

Online Versus Offline Rate in Streaming Codes for Variable-Size Messages

Michael Rudow^{ID} and K. V. Rashmi^{ID}, Member, IEEE

Abstract— One pervasive challenge in providing a high quality-of-service for live communication is to recover lost packets in real-time. Streaming codes are a class of erasure codes that are designed for such strict, low-latency streaming communication settings. Motivated by applications that transmit messages whose sizes vary over time, such as live video streaming, this paper considers the setting of streaming codes under variable-size messages. In practice, streaming codes operate in an “online” setting where the sizes of the future messages are unknown. “Offline” codes, in contrast, have access to the sizes of all messages, including future ones. This paper introduces the first online rate-optimal streaming codes for communicating over a burst-only packet loss channel for two broad parameter regimes. These two online codes match the rates of optimal offline codes for the two settings despite the apparent advantage of the offline setting. This paper further establishes that online codes cannot attain the optimal rate for offline codes for all remaining parameter settings.

Index Terms— Streaming codes, erasure coding, live-streaming applications, videoconferencing, packet loss.

I. INTRODUCTION

REAL-TIME communication with a high quality-of-service is critical for many pervasive streaming applications, including VoIP and videoconferencing. These live streaming applications rely on transmitting packets of information and contend with packet losses during transmission. Although lost packets can be recovered via retransmission, this solution is often infeasible due to strict latency constraints [2]. Therefore, real-time streaming applications often use forward error correction to provide robustness to packet losses. However, using traditional coding schemes to comply with the real-time delay constraint penalizes the rate.

Coding schemes explicitly designed for live streaming communication can attain significantly higher rates than traditional ones, such as maximal distance separable block codes. This improved performance was demonstrated in [3], where the authors proposed a new “streaming model” for real-time communication shown in Figure 1. Under this streaming model,

Manuscript received 3 May 2022; accepted 28 January 2023. Date of publication 14 February 2023; date of current version 19 May 2023. This work was supported in part by NSF under Grant CCF1910813. An earlier version of this paper was presented in part at the 2020 IEEE International Symposium on Information Theory [DOI: 10.1109/ISIT44484.2020.9174458]. (Corresponding author: Michael Rudow.)

The authors are with the Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: mhrudow@gmail.com).

Communicated by R. D. Wesel, Associate Editor for Coding Techniques.

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TIT.2023.3244799>.

Digital Object Identifier 10.1109/TIT.2023.3244799

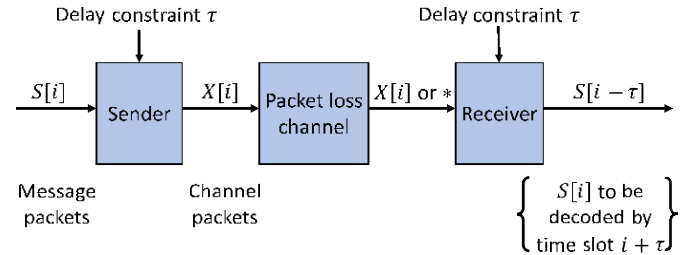


Fig. 1. Overview of the streaming model.

at each time slot i , a sender receives a “message packet” $S[i]$ and transmits a “channel packet” $X[i]$ over a packet loss channel to a receiver. The message packet $S[i]$ is to be decoded at the receiver within delay τ , i.e., by time slot $(i + \tau)$. The authors established an upper bound on the rate, and they introduced a rate-optimal construction for certain settings. Later, a rate-optimal construction for all remaining settings was presented in [4]. Numerous subsequent works have also studied variants of the streaming model (discussed in Section II) [1], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [21], [22], [23], [24], [25], [26], [27].

The streaming model proposed in [3] and studied further in several subsequent works [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [21], [22], [23], [24], [25], [26], [28] considers a setting where all message packets comprise some fixed number of symbols. However, many applications must send a stream of variable-size message packets. For example, video calls consist of compressed video frames of fluctuating sizes. Consequently, a new streaming model incorporating variable-size message packets was introduced in [27].

The streaming model with variable-size message packets differs from that of fixed-size message packets in two key ways: First, the sequence of sizes of message packets affects the optimal rate. In fact, the variability in the sizes of message packets negatively impacts the optimal rate, which is never higher than that of the setting where message packets have fixed sizes [27]. Second, while there are rate-optimal schemes that send each message packet in the corresponding channel packet for the setting of fixed-size message packets, spreading message symbols over multiple channel packets is advantageous in the setting of variable-size message packets. This is because sending a large message packet within a single channel packet leads to many lost symbols when that

channel packet is lost. Spreading message symbols intelligently reduces the maximum number of message symbols lost in a burst—a lower bound on how much redundancy is needed. In contrast, when all message packets are the same size and are sent in the corresponding channel packets, all bursts drop the same number of message symbols. As such, spreading message symbols over multiple channel packets does not offer an advantage.

When the transmission is lossless, sending message symbols over multiple channel packets increases the latency compared to sending each message packet within the corresponding channel packet. In [27], the authors introduce a new delay constraint that captures the trade-off between the rate and the decoding delay in lossless transmission, called the lossless-delay constraint. Specifically, when there are no losses, the receiver must decode each message packet with a delay of τ_L time slots, where τ_L is less than τ .¹

One key challenge in realizing the benefits of spreading is determining how to best spread message symbols over one or more channel packet(s) despite the fact that future message packets' sizes are inherently variable and unknown. For example, a large message packet should be sent in the corresponding channel packet when the next several message packets are even larger to reduce the variability in the sizes of channel packets. In contrast, message symbols of a large message packet should be spread over multiple channel packets when the subsequent several message packets are small. Thus, the optimal strategy for encoding depends on the sizes of future message packets. To capture this dependency introduced by the variability in the size of message packets, the coding schemes can be classified into two classes: (a) "offline" schemes and (b) "online" schemes. Offline coding schemes have access to the sizes of message packets of future time slots, whereas online schemes do not have access to such information. Online constructions are of practical interest, as the sizes of future message packets are typically unknown in live streaming applications. By using future information, optimal offline constructions can always match, and potentially significantly exceed, the rate of online ones. Therefore, a natural question is: "can online coding schemes match the rate of offline coding schemes?"

Main contributions. In this paper, we design the first rate-optimal online coding schemes for two classes of parameter settings. In "Regime 1," b and τ may take any values while $\tau_L = 0$, necessitating that all constructions recover each message packet immediately under lossless conditions—a useful property exhibited by existing rate-optimal constructions [3], [4] for the streaming model where all message packets have the same size. This broad regime is well-suited for applications that require minimal latency during lossless conditions and can tolerate extra latency only during occasional losses. Our rate-optimal construction is systematic, and it sends each message packet in the corresponding channel packet. During each time slot, i , we combine two new methodologies to alleviate the variability. (a) We apply a greedy paradigm for delaying transmitting the parity symbols associated with $S[i]$ until the time slot $(i + \tau)$. (b) We define the number

of parity symbols to be sent in $X[i + \tau]$ while deferring defining the parity symbols themselves until the time slot $(i + \tau)$ to make use of the sizes of message packets $S[i + 1], \dots, S[i + \tau - 1]$. The construction is rate-optimal, even for the offline setting. To prove the construction's optimality, we show that it cumulatively sends no more symbols by each time slot than any offline rate-optimal construction that satisfies the worst-case-delay and lossless-delay constraints. As such, the results show that non-systematic schemes provide no advantage. In "Regime 2," $\tau_L = (\tau - b)$ and $b|\tau$, so τ_L has its maximum value. Here, we show that a simple scheme that encodes each message packet separately matches an upper bound on the rate. Thus, the above results together show that online coding schemes can match the rate of optimal offline coding schemes for two broad parameter regimes even though knowledge about the sizes of future message packets appears advantageous. In addition, we demonstrate that online coding schemes necessarily have lower rates than optimal offline coding schemes for all remaining parameter regimes.

The organization of the paper is as follows. We begin by introducing the model and background in Section II. We then present online constructions that match the optimal rate of offline constructions for two parameter regimes in Section III. Next, we show that the rate of optimal online schemes cannot match that of offline schemes for all remaining settings in Section IV. Finally, we end with a discussion on conclusions and future directions in Section V.

II. BACKGROUND, SYSTEM MODEL, AND RELATED WORK

We begin this section by discussing the background on streaming codes that led to the model considered in this work. We then present the model in detail, as well as the notation used throughout this paper. Finally, we discuss related work on streaming codes.

A. Background

Martinian and Sundberg proposed the streaming model in [3]. It captures the setting of real-time communication of a sequence of message packets of a fixed size over a burst-only packet loss channel. At each time slot i , a sender receives a message packet, $S[i]$, comprising k symbols drawn uniformly at random from a finite field F_q . The sender then transmits to a receiver a channel packet, $X[i]$, consisting of n symbols from F_q over a burst-only channel. Due to real-time latency constraints, the receiver must decode $S[i]$ within a delay of τ time slots (that is, using the channel packets received by time slot $(i + \tau)$). The lossy channel is denoted $C(b, \tau)$ and introduces bursts of length at most b followed by guardspaces of length at least τ . The authors showed an upper bound on the rate of streaming codes of $\frac{\tau}{\tau + b}$ and introduced a class of code constructions applicable to the streaming model, called "streaming codes," meeting this bound for some settings of τ and b . Later, a construction proposed in [4] met this bound, showing that $\frac{\tau}{\tau + b}$ is the capacity for the remaining settings of τ and b .

In applications such as video communication, the sizes of messages fluctuate considerably. Consequently, in [27] a

¹The lossless-delay constraint was denoted as τ_G in [27].

streaming model was introduced that incorporates variable-size messages. The authors showed that $\frac{\tau}{\tau+b}$ remains an upper bound on the rate. The authors also present a streaming code for this new setting, and via an empirical evaluation, show that the construction attains a rate of approximately 89.5% of the upper bound on rate of $\frac{\tau}{\tau+b}$ for the settings considered in the empirical evaluation. The authors also bounded the gap between the construction and $\frac{\tau}{\tau+b}$ when the sizes of message packets are drawn independently from a distribution. The smaller the variance of the distribution, the smaller the gap. However, the gap is nontrivial, and the sizes of message packets for real-time streaming applications are typically not independent.

B. System Model

We consider the streaming model from [27], which considers variable-size message packets (with a few minor changes in how time slots are indexed). During each time slot i the message packet, $S[i]$, comprises $k_i \in \{0, \dots, m\}$ symbols for a natural number m representing the maximum possible size of a message packet. The sender transmits a channel packet, $X[i]$, comprising n_i symbols. The receiver obtains

$$Y[i] = \begin{cases} X[i] & \text{if } X[i] \text{ is received} \\ \emptyset & \text{if } X[i] \text{ is lost.} \end{cases}$$

Transmission occurs over a $C(b, \tau)$ channel. Each channel packet, $X[i]$, depends only on the symbols of previous message packets (i.e. $S[0], \dots, S[i]$). Similar to the model of fixed-size message packets, each $S[i]$ must be decoded by time slot $(i + \tau)$; this requirement is called the worst-case-delay constraint.

Recall from Section I that under the setting of variable-size message packets, spreading message symbols over multiple channel packets can be advantageous. As such, there is an inherent tradeoff between the rate of a code and the decoding delay under lossless transmission (i.e., the number of time slots needed to decode a message packet when all channel packets are received). A new delay constraint capturing this trade-off, called the lossless-delay constraint, was introduced in [27]. When there are no losses, the receiver must decode each message packet $S[i]$ within a delay of $\tau_L (< \tau)$ time slots. The lossless-delay constraint is relevant for applications that can infrequently tolerate a delay of τ in the worst case but require faster decoding for most message packets.

The valid value ranges for the parameters b, τ , and τ_L are $1 \leq b \leq \tau$ and $0 \leq \tau_L \leq (\tau - b)$. A maximum burst length of 0 is omitted because coding is unnecessary for lossless transmission. Furthermore, reliable transmission is impossible when b exceeds τ , since $S[i]$ cannot be decoded by time slot $(i + \tau)$ when $X[i], \dots, X[i + \tau]$ are all lost in a burst. Intrinsically, τ_L cannot be negative, and $S[i]$ is decoded by time slot $(i + \tau - b)$ if there are no losses, since a burst can drop $X[i + \tau - b + 1], \dots, X[i + \tau]$. Since $b > 0$, this means that τ_L is without loss of generality strictly less than τ .

In the setting where message packets all have size k and channel packets all have size n [3], the rate is $\frac{k}{n}$. However, the setting of varying sizes of message packets and channel

packets, necessitates a new definition of rate. The rate is defined [27] for a finite stream of $(t + 1)$ message packets for an arbitrary natural number t as the number of message symbols divided by the number of transmitted symbols:

$$R_t = \frac{\sum_{i=0}^t k_i}{\sum_{i=0}^t n_i}$$

Recall that the rate is at most $\frac{\tau}{\tau+b}$. However, depending on the sizes of the message packets, the upper bound can be loose.

Constructions that during the time slot $i \in \{0, \dots, t\}$ can access all future message packets' sizes (i.e., k_{i+1}, \dots, k_t) are called "offline." Offline schemes have access to the sizes but not the symbols of the future message packets. In contrast, code constructions that do not know the sizes of the future message packets are dubbed "online." Specifically, during time slot i , (k_{i+1}, \dots, k_t) are unknown for an online construction. We distinguish between the feasible rates for offline and online coding schemes. The best possible rate for offline coding schemes is called the "offline-optimal-rate" and for online coding schemes is called the "online-optimal-rate."

Encoding during time slot i is defined as

$$X[i] = \text{Enc}(S[0], \dots, S[i]). \quad (1)$$

To distinguish between online and offline decoding, we use the following quantity to denote the last time slot for which the size of message packets is available to the receiver

$$\lambda_i = \begin{cases} t & \text{if offline} \\ \arg \max_{l \in \{i, \dots, i+\tau\}} 1[Y[l] = X[l]] & \text{if online.} \end{cases}$$

The decoding for message packet $S[i]$ is then defined for two scenarios. First, in a lossless transmission, $S[i]$ is decoded using (a) the previously decoded message packets, (b) the $(\tau_L + 1)$ channel packets received within lossless-delay, and (c) the sizes of the first $(i + \tau_L + 1)$ message packets as follows:

$$S[i] = \text{Dec}^{(L)}(S[0], \dots, S[i - 1], X[i], \dots, X[i + \tau_L], k_0, \dots, k_{i+\tau_L}). \quad (2)$$

Second, when losses occur, $S[i]$ is decoded using (a) the previously decoded message packets, (b) all received channel packets among the $(\tau + 1)$ sent within the worst-case-delay, and (c) the sizes of the first $(\lambda_i + 1)$ message packets as follows:

$$S[i] = \text{Dec}(S[0], \dots, S[i - 1], Y[i], \dots, Y[i + \tau], k_0, \dots, k_{\lambda_i+\tau}). \quad (3)$$

To ensure that the receiver knows the sizes of message packets, a small header containing k_{i-b}, \dots, k_i is added to $X[i]$.² Finally, we note that our work's constructions do not need as much memory as is acceptable under the model. During any time slot, i , the sizes and symbols of message packets and channel packets from before time slot $(i - \tau)$ are not used.

The capacity is defined for any given message size sequence, k_0, \dots, k_t , as the highest rate that can be attained while satisfying Equations 1, 2, and 3.

²In the edge conditions, $(i - \tau)$ is set to 0 for $i < \tau$ and $(i + \tau)$ is set to t for $(i - \tau) > (t - \tau)$.

This paper uses the following notation. The term $[n]$ denotes $\{0, \dots, n\}$. All vectors are row vectors. A vector V has length v and is indexed as $V = (V_0, \dots, V_{v-1})$. For $I = \{i_0, \dots, i_l\} \subseteq [v-1]$ where $i_j < i_{j'}$ for $j < j' \in [l]$, $V_I = (V_{i_0}, \dots, V_{i_l})$. Let A be an $n \times n$ matrix, and $I \subseteq [0, \dots, n-1]$. Then A_I is A restricted to the columns in I . This work refers to k_0, \dots, k_t as the “message size sequence.”

This work uses the following conventions. The sizes of the final τ message packets are each 0, and t is at least τ . Thus, the coding schemes can encode the final message packet of non-zero size using τ extra channel packets. To satisfy this restriction, one can append τ message packets of size 0 to the stream of messages, which will not change the optimal rate.

For $i \in \{1-b, \dots, -1\} \cap \{t+1, \dots, t+b+1\}$, k_i is defined as 0. For $i \in \{1-b, \dots, -1\}$, a burst loss of $X[i], \dots, X[i+b-1]$ denotes a burst loss of $X[0], \dots, X[i+b-1]$. Similarly, for $i \in \{t-b+2, \dots, t\}$ a burst loss of $X[i], \dots, X[i+b-1]$ denotes a burst loss of $X[i], \dots, X[t]$.

C. Other Related Works

Numerous existing works have examined different variations of the streaming model introduced by Martinian and Sundberg in [3]. These streaming models involve fixing the sizes of message packets and channel packets in advance. Badr et al. [5] introduced a new streaming model with fixed-size message packets and channel packets in which every sliding window of w channel packets can include (a) a burst of length b or (b) up to a arbitrary losses. The authors also showed an upper bound on the rate under this sliding window model of loss. Several later works [6], [7], [8], [9], [10], [11] designed streaming codes that matched this upper bound on the rate. Two previous works [12], [13] studied the setting of multiplexing two streams of message packets with different delay constraints. A few works [14], [15], [22] have considered streaming codes where there are two different decoding delay constraints based on two different types of packet loss. In [16], the authors studied the setting where all or some symbols of message packets are recovered for short or long bursts, respectively. Badr et al. investigated [20] streaming codes that recover only some message packets within the delay constraint, depending on the loss patterns. Another work [17] studied streaming codes in terms of the average decoding delay rather than the maximum delay. In [25], the authors evaluate the trade-off between memory, decoding delay, and decoding probability for random linear streaming codes with i.i.d. losses. Several works [5], [18], [19] studied models of streaming codes where multiple channel packets are sent during each time slot. In [23], the authors presented streaming codes to recover multiple bursts within $(\tau+1)$ channel packets. Another work [24] considered unequal error protection for a streaming model with high and low priority messages of two different fixed sizes when the sequence of the priorities of the messages is periodic. Several recent works [21], [29], [30] have applied streaming codes to multi-node relay networks. Future work could compare online and offline constructions for these variants of the streaming model after incorporating message packets of varying sizes.

III. ONLINE CODE CONSTRUCTIONS WITH OPTIMAL RATE

In this section, we present the first rate-optimal online streaming codes, as well as show that they match the offline-optimal-rate, for two broad parameter regimes: Regime 1: ($\tau_L = 0$ and any b and τ) and Regime 2: ($\tau_L = (\tau - b)$ and $b|\tau$).

To begin, we consider Regime 1 (i.e., $\tau_L = 0$ and any b and τ). In this regime, the lossless-delay constraint, $\tau_L = 0$, eliminates the choice of distributing symbols corresponding to a message packet over multiple channel packets. We introduce a systematic construction that sends each message packet within the corresponding channel packet. The construction employs an online greedy paradigm for sending parity symbols. The approach involves (a) identifying during time slot i how many parity symbols will be sent during time slot $(i + \tau)$ (i.e., in advance τ time slots), and (b) defining the parity symbols only during time slot $(i + \tau)$ based on the sizes of $S[i+1], \dots, S[i+\tau-1]$. To show that the construction is rate-optimal, we demonstrate via induction that the cumulative number of symbols sent by each time slot $i \in [t]$ is no more than that which is sent under an arbitrary offline construction.

We next present the rate-optimal online coding scheme for any (τ, b) under Regime 1. The scheme builds on top of the Generalized Maximally Short Codes presented in [5] in such a way so as to mitigate the adverse effects of the variability of the message size sequence. We call the proposed scheme the (τ, b) -Variable-sized Generalized MS Code. The construction is suitable for any field of size at least $2\tau m$. We first provide a high-level description, then present a toy example, and finally present the details of the code construction.

Encoding (high level description). During time slot i , each message packet $S[i]$ is partitioned into two pieces: $S[i] = (U[i], V[i])$. The channel packet $X[i] = (S[i], P[i])$ is then sent, where $P[i]$ comprises parity symbols. The parity symbols are defined as $P[i] = (U[i - \tau] + P'[i])$ where $P'[i]$ consists of carefully designed linear combinations of the symbols of $(V[i - \tau], \dots, V[i - 1])$. The linear equations are defined so that that for all $i \in [t - \tau - b + 1, t]$, $P'[i + b], \dots, P'[i + \tau - 1], V[0], \dots, V[i - 1]$ are sufficient to decode $V[i], \dots, V[i + b - 1]$, as will be fully explained in the detailed description.³

We set $V[i]$ to contain as many symbols of $S[i]$ as possible while meeting the following requirement. For any $j \in \{i - b + 1, \dots, i\}$ and burst loss of $X[j], \dots, X[j + b - 1]$, the sum of the sizes of $V[j], \dots, V[i]$ is at most the number of parity symbols in $X[j + b], \dots, X[j + \tau - 1]$ (i.e., the sum of the sizes of $P[j + b], \dots, P[j + \tau - 1]$). The remaining symbols of $S[i]$ are allocated to $U[i]$. The size of $P[i]$ is set to equal that of $U[i - \tau]$.

Decoding (high level description). A burst loss of $X[i], \dots, X[i + b - 1]$ is recovered in two steps. First, for $j \in \{i + b, \dots, i + \tau - 1\}$, $U[j - \tau]$ is subtracted from $P[j]$ to obtain $P'[j]$. Then $P'[i + b], \dots, P'[i + \tau - 1]$ are used to recover $V[i], \dots, V[i + b - 1]$ during the time slot $(i + \tau - 1)$. Recovery is possible because (a) $P'[i + b], \dots, P'[i + \tau - 1]$ contain at least as many symbols as $V[i], \dots, V[i + b - 1]$ by

³For $i < \tau$, $P[i]$ is empty.

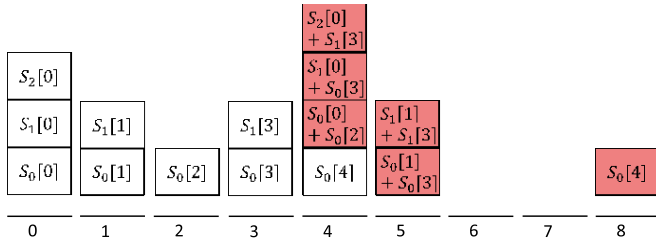


Fig. 2. A toy example of the $(\tau = 4, b = 2)$ -Variable-sized Generalized MS Code. Each message packet, $S[i] = (U[i], V[i])$, is transmitted in the corresponding channel packet, $X[i]$, along with parity symbols, $P[i]$, (when applicable). White boxes with purple dots represent symbols of $U[i]$, white boxes with an orange grid represent symbols of $V[i]$, and solid red boxes represent symbols of $P[i]$. The numbers under the lines at the bottom indicate the time slots.

definition, and (b) the linear equations used to define $P'[i + b], \dots, P'[i + \tau - 1]$ are chosen to be linearly independent. Second, during time slot $j \in \{i + \tau, \dots, i + \tau + b - 1\}$, $V[j - \tau], \dots, V[j - 1]$ are used to compute $P'[j]$. Subtracting $P'[j]$ from $P[j]$ yields $U[j - \tau]$.

Code construction (toy example). We now present a toy example of $(\tau = 4, b = 2)$ -Variable-sized Generalized MS Code for message size sequence $k_0 = 3, k_1 = 2, k_2 = 1, k_3 = 2, k_4 = 1$, and $k_5 = \dots = k_8 = 0$, shown in Figure 2. For $i \in [4]$, $S[i]$ is sent in $X[i]$. This satisfies the lossless-delay constraint. For $i \in \{0, 1, 4\}$, $U[i]$ is defined to equal $S[i]$, and $V[i]$ is defined to be empty (i.e., of size 0). For $i \in \{2, 3\}$, $V[i]$ is set as $S[i]$, and $U[i]$ is defined to be empty. Let $P'[4] = (S_0[2], S_0[3], S_1[3])$ and $P'[5] = (S_0[3], S_1[3])$. Next, $P[4] = (S[0] + P'[4])$ is transmitted in $X[4]$, and $P[5] = (S[1] + P'[5])$ is sent in $X[5]$. Finally, $P_0[8] = S_0[4]$ is transmitted in $X[8]$. The lossless-delay constraint is met, since each message packet is sent within the corresponding channel packet. If any symbols of $V[2]$ and or $V[3]$ are lost, they are recovered using $P[4]$ and $P[5]$ respectively. Any lost symbols of $U[0], U[1]$, and $U[4]$ are each decoded with delay exactly 4 using $P[4], P[5]$, and $P[8]$ respectively (and subtracting $P'[4]$ and $P'[5]$ from $P[4]$ and $P[5]$ respectively). Therefore, the worst-case-delay constraint is satisfied.

Before presenting the detailed description, we introduce some notation. For $Z \in \{S, X, U, V, P, P'\}$ and any $i \leq j \in [t]$, $Z[i]$ is a vector of length $z[i]$, and $Z[i : j] = (Z[i], \dots, Z[j])$.

Code construction (detailed description). During each time slot i , the channel packet $X[i] = (S[i], P[i])$ is sent. The scheme is formally described in three parts: initialization, partitioning $S[i]$ into $(U[i], V[i])$, and defining $P[i]$.

Initialization: For $i \in [b - 1]$, we set $U[i] = S[i]$ and $v[i] = 0$. For $i \in [\tau - 1]$ we set $p[i] = 0$. Let A be a $\tau m \times \tau m$ Cauchy matrix, where m was defined in Section II-B as an upper bound on the sizes of message packets.

Partitioning $S[i]$: For any $i \geq b$, we partition $S[i]$ into $S[i] = (U[i], V[i])$ as follows.⁴ We define an auxiliary variable z_i encapsulating the minimum number of parity symbols available for recovering $S[i]$ when $X[i]$ is

dropped in a burst:

$$z_i = \min_{j \in \{i - b + 1, \dots, i\}} \min_{l = j + b}^{i + \tau - 1} p[l] - \sum_{l = j}^{i - 1} k_l. \quad (4)$$

The first $\min(k_i, z_i)$ symbols of $S[i]$ are set to $V[i]$:

$$V[i] = S_0[i], \dots, S_{\min(k_i, z_i) - 1}[i] \quad (5)$$

The remaining symbols of $S[i]$ are set to $U[i]$:

$$U[i] = S_{\min(k_i, z_i)}[i], \dots, S_{k_i - 1}[i]. \quad (6)$$

Finally,

$$p[i + \tau] = u[i] = k_i - \min(k_i, z_i) = k_i - v[i] \quad (7)$$

parity symbols are assigned to be sent in the channel packet $X[i + \tau]$, although the actual symbols of $P[i + \tau]$ have not yet been identified. The size of $p[i + \tau]$ is never greater than k_i (that is, the maximum possible size of $u[i]$), therefore $p[i + \tau]$ is at most m .

Defining $P[i]$: During time slot $(i \geq \tau)$, we set

$$P[i] = (U[i - \tau] + P'[i]) \quad (8)$$

where the symbols of $P'[i]$ are linear combinations of the symbols of $V[i - \tau], \dots, V[i - 1]$.⁵ The linear combinations are chosen from a Cauchy matrix, as described below. Let $V^{\oplus}[j]$ be the length m vector obtained by appending $(m - v[j])$ 0's to $V[j]$ for $j \in \{i - \tau, \dots, i - 1\}$. We define a vector of length τm , $E[i]$, by placing $V^{\oplus}[j]$, for $j \in \{i - \tau, \dots, i - 1\}$, into m consecutive positions of $E[i]$ starting with position $(j \bmod \tau)m$, as is detailed in Figure 3.⁶ We use the Cauchy matrix A to define

$$P'[i] = E[i]A_{\{(i \bmod \tau)m, \dots, (i \bmod \tau)m + p[i] - 1\}}. \quad (9)$$

The field size requirement is dictated by the Cauchy matrix and is at most $2\tau m$.

In Theorem 1 below, we verify that the Variable-sized Generalized MS Code meets the requirements of the model.

Theorem 1: For any parameters (τ, b) and message size sequence k_0, \dots, k_t , the (τ, b) -Variable-sized Generalized MS Code satisfies the lossless-delay and worst-case-delay constraints over any $C(b, \tau)$ channel.

Proof: The lossless-delay constraint is satisfied for $i \in [t]$ by sending $X[i] = (S[i], P[i])$.

We prove that the worst-case-delay constraint is satisfied by showing for any $i \in [t - \tau]$ that each of $S[i], \dots, S[i + b - 1]$ are recovered within delay τ when $X[i], \dots, X[i + b - 1]$ are lost.⁷ First, we show that $V[i], \dots, V[i + b - 1]$ are recovered by time slot $(i + \tau - 1)$. Second, we show that $U[i], \dots, U[i + b - 1]$ are recovered by time slots $(i + \tau), \dots, (i + \tau + b - 1)$, respectively.

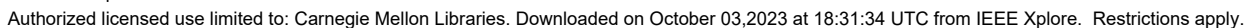
First, for $j \in \{i + b, \dots, i + \tau - 1\}$ subtracting $U[j - \tau]$ from $P[j]$ yields $P'[j]$ (by Equation 8). Combining Equations 5, 6, 7, and 8 shows that the total number of symbols in

⁵Recall that $p[i]$ was defined during initialization for $i < \tau$.

⁶For each $l \in \{i, \dots, i + \tau - 1\}$, $V^{\oplus}[l]$ appears in the same positions of $E[l]$ as in $E[i]$.

⁷Each message packet $S[i]$ for $i > (t - \tau)$ is of size 0 and is known by the receiver due to the termination of the message size sequence.

⁴Recall that partitioning was defined for $i < b$ in initialization.



the greedy paradigm of the (τ, b) -Variable-sized Generalized MS Code sending for each message packet $S[i]$: (a) the minimal number of parity symbols needed to recover $S[i]$ given any burst assuming that no future message packets needs to be recovered, and (b) deferring the transmission of the parity symbols until the decoding deadline for $S[i]$ (i.e., $X[i + \tau]$). The methodology for designing a streaming code using a greedy paradigm and inductively proving that it is rate-optimal form a suitable template for designing new online coding schemes in other regimes, as discussed in Section V.

Theorem 2: For any parameters $(\tau, b, \tau_L = 0)$, the (τ, b) -Variable-sized Generalized MS Code is rate-optimal for transmission over a $C(b, \tau)$ channel.

Proof Sketch: We present the full proof in Appendix A.

For an arbitrary message size sequence k_0, k_1, \dots, k_t , consider any optimal offline construction O . We prove by induction on time slot $i = 0, 1, 2, \dots, t$ that the cumulative number of symbols sent by O is at least as many as that of the (τ, b) -Variable-sized Generalized MS Code.

In the base case, for each $i \in [\tau - 1]$, the channel packet $X[i]$ under O must contain at least k_i symbols to meet the lossless-delay constraint for message packet $S[i]$. Under the (τ, b) -Variable-sized Generalized MS Code, $x[i] = k_i$.

The inductive step for $i \in \{\tau, \dots, t\}$ has two cases.

First, when no parity symbols are sent in $X[i]$ (that is, $X[i] = S[i]$) under the (τ, b) -Variable-sized Generalized MS Code, at least $s[i] = k_i$ symbols are sent in $X[i]$ under O to meet the lossless-delay constraint.

Second, suppose that $X[i] = (S[i], P[i])$ is sent under the (τ, b) -Variable-sized Generalized MS Code where $p[i] > 0$. Applying Lemma 1 shows that there is a burst loss starting at time slot $j \in \{i - \tau - b + 1, \dots, i - \tau\}$ where the number of parity symbols received under the (τ, b) -Variable-sized Generalized MS Code in $X[j + b], \dots, X[i]$ is the smallest for which it is possible to decode message packet $S[j], \dots, S[i - \tau]$. We combine this fact with the lossless-delay constraint for $S[j + b], \dots, S[i]$. We then show that at least as many symbols are sent under O between time slots $(j + b)$ and i as are, respectively, sent under the (τ, b) -Variable-sized Generalized MS Code. Applying the inductive hypothesis for time slot $(j + b - 1)$ concludes the proof. ■

We note that for any values of τ and b , the (τ, b) -Variable-sized Generalized MS Code's rate (i.e., the optimal rate) is highly dependent on the precise sequence of the sizes of the messages. Hence, a closed-form expression is not viable.

Finally, we discuss Regime 2 (i.e., $\tau_L = (\tau - b)$ and $b|\tau$). Under Regime 2, for any parameters (τ, b) , we show that a simple online coding scheme applied to each message packet has rate $\frac{\tau}{\tau + b}$.⁹ Recall that $\frac{\tau}{\tau + b}$ is an upper bound on rate for the streaming model with variable-size message packets [27]. Hence, the simple construction is rate-optimal.

Under this encoding scheme, each message packet $S[i]$ is evenly partitioned into $\frac{\tau}{b}$ components that are transmitted in channel packets $X[i], X[i + b], \dots, X[i + \tau - b]$, respectively. The parity symbols, in the form of the sum of these $\frac{\tau}{b}$ channel packets, are sent as $X[i + \tau] = \sum_{j=0}^{\frac{\tau-b}{b}} X[i + jb]$.¹⁰ Note that in this coding scheme, each transmission occurs exactly b channel packets apart, which is only possible under Regime 2. As such, each burst over $X[i], \dots, X[i + \tau]$ drops precisely one of $X[i], X[i + b], \dots, X[i + \tau - b]$, and $X[i + \tau]$. The remaining channel packets suffice to recover the missing one to meet the worst-case-delay constraint. Finally, we note that sending $S[i]$ over $X[i], \dots, X[i + \tau - b]$ satisfies the lossless-delay constraint, as $(i + \tau_L) = (i + \tau - b)$.

In this section, we presented rate-optimal online streaming codes for Regime 1 and Regime 2. We showed in the proof of Theorem 2 that, for any (τ, b) , the (τ, b) -Variable-sized Generalized MS Code matches the rate of the best offline construction possible for Regime 1. The simple construction for Regime 2 matches the upper bound of the rate of $\frac{\tau}{\tau + b}$. Both of these constructions match the best possible rates of the offline setting, establishing that the online-optimal-rate equals the offline-optimal-rate in both parameter regimes. The construction for Regime 1 can be used for any value of τ_L , although it is not necessarily rate-optimal for $\tau_L > 0$. Next, in Section IV, we show that online codes cannot match the offline-optimal-rate for all other parameter settings.

IV. INFEASIBILITY OF OFFLINE-OPTIMAL-RATE FOR ONLINE SCHEMES

In Section III, we presented online code constructions that matched the offline-optimal-rate under the two broad settings of Regime 1 and Regime 2. A natural question is whether there are any other parameter settings where an online coding scheme can attain the offline-optimal-rate. In this section, we show that the online-optimal-rate is strictly less than the offline-optimal-rate for all other parameter settings.

At a high level, the optimal approach to spreading symbols from a message packet $S[i]$ over channel packets $X[i], \dots, X[i + \tau_L]$ depends on the sizes of future message packets (i.e., k_{i+1}, \dots, k_t). This dependency enables offline coding schemes to have higher rates than online coding schemes in all settings besides Regime 1 and Regime 2, as we will show in Theorem 3.

Theorem 3: For any parameters (τ, b, τ_L) outside of Regime 1 and Regime 2, the online-optimal-rate is strictly less than offline-optimal-rate.

Proof Sketch: The proof consists of three mutually exclusive cases shown via illustrative examples in Sections IV-A, IV-B, and IV-C and in detail in Appendix B, C, and D. In each case, we present two distinct message size sequences of length $(t + 1)$, which match for the first several time slots. We show a lower bound on the offline-optimal-rate for the two message

⁹The construction applies when $(\tau/b) \mid k_i$ for any $i \in [t]$. This condition can be satisfied by padding each message packet with up to $(\tau/b - 1)$ symbols. For real-world live-streaming applications, the amount of padding is typically negligible (e.g., three orders of magnitude smaller than the average size of a message packet).

¹⁰A generalized version of this construction appeared in [27] after the conference version [1] of our work included the construction presented here. A recent work employed a similar interleaving approach in designing a low complexity streaming code with linear field size in the setting of fixed-size message packets [28].

size sequences by presenting an offline coding scheme with rates $R_t^{(1)}$ and $R_t^{(2)}$ on the first and second message size sequences, respectively. To attain a rate of at least $R_t^{(1)}$ on the first message size sequence requires sending symbols in a manner that leads to a lower rate than $R_t^{(2)}$ on the second. ■

Remark 1: Although Theorem 3 is proven for two specific message size sequences, a similar proof holds if the sizes of the message packets were only approximately the sizes corresponding to the message size sequences. As such, the result establishes a broad class of message size sequences for which there is a gap between the online-optimal-rate and the offline-optimal-rate.

A. Case $\tau_L \geq b$ and $\tau_L = (\tau - b)$

This section presents the proof for parameters $(b, \tau_L, \tau) = (3, 4, 7)$; the general case, which builds closely on this example, is proven in Appendix B.

Consider the following two message size sequences :

- 1) $k_0^{(1)} = 2$ and $k_j^{(1)} = 0$ for $j > 0$.
- 2) $k_0^{(2)} = 2, k_1^{(2)} = 2, k_2^{(2)} = 10$, and $k_j^{(2)} = 0$ for $j > 2$.

An offline construction for the two message size sequences is shown in Figures 4 and 5 respectively, over F_q for any prime $q \geq 83$.

For message size sequence 1, the construction sends $X[0] = S_0[0]$, $X[3] = S_1[0]$, and $X[6] = (S_0[0] + S_1[0])$, as shown in Figure 4. The lossless-delay constraint is trivially satisfied. The worst-case-delay constraint is met, as at most one of $X[0]$, $X[3]$, and $X[6]$ is lost.

For message size sequence 2, the construction sends $X[0] = S[0], X[1] = S[1]$, for $i \in \{2, \dots, 6\}$ sends $X[i] = S_{2(i-2)}[2], S_{2(i-2)+1}[2]$, $X[7] = S[0] + \sum_{i=3}^6 X[i]$, $X[8] = S[1] + \sum_{i=4}^6 2^{i-2} X[i]$, and $X[9] = \sum_{i=2}^6 3^{i-2} X[i]$, as shown in Figure 5. The lossless-delay is clearly satisfied. The worst-case-delay constraint is met, as will be shown next through a comprehensive case analysis. For any $l \in \{0, 1\}$ suppose that $X[l]$ is lost, then one obtains $S[l] = X[7+l] - \sum_{j=3+l}^6 (l+1)^{j-2} X[j]$ within 7 time slots. When $X[2]$ is lost, $S[0]$ and $S[1]$ are decoded. Then one can decode

$$\begin{aligned} (S_4[2], S_5[2]) &= 2^{-2} X[8] - S[1] - 2^3 X[5] - 2^4 X[6] \\ (S_2[2], S_3[2]) &= (X[7] - S[0] - X[4] - X[5] - X[6]) \\ &\quad - \sum_{j=3}^6 X^6 3^{j-2} X[j] \\ (S_0[2], S_1[2]) &= \sum_{j=3}^6 X[9] - \sum_{j=3}^6 3^{j-2} X[j] \end{aligned}$$

When a burst starts with $X[3]$, $S[0], S[1]$, and $S_0[2]$ are decoded, and $(S_8[2], S_9[2])$ is received. Combining $S[0], S[1]$, and $X[2]$, with $X[6 : 9]$ yields $\sum_{i=3}^6 X[i]$, $\sum_{i=4}^6 2^{i-2} X[i]$, and $\sum_{i=2}^6 3^{i-2} X[i]$. These three equations are linearly independent and yield $X[3 : 5]$. Thus, $S[2]$ is decoded by time slot 9. When a burst starts with $X[4]$, $S[0], S[1], S_0[2], S_1[2], S_2[2]$, and $S_3[2]$ are received and combined with $X[7], X[8]$, and $X[9]$ to determine $\sum_{j=4}^6 X[j]$, $\sum_{j=4}^6 2^{j-2} X[j]$, and $\sum_{j=4}^6 3^{j-2} X[j]$. These three equations are linearly independent and yield $X[4 : 6]$, which consist of $S_4[2], \dots, S_9[2]$. When a burst

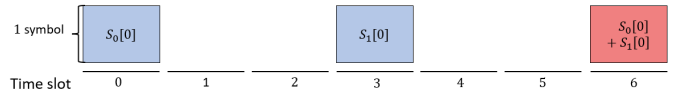


Fig. 4. Offline construction for message size sequence 1 for parameters $(b, \tau_L, \tau) = (3, 4, 7)$.

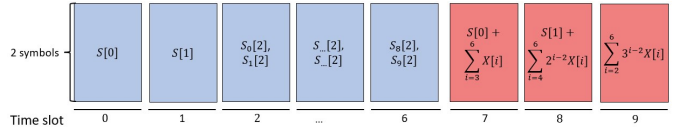


Fig. 5. Offline construction for message size sequence 2 for parameters $(b, \tau_L, \tau) = (3, 4, 7)$.

starts with $X[5]$, $S[1], S_0[2], \dots, S_5[2]$ are received and combined with $X[8]$ and $X[9]$ to determine $\sum_{j=5}^6 2^{j-2} X[j]$, and $\sum_{j=5}^6 3^{j-2} X[j]$. These two equations are linearly independent and yield $X[5]$ and $X[6]$, which include $S_6[2], \dots, S_9[2]$. When a burst starts with $X[6]$, $S_0[2], \dots, S_7[2]$ are received, leading to $(S_8[2], S_9[2]) = 3^{-4} \sum_{j=2}^5 3^{j-2} X[j]$. When $X[0 : 6]$ are received, the message packets are received.

The rate of the offline construction for message size sequence 1 is $2/3$, while its rate for message size sequence 2 is 0.7 . An online construction must send at most 1 symbol in $X[0]$ to have a rate of $2/3$ on message size sequence 1 because $X[0]$ can be lost. We next show that any such scheme cannot attain the rate of 0.7 on message size sequence 2. If message size sequence 2 occurs, the online construction must send at least 13 symbols over $X[1 : 6]$ due to the lossless-delay constraint. At least one of $X[1 : 3]$ and $X[4 : 6]$ must contain at least 7 symbols and may be lost. At least 14 symbols must be received. So the rate is at most $14/21$ (i.e., less than 0.7). Therefore, any online construction with a rate of $2/3$ on message size sequence 1 cannot attain the rate of 0.7 on message size sequence 2, unlike the proposed offline construction.

B. Case $\tau_L < b$ and $\tau_L = (\tau - b)$

This section presents the proof for parameters $(b, \tau_L, \tau) = (2, 1, 3)$; the general case, which builds closely on this example, is proven in Appendix C.

Consider the following two message size sequences:

- 1) $k_0^{(1)} = 2, k_1^{(1)} = 2$, and $k_j^{(1)} = 0$ for $j > 1$.
- 2) $k_0^{(1)} = 2, k_1^{(1)} = 2, k_2^{(1)} = 2$, and $k_j^{(1)} = 0$ for $j > 2$.

An offline construction for the two message size sequences is shown in Figures 6 and 7 respectively over any finite field, F_q .

For message size sequence 1, the construction sends $X[0] = S[0]$, $X[1] = S_0[1]$, $X[2] = S_1[1]$, $X[3] = (S[0] + (0, S_1[1]))$, and $X[4] = (S_0[1] + S_1[1])$, as shown in Figure 6. The lossless-delay constraint is trivially satisfied. The worst-case-delay constraint is met for $S[0]$ because either $S[0]$ is received, or $S_1[1]$ and $X[3]$ are received, yielding $S[0]$. When $X[1]$ is lost, $(0, S_1[1]) = (X[3] - S[0])$ is obtained, leading to $S_0[1] = (X[4] - S_1[1])$. When $X[2]$ is lost, $S_0[1]$ is decoded, leading to $S_1[1] = (X[4] - S_0[1])$. As such, the worst-case-delay is satisfied for $S[1]$.

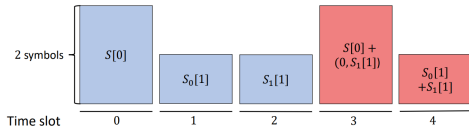


Fig. 6. Offline construction for message size sequence 1 for parameters $(b, \tau_L, \tau) = (2, 1, 3)$.



Fig. 7. Offline construction for message size sequence 2 for parameters $(b, \tau_L, \tau) = (2, 1, 3)$.

For message size sequence 2, the construction sends $X[0] = S[0]$, $X[1] = S[1]$, $X[2] = S[2]$, $X[3] = (S[0] + S[2])$, and $X[4] = (S[1] + S[2])$, as shown in Figure 7. The lossless-delay is clearly satisfied. The worst-case-delay constraint is met for $S[0]$ as either $X[0] = S[0]$ is received, or $S[0] = (X[3] - X[2])$ is obtained. The worst-case-delay constraint is satisfied for $S[1]$ since either $X[1]$ is received, or $S[0]$ is decoded, leading to $S[2] = (X[3] - S[0])$, and $S[1] = (X[4] - S[2])$. The worst-case-delay constraint is satisfied for $S[2]$ because either $X[2]$ is received, or $S[1]$ is decoded, yielding $S[2] = (X[4] - S[1])$.

The offline construction's rate for message size sequence 1 is $4/7$, while its rate for message size sequence 2 is 0.6 . An online construction with a rate of $4/7$ on message size sequence 1 must send at most 3 symbols in $X[0 : 1]$, since at least 4 symbols are sent in $X[2 : 4]$ in case $X[0 : 1]$ is lost. Also, the construction sends at least 2 symbols over $X[0 : 1]$ to recover $S[0]$ under lossless transmission. Next, we show that any such scheme cannot attain the rate of 0.6 on message size sequence 2 due to sending fewer than 4 symbols over $X[0 : 2]$. Thus, any online construction with a rate of $4/7$ on message size sequence 1 cannot attain the rate of 0.6 on message size sequence 2, unlike the proposed offline construction.

First, suppose that exactly 2 symbols are sent in $X[0 : 1]$. Then $X[2 : 3]$ suffices to recover $S[0]$. Recall that the 2 symbols in $X[0 : 1]$ only contain information about $S[0]$, as they suffice to recover $S[0]$ under a lossless transmission. Thus, $X[0 : 1]$ are recovered as a function of $S[0]$, leaving the transmission lossless, so $S[1 : 2]$ are recovered. Thus, $X[2 : 3]$ contains at least 6 symbols. At least 6 symbols are sent outside of $X[2 : 3]$ in case $X[2 : 3]$ is lost, so the rate is at most $6/12$.

Second, due to the upper bound on the rate of $\frac{\tau}{\tau+b} = \frac{3}{5}$ and worst-case-delay, at least $10 = 6 \cdot \frac{5}{3}$ symbols must be sent by time slot 5. Suppose exactly 3 symbols are sent in $X[0 : 1]$. Consider the 5 periodic erasure channels, C_0, \dots, C_4 , where for $i \in [4]$, C_i drops packets $X[j : j+1]$ for all $j \equiv i \pmod{5}$. Each packet is dropped by 2 of these channels, so the channels drop at least $\frac{2}{5} \cdot 10 \geq 4$ symbols on average. At least 6 symbols must be received to ensure recovery. If any channel dropped 5 or more symbols, the rate would be at most $6/11$. Thus, each channel must drop exactly 4 symbols to attain a rate of 0.6 . Therefore, C_0 drops exactly 4 symbols—3 over

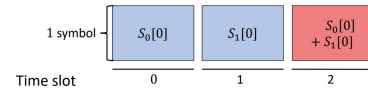


Fig. 8. Offline construction for message size sequence 1 for parameters $(b, \tau_L, \tau) = (1, 1, 3)$.

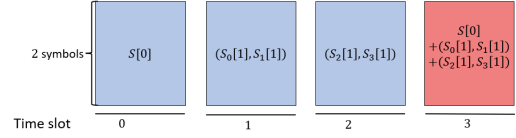


Fig. 9. Offline construction for message size sequence 2 for parameters $(b, \tau_L, \tau) = (1, 1, 3)$.

$X[0 : 1]$ and 1 in $X[5]$. Each of C_4, C_3 , and C_2 must drop 4 symbols (i.e., $n_4 + n_5 = 4$, $n_3 + n_4 = 4$, $n_2 + n_3 = 4$). Hence, $X[4]$ contains 3 symbols, $X[3]$ contains 1 symbol, and $X[2]$ contains 3 symbols. In total, $(3 + 3 + 1 + 3 + 1) = 11$ symbols are sent over $X[0 : 1]$, $X[2]$, $X[3]$, $X[4]$, and $X[5]$, leading to a rate of $6/11$, which is less than 0.6 .

Therefore, any online construction that matches the rate of $4/7$ on message size sequence 1 cannot attain the rate of 0.6 on message size sequence 2, unlike the offline construction.

C. Case $\tau_L < (\tau - b)$

This section presents the proof for parameter $(b, \tau_L, \tau) = (1, 1, 3)$; the general case, which builds closely on this example, is proven in Appendix D.

Consider the following two message size sequences :

- 1) $k_0^{(1)} = 2$ and $k_j^{(1)} = 0$ for $j > 0$.
- 2) $k_0^{(1)} = 2$, $k_1^{(1)} = 4$, and $k_j^{(1)} = 0$ for $j > 1$.

An offline construction for the two message size sequences is shown in Figures 8 and 9 respectively over any finite field, F_q .

For message size sequence 1, the construction sends $X[0] = S_0[0]$, $X[1] = S_1[0]$, and $X[2] = (S_0[0] + S_1[0])$, as is shown in Figure 8. The lossless-delay constraint is trivially satisfied. The worst-case-delay constraint is met because at most one of $X[0]$, $X[1]$, or $X[2]$ is lost and $X[2] = (X[0] + X[1])$.

For message size sequence 2, the construction sends $X[0] = S[0]$, $X[1] = (S_0[1], S_1[1])$, $X[2] = (S_2[1], S_3[1])$, and $X[3] = (S[0] + (S_0[1], S_1[1]) + (S_2[1], S_3[1]))$, as shown in Figure 9. The lossless-delay is clearly satisfied. The worst-case-delay constraint is met, since at most one of $X[0]$, $X[1]$, $X[2]$, or $X[3] = \sum_{i=0}^2 X[i]$ is lost.

The offline construction's rate for message size sequence 1 is $2/3$, while its rate for message size sequence 2 is 0.75 . For an online construction to attain a rate of $2/3$ on message size sequence 1, it must send exactly 1 symbol in each of $X[0]$ and $X[1]$ due to (a) the lossless-delay constraint and (b) ensuring at most 1 symbol is lost—a necessity to attain the rate of $2/3$. Next, we show that any such scheme cannot attain the rate of 0.75 on message size sequence 2. If message size sequence 2 occurs, at least 6 symbols are sent over $X[0 : 2]$ due to the lossless-delay constraint. The average number of symbols per packet is at least 2. If $X[0]$ contains one symbol, at least one of $X[1]$ or $X[2]$ contains at least 3 symbols. At least 6 symbols

must be received to satisfy the worst-case-delay constraint. Since at least 3 symbols may be lost, at least 9 symbols must be sent in total. As such, the rate is at most $2/3$, which is less than 0.75. Therefore, any online construction that matches the rate of $2/3$ on message size sequence 1 cannot attain the rate of 0.75 on message size sequence 2, unlike the offline construction.

V. CONCLUSION

Real-time streaming applications, such as videoconferencing, transmit a sequence of messages of varying sizes. These applications operate in an online setting without access to future message sizes. However, previously studied upper bounds on the rate apply to an offline setting with advance access to the sizes of all messages, leaving the best possible rate of the online setting an open question. We introduce the first rate-optimal online coding schemes for two broad parameter regimes (that is, Regime 1 and Regime 2) which are optimal even for the offline setting. To do so, we propose a framework for designing online constructions using a greedy paradigm for sending parity symbols and inductively analyzing the rate that is suitable for future works for message packets of varying sizes. We also show for all other parameter regimes that the best way to spread the symbols of messages over multiple transmissions depends on the sizes of future messages. Consequently, no online coding scheme can match the optimal rate of offline coding schemes.

The gap between the online-optimal-rate and offline-optimal-rate prompts three directions of further study for the parameter settings outside of Regime 1 and Regime 2. First, how can one design rate-optimal offline code constructions? Second, what does it mean to be rate-optimal in the online setting, given that the rate depends on the specific sequence of sizes of future messages? Third, can one use the proposed methodology to design and analyze online constructions to design rate-optimal or approximately rate-optimal online streaming codes? These questions have been partially answered for the smallest lossless-delay where spreading message symbols can alleviate the variability of the sizes of message packets (i.e., $\tau_L = 1$) in [31]; the questions remain open for large values of τ_L .

APPENDIX

A. Proof of Theorem 2

In this section, we will prove Theorem 2. At a high level, the proof is inductive and shows that the cumulative number of symbols sent by each time slot under the (τ, b) -Variable-sized Generalized MS Code is the minimum possible. For time slots where no parity symbols are sent, it follows immediately by the lossless-delay constraint. Otherwise, there is some burst for which every parity symbol in the received channel packets is needed to recover the burst within the worst-case-delay.

We begin by introducing the preliminary notation for the proof. We then include a few auxiliary Lemmas used throughout the proof. Finally, we present the full proof itself.

Let t be an arbitrary natural number, and consider any length $(t+1)$ message size sequence k_0, \dots, k_t . Let O be an arbitrary

offline code construction that satisfies the lossless-delay and worst-case-delay constraints over a $C(b, \tau)$ channel for the message size sequence. Let the channel packet transmitted during time slot $j \in [t]$ under construction O and under the (τ, b) -Variable-sized Generalized MS Code be labeled as $X_O[j]$ and $X_V[j]$, respectively. Let the cumulative number of symbols transmitted through time slot j under construction O and under the (τ, b) -Variable-sized Generalized MS Code be denoted $n_{O,j}^+ = \sum_{i=0}^j x_O[i]$ and $n_{V,j}^+ = \sum_{i=0}^j x_V[i]$, respectively. Recall from Section II that each message packet comprises symbols drawn independently and uniformly at random from the finite field F_q . Let S be a random variable representing a uniformly random element of F_q .

Next, we show that the lossless-delay constraint necessitates transmitting at least as many symbols as the size of the message packet for each time slot.

Lemma 2: Consider any parameters $(\tau, b, \tau_L = 0)$, an arbitrary message size sequence k_0, k_1, \dots, k_t , and any code construction which satisfies the lossless-delay and worst-case-delay constraints over a $C(b, \tau)$ channel. For any $j \in [t]$, $n_j \geq k_j$.

Proof: Follows directly from (a) the independence of message packets, and (b) the lossless-delay constraint for $\tau_L = 0$. ■

Next, we establish that whenever a burst of length b occurs, all message packets from time slots before the burst must be decoded before the burst to satisfy both the lossless-delay and worst-case-delay constraints.

Lemma 3: Consider any parameters $(\tau, b, \tau_L = 0)$, an arbitrary message size sequence k_0, k_1, \dots, k_t , $j \in [t]$, and any code construction which satisfies the lossless-delay and worst-case-delay constraints over a $C(b, \tau)$ channel. When $X[j], \dots, X[j+b-1]$ are lost in a burst, $S[0:j-1]$ are decoded by time slot $(j-1)$.

Proof: By the worst-case-delay constraint, $S[0:j-\tau-1]$ are all decoded by time slot $(j-1)$. Under the $C(b, \tau)$ channel, when $X[j], \dots, X[j+b-1]$ are lost, $X[j-\tau], \dots, X[j-1]$ are necessarily received.¹¹ By the lossless-delay constraint, $S[0:j-\tau-1]$ and $X[j-\tau:j-1]$ suffice to decode $S[j-\tau:j-1]$. ■

Finally, we prove Theorem 2 below.

Proof of Theorem 2: Let k_0, k_1, \dots, k_t be an arbitrary message size sequence. We will show by induction that the cumulative number of symbols sent through time slot $i \in [t]$ under an arbitrary offline construction, O , is at least as many as that of the (τ, b) -Variable-sized Generalized MS Code (i.e., $n_{O,i}^+ \geq n_{V,i}^+$). Consequently, the (τ, b) -Variable-sized Generalized MS Code matches the offline-optimal-rate.

In the base case, we consider $j \in [\tau-1]$. Applying Lemma 2 determines that $x_O[j] \geq k_j = x_V[j] \in [\tau-1]$.

For the inductive hypothesis, we assume that for some $(i \geq \tau-1)$, for all $l \in [i]$, $n_{O,l}^+ \geq n_{V,l}^+$.

For the inductive step, consider the time slot $(i = i_{\text{ind}} + 1 \geq \tau)$.

By the inductive hypothesis, $n_{O,i-1}^+ \geq n_{V,i-1}^+$. We will show that $n_{O,i}^+ \geq n_{V,i}^+$ using two cases.

¹¹When $j < \tau$, $X[0:j-1]$ are received.

Case $x_v[i] = k_i$:

Applying Lemma 2 determines that $x_o[i] \geq k_i$. Therefore, $(n_{o,i}^+ = n_{o,i-1}^+ + k_i \geq n_{v,i-1}^+ + k_i = n_{v,i}^+)$.

Case $x_v[i] > k_i$: We first provide a high-level intuition of the proof and then the detailed derivation.

High-level summary: Applying Lemma 1 shows that there is a burst starting in time slot $j \in \{i-\tau-b+1, \dots, i-\tau\}$ for which the (τ, b) -Variable-sized Generalized MS Code receives minimum required number of parity symbols to decode message packets $S[j : i-\tau]$ by time slot i . Combining this fact with meeting the lossless-delay constraint for $S[j+b : i]$ shows that the number of symbols sent under O between time slots $(j+b)$ and i is at least as many as that of the (τ, b) -Variable-sized Generalized MS Code.

Detailed derivation: By Lemma 1, there is some $j \in \{i-\tau-b+1, \dots, i-\tau\}$ such that $\sum_{l=j+b}^i p[l] = \sum_{l=j}^{i-\tau} k_l$. Therefore,

$$x_v[i] = \sum_{l=j+b}^i x_v[l] = \sum_{l=j}^{i-\tau} k_l + \sum_{l=j+b}^i k_l. \quad (10)$$

Next, we show that at least as many symbols are sent over $X_o[j+b : i]$ as are sent over $X_v[j+b : i]$. Consider a burst loss of $X[j], \dots, X[j+b-1]$. Applying Lemma 3 shows that $S[0 : j-1]$ are known by the receiver by time slot $(j-1)$. By the worst-case-delay constraint,

$$H(S[j : i-\tau] | X_o[j+b : i], S[0 : j-1]) = 0. \quad (11)$$

We next bound the number of symbols sent over $X_o[j+b : i]$ as

$$H(S[j : i-\tau]) + H(X_o[j+b : i] | S[0 : i-\tau]) = \quad (12)$$

$$H(X_o[j+b : i], S[j : i-\tau] | S[0 : j-1]) = \quad (13)$$

$$H(X_o[j+b : i] | S[0 : j-1]) + \quad (14)$$

$$H(S[j : i-\tau] | S[0 : j-1], X_o[j+b : i]) = \quad (15)$$

$$H(X_o[j+b : i] | S[0 : j-1]),$$

where Equation 13 follows from the chain rule and independence of message packets, Equation 14 follows from the chain rule, and Equation 15 follows from Equation 11.

Combining Equations 13 and 15 with the fact that conditioning reduces entropy yields

$$H(X_o[j+b : i]) \geq H(X_o[j+b : i] | S[0 : j-1]) \geq H(S[j : i-\tau]) + H(X_o[j+b : i] | S[0 : j+b-1]). \quad (16)$$

Next, we evaluate the size of $H(X_o[j+b : i] | S[0 : j+b-1])$ as

$$H(S[j+b : i], X_o[j+b : i] | S[0 : j+b-1]) = \quad (17)$$

$$H(S[j+b : i]) + H(X_o[j+b : i] | S[0 : i]) = \quad (18)$$

$$H(S[j+b : i]) = \quad (19)$$

$$H(X_o[j+b : i] | S[0 : j+b-1]) + \quad (20)$$

$$H(S[j+b : i] | S[0 : j+b-1], X_o[j+b : i]) = \quad (21)$$

$$H(X_o[j+b : i] | S[0 : j+b-1]),$$

where Equation 18 follows from conditioning and independence of message packets, Equation 19 follows from the fact that for $l \in [t]$, $X_o[l]$ is a function of $S[0 : l]$, Equation 20

follows from conditioning, and Equation 21 follows from the lossless-delay constraint.

For any $i \in [t]$,

$$H(S[i]) = H(S)k_i \quad (22)$$

$$H(X[i]) \leq H(S)n_i \quad (23)$$

where S was defined as a random variable drawn uniformly at random from the underlying field, F_q . This follows from the definition of message packets, and the fact that the maximum possible entropy of n_i symbols is $n_i H(S)$. Applying Equation 21 and 19 to Equations 16, 22, and 23 yields

$$\begin{aligned} H(S) \sum_{l=j+b}^i X_l & \geq H(X_o[j+b : i]) \geq \\ & H(S[j : i-\tau]) + H(S[j+b : i]) = \\ & H(S) \sum_{l=j}^{i-\tau} k_l + \sum_{l=j+b}^i k_l. \end{aligned} \quad (24)$$

Combining Equations 24 and 10 determines that

$$H(S) \sum_{l=j+b}^i X_l \geq H(S) \sum_{l=j+b}^i x_v[l]. \quad (25)$$

By definition, $(n_{o,i}^+ = n_{o,j+b-1}^+ + \sum_{l=j+b}^i x_o[l])$ and $(n_{v,i}^+ = n_{v,j+b-1}^+ + \sum_{l=j+b}^i x_v[l])$. Applying the inductive hypothesis to $(j+b-1 < i)$ shows that $(n_{v,j+b-1}^+ \leq n_{o,j+b-1}^+)$. Combining the above equations with Equation 25 determines that $n_{v,i}^+ \leq n_{o,i}^+$. The inductive hypothesis is proven, and the result follows immediately. ■

B. Proof of Theorem 3 case $\tau_L \geq b$ and $\tau_L = (\tau - b)$

Let $(a \equiv \tau \bmod b)$ and $(e \equiv \tau_L \bmod b)$. Theorem 3 does not apply when $\tau = (\tau_L - b)$ and $b | \tau$, necessitating that $(e > 0)$. Let d be an arbitrary multiple of $(a + 1)$.

Consider the following two message size sequences for which the offline construction will be shown below in Figures 10 and 11 respectively:

- 1) $k_0^{(1)} = \dots = k_{e-1}^{(1)} = d$, and $k_e^{(1)} = \dots = k_t^{(1)} = 0$.
- 2) $k_0^{(2)} = \dots = k_{b-2}^{(2)} = d$, $k_{b-1}^{(2)} = d(\tau_L + 1)$, and $k_b^{(2)} = \dots = k_t^{(2)} = 0$.

Before going into the details of the proof, we note that the proof applies for any value of d . When d is sufficiently large, the proof could also be extended to message size sequences where the message packets' sizes may only approximately equal the ones in the message size sequence. More generally, the proof also applies for any message size sequences for which there is a subsequence of (a) τ message packets whose sizes are $\geq d$, then (b) one of the two above message size sequences, then (c) another τ message packets whose sizes are $\geq d$.

We present an offline coding scheme for message size sequences 1 and 2, which has rates

$$R_t^{(1)} = \frac{a+1}{a+2}, \quad R_t^{(2)} = \frac{\tau}{\tau+b} \quad (26)$$

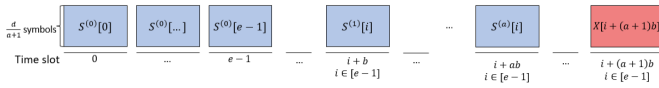


Fig. 10. The offline scheme for message size sequence 1 for case $\tau_L \leq b$ and $\tau_L = (\tau - b)$. Blue channel packets consist of message symbols, and red channel packets consist of parity symbols. The numbers under the lines at the bottom indicate the time slots. The offline scheme sends $\frac{d}{a+1}$ symbols in each of the first e channel packets.

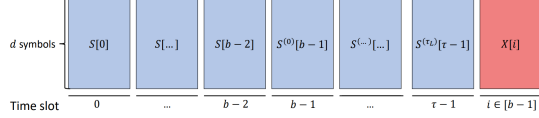


Fig. 11. The offline scheme for message size sequence 2 for case $\tau_L \leq b$ and $\tau_L = (\tau - b)$. Blue channel packets consist of message symbols and red channel packets consist of parity symbols. The numbers under the lines at the bottom indicate the time slots. The offline scheme sends d symbols in each of the first e channel packets.

on the two message size sequences, respectively. We describe and then validate the scheme for each message size sequence.

Offline scheme for message size sequence 1: Each message packet is encoded separately with parameters $(\tau' = \lceil \frac{\tau}{b} \rceil b, b' = b, \tau'_L = \tau' - b)$ as described in Section III, shown in Figure 10, and detailed below.

- For $i \in [e-1]$, $S[i]$ is evenly divided into $(a+1)$ components of size d each: $S^{(0)}[i], \dots, S^{(a)}[i]$. For $j \in [a]$, $X[i + jb] = S^{(j)}[i]$.
- For $i \in [e-1]$, $X[i + (a+1)b] = \sum_{z=0}^a X[i + zb]$.

Decoding: For $i \in [e-1]$, $S[i]$ is sent evenly over $X[i]$, $X[i + b]$, \dots , $X[i + ab]$ where $(i + ab) \leq (i + \tau_L)$ and at most one of $X[i]$, $X[i + b]$, \dots , $X[i + ab]$, or $X[i + (a+1)b] = \sum_{j=0}^a X[i + jb]$ is lost. Each message packet is decoded within delay τ_L when the transmission is lossless and using a linear combination of the relevant $(a+1)$ channel packets within delay τ otherwise.

Offline scheme for message size sequence 2: The first $(b-1)$ message packets are sent with no delay and the symbols of the next message packet are transmitted evenly over $X[b-1], \dots, X[\tau-1]$. The symbols of $X[0 : \tau-1]$ are used to create d blocks of the rate $\frac{\tau}{\tau+b}$ systematic block code from [8]. Each of the d blocks includes b parity symbols that are sent in $X[\tau], \dots, X[\tau+b-1]$ respectively. The block code maps τ input symbols $(s_0, \dots, s_{\tau-1})$ to $(\tau+b)$ codeword symbols $(s_0, \dots, s_{\tau-1}, p_0, \dots, p_{b-1})$. For each $j \in [\tau-1]$ and any burst erasing up to b codeword symbols, the non-erased symbols of $(s_0, \dots, s_{\tau-1}, p_0, \dots, p_{\min(b-1, j)})$ are sufficient to decode s_j . Therefore, each symbol is recovered within τ symbols. We note that although we use the block code from [8], any other block code from [6], [7], [9], [11] also works. The scheme is described in detail below and shown in Figure 11:

- For $j \in [b-2]$, $X[j] = S[j]$.
- $S[b-1]$ is divided evenly into $(\tau_L + 1)$ components of size d : $S^{(0)}[b-1], \dots, S^{(\tau_L)}[b-1]$.
- For $j \in \{b-1, \dots, b-1 + \tau_L\}$, $X[j] = S^{(j-b+1)}[b-1]$.
- For each $z \in [d-1]$, an instance of the block code from [8] is created which maps $(X_z[0], \dots, X_z[\tau-1])$ to $(X_z[0], \dots, X_z[\tau-1], p_0^{(z)}, \dots, p_{b-1}^{(z)})$.
- For $j \in [b-1]$, $X[\tau + j] = (p_j^{(0)}, \dots, p_j^{(d-1)})$.

Decoding: Each message packet is transmitted within the current and next τ_L channel packets and is, thus, decoded when the transmission is lossless. Each symbol $X_z[i]$ for $z \in [d-1]$ and $i \in [\tau-1]$ is decoded within the delay τ or by time slot $(\tau + b - 1)$ using the block code $(X_z[0], \dots, X_z[\tau-1], p_0^{(z)}, \dots, p_{b-1}^{(z)})$. Hence, the worst-case-delay constraint is met.

Proof of the converse result : The offline-optimal-rate is at least $R_t^{(1)}$ and $R_t^{(2)}$ (that is, the rate of the offline scheme from Equation 26) for message size sequences 1 and 2, respectively. Next, we show mutually exclusive conditions for the sum of the sizes of $X[0], \dots, X[e-1]$ to have rates at least $R_t^{(1)}$ and $R_t^{(2)}$ on message size sequences 1 and 2 respectively. All online coding schemes, thus, fail the condition for at least one message size sequence since they are identical until time slot e .

Condition for rate $R_t^{(1)}$ on message size sequence 1 : Consider any coding scheme for message size sequence 1. At least d symbols are sent over $X[b], \dots, X[t]$ since $X[0], \dots, X[b-1]$ could be lost. At most $d \frac{e}{a+1}$ symbols can be sent over $X[0], \dots, X[b-1]$ if the rate is at least $R_t^{(1)}$.

Condition for rate $R_t^{(2)}$ on message size sequence 2 : Consider an arbitrary coding scheme for message size sequence 2. At least $d\tau$ symbols are sent in $X[0], \dots, X[\tau-1]$ to meet the lossless-delay constraint. For each $i \in [a]$, at least $d\tau$ symbols are sent outside of $X[e + ib : e + (i+1)b - 1]$ in case $X[e + ib : e + (i+1)b - 1]$ is lost. Since the rate is $R_t^{(2)}$, at most db symbols are sent in $X[e + ib : e + (i+1)b - 1]$. As such, at least $(d\tau - d(a+1)b = de)$ symbols are sent in $X[0 : e-1]$.

Summary : Any online scheme whose rate is at least $R_t^{(1)}$ on message size sequence 1 sends at most $d \frac{e}{a+1}$ symbols in $X[0 : b-1]$. As such, its rate is lower than $R_t^{(2)}$ on message size sequence 2.

C. Proof of Theorem 3 case $\tau_L < b$ and $\tau_L = (\tau - b)$

Let d be an arbitrary positive even integer. Consider the following two message size sequences for which the offline construction will be shown below in Figures 12 and 13 respectively:

- 1) $k_0^{(1)} = \dots = k_{b-\tau_L}^{(1)} = d$, and $k_{b-\tau_L+1}^{(1)} = \dots = k_t^{(1)} = 0$.
- 2) $k_0^{(2)} = \dots = k_{b-\tau_L}^{(2)} = d$, $k_{b-\tau_L+1}^{(2)} = \dots = k_{b-1}^{(2)} = 0$, $k_b^{(2)} = d$, and $k_{b+1}^{(2)} = \dots = k_t^{(2)} = 0$.

Before presenting the proof in detail, we observe that the proof could also be extended to similar message size sequences where the sizes of each message packet is perturbed by a small amount as long as d is large. More generally, the proof also applies to any message size sequence that contains one of the two above message size sequences proceeded and followed by τ message packets sufficiently small relative to d .

We will present an offline coding scheme for the two message size sequences with rates

$$R_t^{(1)} = \frac{b - \tau_L + 1}{2b - 2\tau_L + 1.5}, \quad R_t^{(2)} = \frac{b - \tau_L + 2}{2b - 2\tau_L + 3} \quad (27)$$

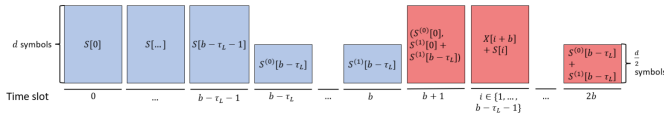


Fig. 12. The offline scheme for message size sequence 1 for case $\tau_L < b$ and $\tau_L = (\tau - b)$. Blue channel packets consist of message symbols, and red channel packets consist of parity symbols. The numbers under the lines at the bottom indicate the time slots. The offline scheme sends $\frac{d}{2}$ symbols in $X[b - \tau_L]$.

on message size sequence 1 and 2, respectively. After presenting the scheme for each message size sequence, we verify that it satisfies the lossless-delay and worst-case-delay constraints.

Offline scheme for message size sequence 1: The first $(b - \tau_L)$ message packets are sent in the corresponding channel packets. The message packet $S[b - \tau_L]$ is divided in half to be evenly transmitted over $X[b - \tau_L]$ and $X[b]$. Each of the next $(b - \tau_L)$ channel packets comprises d parity symbols used to decode (a) the first $(b - \tau_L)$ message packets if the corresponding channel packets are lost and (b) $X[b]$ if $X[b - \tau_L]$ and $X[b]$ are both lost. The summation of $X[b - \tau_L]$ and $X[b]$ is later sent in $X[2b]$ to ensure decoding of $S[b - \tau_L]$ within delay τ . The scheme is detailed below and shown in Figure 12:

- $S[0]$ and $S[b - \tau_L]$ are each evenly divided into two components of $d/2$ symbols each: $S[0] = (S^{(0)}[0], S^{(1)}[0])$ and $S[b - \tau_L] = (S^{(0)}[b - \tau_L], S^{(1)}[b - \tau_L])$.
- For $i \in [b - \tau_L - 1]$, $X[i] = S[i]$.
- $X[b - \tau_L] = S^{(0)}[b - \tau_L]$.
- $X[b] = S^{(1)}[b - \tau_L]$.
- $X[b + 1] = (S^{(0)}[0], S^{(1)}[0] + S^{(1)}[b - \tau_L])$.
- For $i \in \{1, \dots, b - \tau_L - 1\}$, $X[i + b + 1] = (X[i + b] + S[i])$.
- $X[b - \tau_L + \tau] = X[2b] = (S^{(0)}[b - \tau_L] + S^{(1)}[b - \tau_L])$.

Decoding: Each message packet is sent within the current and perhaps next τ_L channel packets and is decoded when the transmission is lossless. We now discuss how message packets are recovered within a delay of τ under lossy conditions. Either $X[0] = S[0]$ is received, or $X[0]$ is lost. In the latter case, both $X[b] = S^{(1)}[b - \tau_L]$ and $X[b + 1] = (S^{(0)}[0], S^{(1)}[0] + S^{(1)}[b - \tau_L])$ are received. Therefore, $S[0]$ is decoded within the delay of τ . Next, for $i \in \{1, \dots, b - \tau_L - 1\}$, either $X[i] = S[i]$ is received, or both $X[i + b]$ and $X[i + b + 1] = (X[i + b] + S[i])$ are received. Thus, $S[i]$ is recovered within delay $(b + 1 \leq \tau)$. Either $X[b - \tau_L] = S^{(0)}[b - \tau_L]$ is received, or $X[2b - \tau_L] = (S^{(0)}[0], S^{(1)}[0] + S^{(1)}[b - \tau_L]) + P^{b - \tau_L - 1} S[i]$ is received. In the latter case, $S[0], \dots, S[b - \tau_L - 1]$ are decoded by time slot $(2b - 1)$ and combined with $X[2b - \tau_L]$ to decode $S^{(1)}[b - \tau_L]$. $S^{(1)}[b - \tau_L]$ is then combined with $X[2b] = (S^{(0)}[b - \tau_L] + S^{(1)}[b - \tau_L])$ to recover $S^{(0)}[b - \tau_L]$ within a delay of τ . Therefore, $S^{(0)}[b - \tau_L]$ is decoded within delay τ . Either $X[b] = S^{(1)}[b - \tau_L]$ is received, or $X[2b] = (S^{(0)}[b - \tau_L] + S^{(1)}[b - \tau_L])$ is received. Recall that $S^{(0)}[b - \tau_L]$ is decoded by time slot $2b$. Thus, $S^{(1)}[b - \tau_L]$ is recovered within delay τ .

Offline scheme for message size sequence 2: Each message packet $S[i]$ is transmitted in the corresponding channel packet $X[i]$. The next τ_L channel packets each comprise d parity symbols. These $d\tau_L$ symbols are used to decode (a) the first

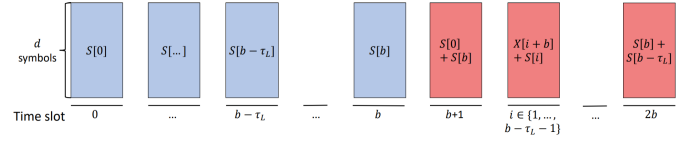


Fig. 13. The offline scheme for message size sequence 2 for case $\tau_L < b$ and $\tau_L = (\tau - b)$. Blue channel packets consist of message symbols and red channel packets consist of parity symbols. The numbers under the lines at the bottom indicate the time slots. The offline scheme sends d symbols in $X[b - \tau_L]$.

$(b - \tau_L)$ message packets when the corresponding channel packets are lost, and (b) $S[b]$ when both $X[b - \tau_L] = S[b - \tau_L]$ and $X[b] = S[b]$ are lost. The sum of $S[b - \tau_L]$ and $S[b]$ is sent in $X[2b]$ to ensure that $S[b - \tau_L]$ is recovered if $X[b - \tau_L]$ is dropped. The scheme is described in full detail below and shown in Figure 13:

- For $i \in [b - \tau_L] \cap \{b\}$, $S[i] = X[i]$.
- $X[b + 1] = (S[0] + S[b])$.
- For $i \in \{1, \dots, b - \tau_L - 1\}$, $X[i + b + 1] = (X[i + b] + S[i])$.
- $X[2b] = (S[b] + S[b - \tau_L])$.

Decoding: Each message packet is transmitted within the corresponding channel packet and is decoded when the transmission is lossless. We now discuss how each message packet is decoded within a delay of τ under lossy conditions. Either $X[0] = S[0]$ is received or both $X[b] = S[b]$ and $X[b + 1] = (S[0] + X[b])$ are received. Consequently, $S[0]$ is decoded within delay $(b + 1 \leq \tau)$. For $i \in \{1, \dots, b - \tau_L - 1\}$, either $X[i] = S[i]$ is received, or both $X[i + b]$ and $X[i + b + 1] = (X[i + b] + S[i])$ is received. Therefore, each $S[i]$ is recovered within delay $(b + 1 \leq \tau)$. Either $X[b - \tau_L] = S[b - \tau_L]$ is received, or $X[2b - \tau_L] = (S[b] + P^{b - \tau_L - 1} S[i])$ is received. In the latter case, $S[0], \dots, S[b - \tau_L - 1]$ are decoded by time slot $(b - \tau_L + \tau)$ and combined with $X[2b - \tau_L]$ to decode $S[b]$. Then, $S[b]$ and $X[2b] = (S[b] + S[b - \tau_L])$ used to recover $S[b - \tau_L]$. Therefore, $S[b - \tau_L]$ is decoded within delay τ . Either $X[b] = S[b]$ is received, or $X[2b] = (S[b] + S[b - \tau_L])$ is received. In the latter case, subtracting $S[b - \tau_L]$ yields $S[b]$. Hence, $S[b]$ is recovered within a delay of τ .

Proof of the converse result: The offline-optimal-rate is at least $R_t^{(1)}$ and $R_t^{(2)}$ on message size sequences 1 and 2, respectively (i.e., the rate of the offline scheme from Equation 27). Next, we present necessary and mutually exclusive conditions on the total number of symbols sent in $X[0], \dots, X[b - 1]$ for a code construction to attain rates at least $R_t^{(1)}$ and $R_t^{(2)}$ on the two respective message size sequences. The two message size sequences are the same until time slot b . Therefore, no online coding scheme can satisfy the condition for both message size sequences.

Condition for rate $R_t^{(1)}$ on message size sequence 1: Consider an arbitrary coding scheme for message size sequence 1. At least $d(b - \tau_L + 1)$ symbols are transmitted in $X[b], \dots, X[t]$ since $X[0], \dots, X[b - 1]$ could be dropped in a burst. At most, an additional $d(b - \tau_L + .5)$ symbols can be sent over $X[0], \dots, X[b - 1]$ if the rate is at least $R_t^{(1)}$.

Condition for rate $R_t^{(2)}$ on message size sequence 2: Consider any coding scheme for message size sequence 2.

We will show that if

$$d' = \sum_{i=0}^{b-1} n_i \leq d(b - \tau_L + .5) \quad (28)$$

then the rate is strictly less than $R_t^{(2)}$. At a high level, at least $d(b - \tau_L + 2)$ symbols are sent in $X[0], \dots, X[b - 1], X[2b], \dots, X[t]$ to satisfy the worst-case-delay constraint when $X[b], \dots, X[2b - 1]$ are lost. At least $d(b - \tau_L + 1.5)$ symbols must be sent in $X[b], \dots, X[2b - 1]$ for the lossless-delay and worst-case-delay constraints to be satisfied, as will be shown shortly. In total, $d(2b - 2\tau_L + 3.5)$ symbols are sent, whereas at most $d(2b - 2\tau_L + 3)$ symbols are transmitted as part of a scheme with a rate of at least $R_t^{(2)}$.

Next, the fact that sending at most $d(b - \tau_L + .5)$ symbols over $X[0], \dots, X[b - 1]$ leads to a rate of less than $R_t^{(2)}$ on message size sequence 2 is proven in detail. Let S be a random variable drawn uniformly at random from the finite field F_q . Recall from Appendix A that for any $i \in [t]$, (a) $H(S[i]) = H(S)k_i$, and (b) $H(X[i]) \leq H(S)n_i$ (Equations 22 and 23).

We provide an upper bound on the sizes of the channel packets as follows

$$d(b - \tau_L + 2)H(S) = H(S[0 : b]) \leq \quad (29)$$

$$H(S[0 : b], X[0 : b - 1], X[2b : b + \tau]) = \quad (30)$$

$$H(X[0 : b - 1], X[2b : b + \tau]) + H(S[0 : b]X[0 : b - 1], X[2b : b + \tau]) = \quad (31)$$

$$H(X[0 : b - 1], X[2b : b + \tau]) \leq \quad (32)$$

$$H(S) \sum_{i=0}^{b-1} n_i + \sum_{i=2b}^{b+\tau} n_i. \quad (33)$$

Equation 29 follows from Equation 22, Equation 30 follows from the definition of entropy, Equation 31 follows from the chain rule, Equation 32 follows from the worst-case-delay constraint, and Equation 33 follows from Equation 23.

Next, we will prove that $H(X[b : 2b - 1]) \geq d(b - \tau_L + 1.5)H(S)$ as follows

$$\begin{aligned} H(X[0 : b - 1], S[0 : b - \tau_L - 1]) &= \\ H(X[0 : b - 1]) + H(S[0 : b - \tau_L - 1]X[0 : b - 1]) &= \end{aligned} \quad (34)$$

$$H(X[0 : b - 1]) \leq d'H(S) \quad (35)$$

$$\begin{aligned} H(X[0 : b - 1], S[0 : b - \tau_L - 1]) &= \\ H(S[0 : b - \tau_L - 1]) + \\ H(X[0 : b - 1]S[0 : b - \tau_L - 1]) &= \end{aligned} \quad (36)$$

$$d(b - \tau_L)H(S) + H(X[0 : b - 1]S[0 : b - \tau_L - 1]) \quad (37)$$

where Equation 34 follows from the chain rule, Equation 35 follows from the lossless-delay constraint and Equation 28, Equation 36 follows from the chain rule, and Equation 37 follows from applying Equation 22 to $S[0], \dots, S[b - \tau - 1]$.

Rearranging terms yields

$$H(X[0 : b - 1]S[0 : b - \tau_L - 1]) \leq (d' - d(b - \tau_L))H(S) \quad (38)$$

Next, we bound the sizes of $X[b], \dots, X[2b - 1]$ using

$$d(b - \tau_L + 2)H(S) \leq H(S[0 : b]) \leq \quad (39)$$

$$H(S[0 : b], X[0 : 2b - 1]) \leq \quad (40)$$

$$\begin{aligned} H(X[b : 2b - 1]) + H(S[0 : b - \tau_L - 1]X[b : 2b - 1] + \\ H(X[0 : b - 1]S[0 : b - \tau_L - 1]) + \\ H(S[b - \tau_L : b]X[0 : 2b - 1]) = \end{aligned} \quad (41)$$

$$H(X[b : 2b - 1]) + H(X[0 : b - 1]S[0 : b - \tau_L - 1]) \quad (42)$$

$$\leq H(X[b : 2b - 1]) + (d' - d(b - \tau_L))H(S), \quad (43)$$

where Equation 39 follows from Equation 22, Equation 40 follows from the definition of entropy, Equation 41 follows from the definition of conditioning, Equation 42 follows from the worst-case-delay constraint (i.e., $\tau = (\tau_L + b)$) and the lossless-delay (i.e., $\tau_L < b$), and Equation 43 follows from Equation 38.

Rearranging terms yields

$$\begin{aligned} (d(2b - 2\tau_L + 2) - d')H(S) &\leq H(X[b : 2b - 1]) \\ &\leq H(S) \sum_{i=b}^{2b-1} n_i. \end{aligned} \quad (44)$$

The total number of symbols sent in $X[0 : b - 1]$ and $X[2b : b + \tau]$ is at least $d(b - \tau_L + 2)$ by Equations 29 through 33. At least $(d(2b - 2\tau_L + 2) - d')$ symbols are sent in $X[b : 2b - 1]$ by Equation 44. In total, at least

$$\begin{aligned} d(3b - 3\tau_L + 4) - d' &\geq d(3b - 3\tau_L + 4) - d(b - \tau_L + .5) \\ &= d(2b - 2\tau_L + 3.5) \end{aligned}$$

symbols are sent. Thus, the rate is strictly lower than $R_t^{(2)}$.

Summary : Any online scheme with rate at least $R_t^{(1)}$ on message size sequence 1 sends at most $d(b - \tau_L + .5)$ symbols over $X[0], \dots, X[b - 1]$. Consequently, its rate is strictly less than $R_t^{(2)}$ on message size sequence 2.

D. Proof of Theorem 3 case $\tau_L < (\tau - b)$

Let d be an arbitrary positive even integer. Consider the following two message size sequences for which the offline construction will be shown below in Figures 14 and 15 respectively:

- 1) $k_0^{(1)} = \dots = k_{b-1}^{(1)} = d$, and $k_b^{(1)} = \dots = k_t^{(1)} = 0$.
- 2) $k_0^{(2)} = \dots = k_{\tau-\tau_L-2}^{(2)} = d$, $k_{\tau-\tau_L-1}^{(2)} = d(\tau_L + 1)$, and $k_{\tau-\tau_L}^{(2)} = \dots = k_t^{(2)} = 0$.

Before we present the details of the proof, we point out that a similar proof applies to when the sizes of the message packets are approximately equal to those of the message size sequences, as long as the deviation is small relative to d . In addition, the proof extends to scenarios where one of the two above message size sequence occurs at some point in the transmission proceeded and followed by τ message packets whose sizes are much less than d .

We will describe an offline coding scheme for message size sequences 1 and 2 with rates

$$R_t^{(1)} = \frac{b}{2b - .5}, \quad R_t^{(2)} = \frac{\tau}{\tau + b} \quad (45)$$

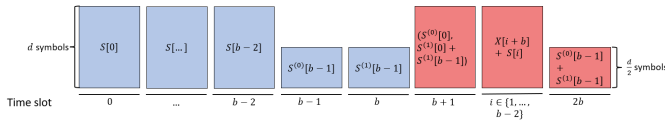


Fig. 14. The offline scheme for message size sequence 1 for case $\tau_L < (\tau - b)$. Blue channel packets consist of message symbols, and red channel packets consist of parity symbols. The numbers under the lines at the bottom indicate the time slots. The offline scheme sends $\frac{d}{2}$ symbols in $X[b-1]$.

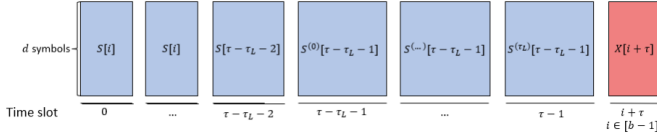


Fig. 15. The offline scheme for message size sequence 2 for case $\tau_L < (\tau - b)$. Blue channel packets consist of message symbols and red channel packets consist of parity symbols. The numbers under the lines at the bottom indicate the time slots. The offline scheme sends d symbols in $X[b-1]$.

on the two respective message size sequences. We also verify that the lossless-delay and worst-case-delay constraints are satisfied.

Offline scheme for message size sequence 1: Each of $S[0], \dots, S[b-2]$ is transmitted immediately as part of the corresponding channel packet. Then $S[b-1]$ is divided in half and evenly sent over $X[b-1]$ and $X[b]$. The next $(b-1)$ channel packets each comprise d parity symbols. These $d(b-1)$ parity symbols are used to decode (a) message packets $S[0], \dots, S[b-2]$ when the corresponding channel packets are lost, and (b) $X[b]$ when both $X[b-1]$ and $X[b]$ are lost. The summation of $X[b-1]$ and $X[b]$ is sent in $X[2b]$ to ensure that $S[b-1]$ is decoded within a delay of τ . The scheme is described in detail below and shown in Figure 14 :

- The message packets $S[0]$ and $S[b-1]$ are divided in half into $S[0] = (S^{(0)}[0], S^{(1)}[0])$ and $S[b-1] = (S^{(0)}[b-1], S^{(1)}[b-1])$.
- For $j \in [b-2]$, $X[j] = S[j]$.
- $X[b-1] = S^{(0)}[b-1]$.
- $X[b] = S^{(1)}[b-1]$.
- $X[b+1] = (S^{(0)}[0], S^{(1)}[0] + S^{(1)}[b-1])$.
- For $i \in \{1, \dots, b-2\}$, $X[i+b+1] = (X[i+b] + S[i])$.
- $X[2b] = (S^{(0)}[b-1] + S^{(1)}[b-1])$.

Decoding: Each message packet is sent within the current and perhaps next channel packets and is decoded when the transmission is lossless. We now discuss how each message packet is decoded within delay τ under lossy conditions. Either $X[0] = S[0]$ is received, or both $X[b] = S_1[b-1]$ and $X[b+1] = (S^{(0)}[0], S^{(1)}[0] + S^{(1)}[b-1])$ are received. Thus, $S[0]$ is decoded within a delay of $(b+1 \leq \tau)$. For $j \in \{1, \dots, b-2\}$, either $X[j] = S[j]$ is received, or both $X[j+b]$ and $X[j+b+1] = (X[j+b] + S[j])$ are received. Therefore, $S[j]$ is decoded within delay $(b+1 \leq \tau)$. Either $X[b-1] = S^{(0)}[b-1]$ is received, or $X[2b-1]$ is received. In the latter case, $S[0], \dots, S[b-2]$ are decoded by time slot $(2b-1)$ and are combined with $X[2b-1] = (S^{(0)}[0], S^{(1)}[0] + S^{(1)}[b-1]) + \sum_{i=1}^{b-2} S[i]$

to recover $S^{(1)}[b-1]$. The receiver then decodes $S^{(0)}[b-1] = (X[2b] - S^{(1)}[b-1])$ within delay $(b+1 \leq \tau)$. Either $X[b] = S^{(1)}[b-1]$ is received, or $X[2b] = (S^{(0)}[b-1] + S^{(1)}[b-1])$ is received and combined with $S^{(0)}[b-1]$ to recover $S^{(1)}[b-1]$ within delay τ .

Offline scheme for message size sequence 2: Each of $S[0], \dots, S[\tau - \tau_L - 2]$ is transmitted within the corresponding channel packet. The symbols of $S[\tau - \tau_L - 1]$ are evenly divided into $(\tau_L + 1)$ components sent over $X[\tau - \tau_L - 1], \dots, X[\tau - 1]$ respectively. Each of $X[\tau], \dots, X[\tau + b - 1]$ comprises d symbols, which creates d blocks of the $[\tau + b, \tau]$ systematic block codes (described in Section B). The scheme is presented in detail below and shown in Figure 15:

- For $j \in [\tau - \tau_L - 2]$, $X[j] = S[j]$.
- The message packet $S[\tau - \tau_L - 1]$ is evenly divided into $(\tau_L + 1)$ components of size d : $(S^{(0)}[\tau - \tau_L - 1], \dots, S^{(\tau_L)}[\tau - \tau_L - 1])$.
- For $j \in \{\tau - \tau_L - 1, \dots, \tau - 1\}$, $X[j] = S^{(j - \tau + \tau_L + 1)}[\tau - \tau_L - 1]$.
- For each $z \in [d - 1]$, an instance of the block code from [8] is created that maps $(X_z[0], \dots, X_z[\tau - 1])$ to $(X_z[0], \dots, X_z[\tau - 1], p_0^{(z)}, \dots, p_{b-1}^{(z)})$.
- For $j \in [b - 1]$, $X[\tau + j] = (p_j^{(0)}, \dots, p_j^{(d-1)})$.

Decoding: Each message packet is sent over the current and perhaps next τ_L channel packets and is decoded when the transmission is lossless. Under lossy conditions, the block code $(X_z[0], \dots, X_z[\tau - 1], p_0^{(z)}, \dots, p_{b-1}^{(z)})$ is used for decoding. For $z \in [d - 1]$: (a) Each symbol $X_z[i]$, for $i \in [b - 1]$, is decoded within a delay of τ . (b) Each symbol $X_z[i]$, for $i \in [\tau - 1] \setminus [b - 1]$, is decoded by time slot $(\tau + b - 1)$. Thus, the worst-case-delay constraint is satisfied.

Proof of the converse result : The rates $R_t^{(1)}$ and $R_t^{(2)}$ of the above construction (Equation 45) for message size sequences 1 and 2, respectively, serve as a lower bound on the offline-optimal-rate for the two message size sequences. Next, we present mutually exclusive conditions on the number of symbols transmitted in the first b channel packets to have rates at least $R_t^{(1)}$ or $R_t^{(2)}$ on message size sequences 1 or 2, respectively. The online coding schemes cannot differentiate between the two message size sequences before the time slot b . Hence, the number of symbols sent in $X[0], \dots, X[b-1]$ by any online scheme violates the condition for at least one message size sequence.

Condition for rate $R_t^{(1)}$ on message size sequence 1 : Consider any coding scheme for message size sequence 1. At least db symbols are transmitted in $X[b], \dots, X[t]$ in case there is a burst loss of $X[0], \dots, X[b-1]$. The rate is at least $R_t^{(1)}$, so at most $d(b - .5)$ additional symbols are sent in $X[0], \dots, X[b-1]$.

Condition for rate $R_t^{(2)}$ on message size sequence 2 : Consider any coding scheme for message size sequence 2. We will demonstrate that if

$$\sum_{i=0}^{b-1} n_i \leq d(b - .5) \quad (46)$$

then the rate is strictly less than $R_t^{(2)} = \frac{\tau}{\tau+b}$ in two steps. First, we will show that all symbols are transmitted by $X[\tau + b - 1]$ without loss of generality. Second, we prove that strictly more than db symbols may be lost. At least $d\tau$ additional symbols are sent to meet the worst-case-delay constraint, leading to a lower rate than $R_t^{(2)}$.

Step 1: If $X[\tau + b - 1]$ is lost, then $X[0 : \tau - 1]$ are received, which yields $S[0 : \tau - \tau_L - 1]$ by the lossless-delay constraint. Thus, all symbols sent after the time slot $(\tau + b)$ can instead be sent in $X[\tau + b - 1]$.

Step 2: Consider the following erasure channels C_i for $i \in [\tau + b - 1]$. Each C_i introduces bursts of packet losses in $\{X[j], \dots, X[j + b - 1] \mid j \equiv i \pmod{\tau + b}\}$ and results in l_i lost (dropped) symbols.¹² At least $d(\tau + b)$ symbols are sent in total due to the upper bound on the rate of $\frac{\tau}{\tau+b}$, leading to

$$\sum_{i=0}^{\tau} x^{b-1} l_i \geq db(\tau + b) \quad (47)$$

$$\sum_{i=1}^{\tau} x^{b-1} l_i \geq db(\tau + b - 1) + .5d \quad (48)$$

$$\frac{1}{\tau + b - 1} \sum_{i=1}^{\tau} x^{b-1} l_i \geq db + \frac{.5d}{\tau + b - 1}, \quad (49)$$

where Equation 47 follows from each packet (and hence each symbol) being dropped by exactly b channels, and Equation 48 follows from Equation 46.

Hence, there is some $i \in \{1, \dots, \tau + b - 1\}$ for which $l_i \geq (db + \frac{.5d}{\tau + b - 1})$. In order to satisfy the worst-case-delay constraint over channel C_i , at least $d\tau$ symbols are received outside of the channel packets dropped by C_i . Thus, the total number of symbols sent is at least $d(\tau + b + \frac{.5}{\tau + b - 1})$. In contrast, at most $d(\tau + b)$ symbols are sent if the rate is at least $R_t^{(2)}$.

Summary : Any online coding scheme with a rate of at least $R_t^{(1)}$ on message size sequence 1 sends at most $d(b - .5)$ symbols in $X[0], \dots, X[b-1]$. Consequently, its rate is strictly lower than $R_t^{(2)}$ on message size sequence 2.

ACKNOWLEDGMENT

The authors would like to thank Francisco Maturana for his feedback in editing this work.

REFERENCES

- [1] M. Rudow and K. V. Rashmi, "Online versus offline rate in streaming codes for variable-size messages," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2020, pp. 509–514.
- [2] A. Badr, A. Khisti, W.-T. Tan, and J. Apostolopoulos, "Perfecting protection for interactive multimedia: A survey of forward error correction for low-delay interactive applications," *IEEE Signal Process. Mag.*, vol. 34, no. 2, pp. 95–113, Mar. 2017.
- [3] E. Martinian and C. E. W. Sundberg, "Burst erasure correction codes with low decoding delay," *IEEE Trans. Inf. Theory*, vol. 50, no. 10, pp. 2494–2502, Oct. 2004.
- [4] E. Martinian and M. Trott, "Delay-optimal burst erasure code construction," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2007, pp. 1006–1010.

¹²A similar argument was used to show the upper bound on rate of $\frac{\tau}{\tau+b}$ in [4].

- [5] A. Badr, P. Patil, A. Khisti, W.-T. Tan, and J. Apostolopoulos, "Layered constructions for low-delay streaming codes," *IEEE Trans. Inf. Theory*, vol. 63, no. 1, pp. 111–141, Jan. 2017.
- [6] S. L. Fong, A. Khisti, B. Li, W.-T. Tan, X. Zhu, and J. Apostolopoulos, "Optimal streaming codes for channels with burst and arbitrary erasures," *IEEE Trans. Inf. Theory*, vol. 65, no. 7, pp. 4274–4292, Jul. 2019.
- [7] M. N. Krishnan and P. V. Kumar, "Rate-optimal streaming codes for channels with burst and isolated erasures," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 1809–1813.
- [8] M. N. Krishnan, D. Shukla, and P. V. Kumar, "A quadratic field-size rate-optimal streaming code for channels with burst and random erasures," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2019, pp. 852–856.
- [9] D. Dudzicz, S. L. Fong, and A. Khisti, "An explicit construction of optimal streaming codes for channels with burst and arbitrary erasures," *IEEE Trans. Commun.*, vol. 68, no. 1, pp. 12–25, Jan. 2020.
- [10] M. N. Krishnan, D. Shukla, and P. V. Kumar, "Rate-optimal streaming codes for channels with burst and random erasures," *IEEE Trans. Inf. Theory*, vol. 66, no. 8, pp. 4869–4891, Aug. 2020.
- [11] E. Domanovitz, S. L. Fong, and A. Khisti, "An explicit rate-optimal streaming code for channels with burst and arbitrary erasures," *IEEE Trans. Inf. Theory*, vol. 68, no. 1, pp. 47–65, Jan. 2022.
- [12] A. Badr, D. Lui, A. Khisti, W.-T. Tan, X. Zhu, and J. Apostolopoulos, "Multiplexed coding for multiple streams with different decoding delays," *IEEE Trans. Inf. Theory*, vol. 64, no. 6, pp. 4365–4378, Jun. 2018.
- [13] S. L. Fong, A. Khisti, B. Li, W.-T. Tan, X. Zhu, and J. Apostolopoulos, "Optimal multiplexed erasure codes for streaming messages with different decoding delays," *IEEE Trans. Inf. Theory*, vol. 66, no. 7, pp. 4007–4018, Jul. 2020.
- [14] A. Badr, A. Khisti, and E. Martinian, "Diversity embedded streaming erasure codes (DE-SCo): Constructions and optimality," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 5, pp. 1042–1054, May 2011.
- [15] A. Badr, D. Lui, and A. Khisti, "Streaming codes for multicast over burst erasure channels," *IEEE Trans. Inf. Theory*, vol. 61, no. 8, pp. 4181–4208, Aug. 2015.
- [16] M. Haghifam, M. N. Krishnan, A. Khisti, X. Zhu, W.-T. Tan, and J. Apostolopoulos, "On streaming codes with unequal error protection," *IEEE J. Sel. Areas Inf. Theory*, vol. 2, no. 4, pp. 1165–1179, Dec. 2021.
- [17] N. Adler and Y. Cassuto, "Burst-erasure correcting codes with optimal average delay," *IEEE Trans. Inf. Theory*, vol. 63, no. 5, pp. 2848–2865, May 2017.
- [18] D. Leong and T. Ho, "Erasure coding for real-time streaming," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2012, pp. 289–293.
- [19] D. Leong, A. Qureshi, and T. Ho, "On coding for real-time streaming under packet erasures," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2013, pp. 1012–1016.
- [20] A. Badr, A. Khisti, W. T. Tan, and J. Apostolopoulos, "Streaming codes with partial recovery over channels with burst and isolated erasures," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 3, pp. 501–516, Apr. 2015.
- [21] S. L. Fong, A. Khisti, B. Li, W.-T. Tan, X. Zhu, and J. Apostolopoulos, "Optimal streaming erasure codes over the three-node relay network," *IEEE Trans. Inf. Theory*, vol. 66, no. 5, pp. 2696–2712, May 2020.
- [22] A. Badr, A. Khisti, W.-T. Tan, X. Zhu, and J. Apostolopoulos, "FEC for VoIP using dual-delay streaming codes," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2017, pp. 1–9.
- [23] Z. Li, A. Khisti, and B. Girod, "Correcting erasure bursts with minimum decoding delay," in *Proc. 45th Asilomar Conf. Signals, Syst. Comput. (ASILOMAR)*, Nov. 2011, pp. 33–39.
- [24] Y. Wei and T. Ho, "On prioritized coding for real-time streaming under packet erasures," in *Proc. 51st Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Oct. 2013, pp. 327–334.
- [25] P.-W. Su, Y.-C. Huang, S.-C. Lin, I.-H. Wang, and C.-C. Wang, "Random linear streaming codes in the finite memory length and decoding deadline regime," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2021, pp. 730–735.
- [26] M. Haghifam, A. Badr, A. Khisti, X. Zhu, W.-T. Tan, and J. Apostolopoulos, "Streaming codes with unequal error protection against burst losses," in *Proc. 29th Biennial Symp. Commun. (BSC)*, Jun. 2018, pp. 1–5.

- [27] M. Rudow and K. V. Rashmi, "Streaming codes for variable-size messages," *IEEE Trans. Inf. Theory*, vol. 68, no. 9, pp. 5823–5849, Sep. 2022.
- [28] M. N. Krishnan, V. Ramkumar, M. Vajha, and P. V. Kumar, "Simple streaming codes for reliable, low-latency communication," *IEEE Commun. Lett.*, vol. 24, no. 2, pp. 249–253, Feb. 2019.
- [29] M. N. Krishnan, G. K. Facenda, E. Domanovitz, A. Khisti, W.-T. Tan, and J. Apostolopoulos, "High rate streaming codes over the three-node relay network," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Oct. 2021, pp. 1–6.
- [30] E. Domanovitz, A. Khisti, W.-T. Tan, X. Zhu, and J. Apostolopoulos, "Streaming erasure codes over multi-hop relay network," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2020, pp. 497–502.
- [31] M. Rudow and K. V. Rashmi, "Learning-augmented streaming codes are approximately optimal for variable-size messages," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2022, pp. 474–479.

K. V. Rashmi received the Ph.D. degree from the University of California at Berkeley, CA, USA, in 2016. She was a Post-Doctoral Scholar at UC Berkeley from 2016 to 2017. She is currently an Assistant Professor with the Computer Science Department, Carnegie Mellon University. Her research interests include information/coding theory and computer/networked systems. She was a recipient of the Facebook Communications and Networking Research Award in 2017, the Google Faculty Research Award in 2018, the Facebook Distributed Systems Research Award in 2019, the Tata Institute of Fundamental Research Memorial Lecture Award in 2020, the VMware Systems Research Award in 2021, the Meta Research Award in 2022, the Sloan Research Fellowship in 2023, the NSF CAREER Award from 2020 to 2025, the USENIX NSDI 2021 Community (Best Paper) Award, and the IEEE Data Storage Best Paper and the Best Student Paper Awards from 2011 to 2012. Her Ph.D. thesis was awarded the UC Berkeley Eli Jury Dissertation Award in 2016, the Facebook Fellowship from 2012 to 2013, the Microsoft Research Ph.D. Fellowship from 2013 to 2015, and the Google Anita Borg Memorial Scholarship from 2015 to 2016.

Michael Rudow received the bachelor's degree in computer science and the master's degree in mathematics from the University of Pennsylvania, Philadelphia, PA, USA. He is currently pursuing the Ph.D. degree with the Computer Science Department, Carnegie Mellon University, advised by Prof. Rashmi Vinayak. He is broadly interested in the intersection of coding theory and machine learning.