \sqrt{(2d + \bar{q}_2 a + b)^2 + 4(bc + cd - d + daq_2)}}{2}\right)$$  \hspace{1cm} (51)

The expressions for throughput $\tau_1$ and in-order decoding exponent $\lambda_1$ of $U_1$ are same as (50) and (51) with $b$ and $c$, and $q_1$ and $q_2$ interchanged, $\pi_{-1}$ replaced by $\pi_1$, and $\mu$ replaced by $\rho$.

**Proof:** User $U_1$ receives an innovative in every successful slot except when the source (with probability $q_2$) gives priority to $U_2$ in states $i$, $i \geq 1$. Thus, its throughput is given by

$$\tau_1 = p_1 \left(1 - q_2 \sum_{i=1}^{\infty} \pi_i\right) = p_1 \left(1 - \frac{q_2 \pi_1}{1 - \rho}\right)$$  \hspace{1cm} (52)

Similarly we have,

$$\tau_2 = p_2 \left(1 - q_1 \sum_{i=-\infty}^{-1} \pi_i\right) = p_2 \left(1 - \frac{q_1 \pi_{-1}}{1 - \mu}\right)$$  \hspace{1cm} (53)

Similar to Section IV-B we determine the in-order decoding exponent $\lambda_2$ of user $U_2$ by adding an absorbing state $F$ to the Markov chain as shown in Fig. 9 such that the system transitions to $F$ when an in-order packet is decoded by $U_2$. In Fig. 9 all the states $i$ and $i'$ for $i \geq 1$ are fused into states $I$ and $I'$ because this does not affect the probability distribution of the time to reach the absorbing state $F$. The in-order decoding exponent $\lambda_2 = -\log \xi_2$ where $\xi_2$ is the second largest eigenvalue of its state transition matrix which is given by,

$$\xi_2 = \max \left(d + q_1 c + \bar{q}_1 b, \frac{2d + \bar{q}_2 a + b + \sqrt{(2d + \bar{q}_2 a + b)^2 + 4(bc + cd - d + daq_2)}}{2}\right).$$  \hspace{1cm} (54)

The in-order decoding exponent $\lambda_2 = -\log \xi_2$ and is given by (51). The expression for the in-order decoding exponent $\lambda_1$ of user $U_1$ is same as (51) with $b$ and $c$, and $q_1$ and $q_2$ interchanged.
In Fig. 9, we plot the throughput-delay trade-off for user $U_2$ with $p_1 = 0.5$, $q_1 = 1$ and with different values of $p_2$. Along each curve, $q_2$ increases from 0 to 1. As $q_2$ increases, the source gives priority to $U_2$ with higher probability resulting in a higher $\lambda_2$, but at the cost of $\tau_2$, as well as $\tau_1$ (not shown in the plot). Since $q_1$ is fixed at 1, the in-order decoding exponent of user $U_1$ is $\lambda_1 = -\log(1 - p_1)$, its optimal value. In Fig. 11, we show the effect of increasing $q_1$ and $q_2$ simultaneously for different values of $p_1 = p_2$. As $q_1 = q_2$ increases, we get a better in-order decoding exponent for both users, but at the cost of loss of throughput.

**Conjecture 3** (Optimality of Priority-$q$ codes). *Having a different value of $q$ for different state indices does not improve the throughput-delay trade-off.*

**V. Concluding Remarks**

**A. Major Implications**

In this paper we consider the problem of streaming over an erasure channel when the packets are required in-order by the receiver application. We investigate the trade-off between the throughput $\tau$ and the smoothness of in-order packet decoding, which is measured by the in-order decoding exponent $\lambda$. 
We first study the effect of block-wise feedback on the throughput-delay trade-off for point-to-point streaming. Our analysis shows that frequent feedback is crucial to achieving smooth in-order packet delivery. Given that block-wise feedback delay, we present a spectrum of coding schemes that span different points on the \((\tau, \lambda)\) trade-off. Depending upon the delay-sensitivity and bandwidth limitations of the applications, these codes provide the flexibility to choose a suitable operating point on trade-off.

Next we consider the problem where multiple users are streaming data from the source over a broadcast channel, over independent erasure channels with immediate feedback. Since different
users decode different sets of packets, the source has to strike balance between giving priority to ensuring in-order packet decoding at each of the users. We study the inter-dependence between the throughput-delay trade-offs for the case of two users and develop coding schemes to tune the priority given to each user.

B. Future Perspectives

In this paper, we assume strict in-order packet delivery, without allowing packet dropping, which can improve the in-order decoding exponent. Determining how significant this improvement is remains to be explored. Another possible future research direction is to extend the multicast streaming analysis to more users using the Markov chain-based techniques developed here for the two user case. A broader research direction is to consider the problem of streaming from distributed sources that are storing overlapping sets of the data. Using this diversity can improve the delay in decoding each individual packet.

REFERENCES