Throughput and Delay Analysis for Coded ARQ

Derya Malak, Ohad Elishco, and Muriel Médard
Research Laboratory of Electronics, MIT, Cambridge, MA, USA
Email: {deryam, ohadeli, medard}@mit.edu

Edmund M. Yeh
Northeastern University, Boston, MA, USA
Email: eyeh@ece.neu.edu

***Abstract*—We propose a Coded selective-repeat ARQ protocol with cumulative feedback, by building on the uncoded baseline scheme for ARQ, developed by Ausavapattanakun and Nosratinia. Our method leverages discrete-time queuing and coding theory to analyze the performance of the proposed data transmission method. We incorporate forward error-correction (FEC) to reduce in-order delivery delay, and exploit a matrix signal-flow graph approach to analyze the throughput and delay. We demonstrate and contrast the performance of the Coded ARQ protocol with that of the uncoded ARQ scheme, with minimum coding, i.e., with a sliding window of size 2. Coded ARQ can provide gains up to about 40% in terms of throughput. It also provides delay guarantees, and is robust to various challenges such as imperfect and delayed feedback, burst erasures, and round-trip time fluctuations.**

I. Introduction

Automatic Repeat reQuest (ARQ) and hybrid ARQ (HARQ) methods have been used in 5G mobile networks [1], to boost the performance of wireless technologies such as HSPA, WiMax and LTE [2]. HARQ technique combines the important features of both forward error-correction (FEC) and ARQ error control. A review on HARQ mechanisms that provide robustness in 4G LTE networks is given in [3]. A network-coding-based HARQ algorithm for video broadcast over wireless networks is proposed in [4]. ARQ and HARQ protocols perform together, and provide the system with reliable packet delivery over non-deterministic channel conditions. Here, failure in the Media Access Control layer HARQ operation is compensated for by the radio link control layer ARQ in acknowledged mode at the expense of extra latency for the packet [5].

Unreliable feedback in ARQ has been studied in [2], where a new method of acknowledging packet delivery for retransmission protocols is proposed. The proposed method is based on backwards composite acknowledgment from multiple packets. Compared to ARQ, the scheme exhibits increases reliability under varying channel conditionns, at the cost of a small increase in average experienced delay.

The role of the feedback channel is to limit retransmissions and increase data channel efficiency. Inevitable feedback channel impairments may cause unreliability in packet delivery. Attempts to increase feedback reliability, e.g., by means of repetition coding, is costly to the receiver node while erroneous feedback detection may increase packet delivery latency and diminish throughput and reliability. In LTE, blind HARQ retransmissions of a packet are proposed to avoid feedback complexity and increase reliability [6]. However, this approach can severely decrease resource utilization efficiency.

Uncoded SR ARQ protocols via signal-flow graphs have been analyzed in [7]. Scalar-flow graphs have been used to find the moment generating functions (MGFs) for the transmission and delay times in [8]. Matrix signal-flow graphs (MSFGs) have been extensively used in the state-space formulation of feedback theory [9]. They can be used to model channel erasures, incorporating unreliable feedback.

Different classes of codes have been proposed to correct errors over packet erasure channels. Block codes require a bit/packet stream to be partitioned into blocks, each block being treated independently from the rest. Block codes for error correction have been considered in [10]. Streaming codes, e.g. convolutional codes, have the flexibility of grouping the blocks of information in an appropriate way, and decoding the part of the sequence with fewer erasures. They can correct more errors than classical block codes when considering the erasure channel [10], [11]. Fountain codes have efficient encoding and decoding algorithms, and are capacity-achieving. However, they are not suitable for streaming because the decoding delay is proportional to the size of the data [12].

Using FEC, in-order delivery delay over packet erasure channels can be reduced [10], and the performance of SR ARQ protocols can be boosted. Delay bounds for convolutional codes have been provided in [13]. Packet dropping to reduce playback delay of streaming over an erasure channel is investigated in [14]. Delay-optimal codes without feedback for burst erasure channels, and the decoding delay of codes for more general erasure models have been analyzed in [15].

In this paper, we use a MSFG approach to analyze the throughput and delay performance of Coded SR ARQ protocols over packet erasure channels with unreliable feedback. Erasures can occur in both the forward and reverse channels. However, an acknowledgment (ACK) cannot be decoded as a negative acknowledgment (NACK), and vice versa. We propose a maximum distance separable (MDS) Coded ARQ scheme, by building on the uncoded baseline scheme proposed in [7]. In our coded model, feedback is cumulative, i.e. it acknowledges all the previously transmitted packets. Contrasting the throughput and delay performance of Coded ARQ with the uncoded ARQ scheme, we demonstrate that with minimum coding, i.e., with a sliding window of size 2, Coded ARQ can provide gains up to about 40% in terms of throughput, given the unreliability of the feedback. Coding also has benefits under burst erasures or higher erasure rates. Coded ARQ is

This work was supported in part by the DARPA Program under Prime Award HR0011-17-C-0050, and in part by the Intel Corporation.

more predictable across statistics, hence is more stable.

II. System Model

We have a point-to-point channel model consisting of a sender and a receiver. On the forward link, the sender attempts to transmit a packet to the receiver, and upon the successful reception of the packet, on the reverse link, the receiver acknowledges the sender by transmitting a feedback. The status of a transmission at time t is a Bernoulli random variable taking values in $\mathcal{X} = \{0, 1\}$, where 0 denotes an error-free packet, and 1 means the packet is erased. The erasure rate ϵ is a function of channel condition. Both for the forward and reverse links, we use a Gilbert-Elliott (GE) channel model [16], which is a binary-state Markov process S_t with states G (good) and B (bad)[1], i.e. $\mathcal{S} = \{G, B\}$, and probability transition matrix $\mathbf{P}$. The packet erasure rates in states G and B are ϵ_G and ϵ_B, respectively. We let $\boldsymbol{\epsilon} = [\epsilon_G, \epsilon_B]$. Since the channel state is not the same as the channel observation, the process X_t is a hidden Markov model[2] (HMM), which is driven by the process S_t.

The channel state information is not available at the sender and the receiver. Hence, the sender does not know the state of the forward link at time t, but it observes the status of the feedback at time $t-1$, which is a Bernoulli random variable. Similarly, the receiver does not know the status of the reverse link, but it observes the status of a transmission at time t. The joint probabilities of channel state and observation at time t is computed using the state-transition matrix of the GE channel:

$$\mathbf{P} = \begin{bmatrix} 1-q & q \\ r & 1-r \end{bmatrix}, \tag{1}$$

where the first and second rows correspond to the transition probabilities of states G and B, given the channel state at time $t-1$. Solving $\pi\mathbf{P} = \pi$ and $\pi\mathbf{1} = 1$, where $\mathbf{1}$ is a column vector of ones, the stationary vector of $\mathbf{P}$ is $\pi = [\frac{r}{r+q}, \frac{q}{r+q}]$. Hence, the erasure rate is $\epsilon = \pi\boldsymbol{\epsilon}^\intercal$. Given r, ϵ_G, ϵ_B, and ϵ, we have $q = r\big(\frac{\epsilon_B - \epsilon_G}{\epsilon_B - \epsilon} - 1\big)$. Note that $1/r$ represents the average erasure burst, and burst erasures occur when r is low. The joint probabilities of channel state and observation at time t, given the channel state at time $t-1$, are given as

$$\begin{aligned}&\mathbb{P}(S_t = j, X_t = 1 | S_{t-1} = i) \\ &= \mathbb{P}(S_t = j | S_{t-1} = i)\mathbb{P}(X_t = 1 | S_t = j) = p_{ij}\epsilon_j.\end{aligned}$$

Let $\mathbf{P}_1 = \mathbf{P} \cdot \mathrm{diag}\{\boldsymbol{\epsilon}\}$ be the error matrix on the forward (or reverse) link. Similarly, $\mathbf{P}_0 = \mathbf{P} \cdot \mathrm{diag}\{\mathbf{1} - \boldsymbol{\epsilon}\}$ is the success matrix in either link. Note that $\mathbf{P}_0 + \mathbf{P}_1 = \mathbf{P}$. The entries of $\mathbf{P}_0$ and $\mathbf{P}_1$ are the joint state-transition probabilities given the channel observations [7]. Hence, the HMM can be characterized by $\{\mathcal{S}, \mathcal{X}, \mathbf{P}, \boldsymbol{\epsilon}\}$.

Consider the forward link $\{\mathcal{S}^{(f)}, \mathcal{X}^{(f)}, \mathbf{P}_0^{(f)}, \mathbf{P}_1^{(f)}\}$ and the reverse link $\{\mathcal{S}^{(r)}, \mathcal{X}^{(r)}, \mathbf{P}_0^{(r)}, \mathbf{P}_1^{(r)}\}$ that are mutually independent. The composite channel is characterized by $\{\mathcal{S}^{(c)}, \mathcal{X}^{(c)}, \mathbf{P}_{00}^{(c)}, \mathbf{P}_{01}^{(c)}, \mathbf{P}_{10}^{(c)}, \mathbf{P}_{11}^{(c)}\}$, where $\mathcal{S}^{(c)} = \mathcal{S}^{(f)} \times \mathcal{S}^{(r)}$ are the composite channel states, i.e. the Cartesian product of forward and reverse states, and $\mathcal{X}^{(c)} = \mathcal{X}^{(f)} \times \mathcal{X}^{(r)} = \{00, 01, 10, 11\}$ is the combined observation set. Note that $X_t^{(c)} = 00$ means both the forward and reverse links are good, while $X_t^{(c)} = 10$ means the forward link is erroneous and the reverse link is good. For $X_t^{(c)} = 11$, the joint probability of the combined observation and the composite state at time t, given the composite state at time $t-1$, is

$$\mathbb{P}(S_t^{(c)} = (j, m), X_t^{(c)} = 11 | S_{t-1}^{(c)} = (i, k)) = p_{ij}^{(f)}\epsilon_j^{(f)}p_{km}^{(r)}\epsilon_m^{(r)}.$$

Using the Kronecker product notation $\otimes$, we have $\mathbf{P}_{ij}^{(c)} = \mathbf{P}_i^{(f)} \otimes \mathbf{P}_j^{(r)}$ for the combined observation at time t, i.e. $X_t^{(c)} = ij$, $i, j \in \mathcal{X}$. We assume that both the forward and the reverse channels have the same parameters[3] r, ϵ_G, ϵ_B, and ϵ. Hence, the state-transition matrix for both the forward and reverse channels is given by $\mathbf{P}$. In the rest of the paper, we will drop the superscript $^{(c)}$ and denote the observation probability matrices by $\mathbf{P}_{00}$, $\mathbf{P}_{01}$, $\mathbf{P}_{10}$ and $\mathbf{P}_{11}$. Similar to [7], let $\mathbf{P}_{0x} = \mathbf{P}_{00} + \mathbf{P}_{01}$ and $\mathbf{P}_{1x} = \mathbf{P}_{10} + \mathbf{P}_{11}$ be the probability matrices of success and error on the forward channel, respectively, and let $\mathbf{P}_{x0} = \mathbf{P}_{00} + \mathbf{P}_{10}$ and $\mathbf{P}_{x1} = \mathbf{P}_{01} + \mathbf{P}_{11}$ be the matrices of success and error on the reverse channel, respectively. Furthermore, the matrices $\mathbf{P}$, $\mathbf{P}_0$, and $\mathbf{P}_1$ will denote the composite channel matrices, i.e. the Kronecker product of the forward and reverse channel matrices. The matrices for the GE channel are provided in Appendix A.

III. Analysis of Throughput and Delay of ARQ

In this section, we describe the protocol for the proposed channel model, signal-flow graphs, as well as a primer on the MSFGs for throughput and delay of ARQ protocols. We analyze the throughput and guaranteeable delay of uncoded ARQ, and provide exact expressions for memoryless channels.

A. Protocol

We use a slotted Selective Repeat (SR) ARQ protocol for data transmission. With SR ARQ, the sender sends a number of packets specified by a window size without the need to wait for individual ACK from the receiver. SR ARQ allows the receiver to accept packets out of order, which can be stored in a buffer and sorted at the receiver to ensure in-order delivery. If there is full feedback, ARQ achieves 100% throughput and the lowest possible packet delay over an erasure channel, and it is composable across links [20]. However, when the network is lossy, link-by-link ARQ cannot achieve the capacity

[1] If channel reciprocity holds, then the link estimate of the reverse direction at the sender can directly be used for link adaptation in the forward link. In the current work, we leave the study of the link equivalence (similarity of the transfer functions of the forward and reverse links) as future work.

[2] HMM is a statistical Markov process with unobserved states [17]. Although the state is not directly observed, the output dependent on the state can be observed.

[3] The forward and reverse channels do not necessarily have the same erasure rates or parameters. In practice, data packets and feedback packets typically have different lengths and different coding levels. Furthermore, data packets generally travel downstream from the sender towards receivers, and feedback packets travel upstream from receivers to the sender [18]. Hence, to compensate the channel asymmetry, more bandwidth can be allocated on downlink. In [19], we have studied the role of asymmetric channel conditions, where links might have different erasure or burst rates, and demonstrated the role of cumulative feedback in improving throughput when forward link erasures dominate, and in reducing delay under bursty feedback.

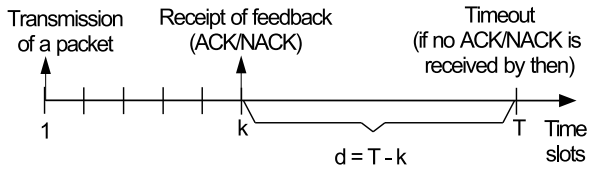

Fig. 1: (Uncoded) SR ARQ protocol description.

of a general network. The receiver may selectively reject the packets, and the sender individually retransmits packets that have timed out. All data packets are available at the transmitter prior to any transmission, and the receiver does not have buffer overflows. There is a handshake mechanism between the sender and receiver that initiates a synchronous transmission. After the start of transmission, the round-trip time (RTT) is k slots, i.e. it takes $k-1$ time slots between the transmission of a packet and receipt of its feedback – ACK/NACK sent by the receiver indicating if it has correctly received a data packet.

At the sender, a timeout mechanism is used to prevent deadlock. When a packet is (re)transmitted, the timeout associated with this packet is set to T, which is greater than or equal to the RTT k. Upon the reception of the first feedback, the waiting will be aborted after the timer expires, i.e., after $d = T - k$ slots. The feedback includes the information about all correctly received packets. The ACK/NACK is sent in each slot. Thus, the packet whose ACK is lost will be acknowledged by the subsequent ACKs/NACKs. If a succeeding ACK/NACK is successfully received before the timer expiration, the packet will not be retransmitted. Otherwise, i.e. if the timeout expires and no ACK is received, the sender retransmits the packet until it receives an ACK. Hence, we do not have an upper bound on the maximum number of retransmissions of a packet to guarantee its reliable delivery. When a packet is lost and its NACK is received, the packet will be retransmitted immediately. If the NACK is lost, the packet will be retransmitted after the timer expires. The protocol for uncoded ARQ is shown in Fig. 1.

B. Signal-Flow Graphs

A signal-flow graph is a diagram that consists of a set of nodes that denote the different states of the system, and a set of directed branches that represent the functional relationships among the states. The analysis of finite-state HMMs can be streamlined by using signal-flow graphs, and labeling the branches of flow graphs with observation probabilities [21], [22]. We next detail how to build the flow graphs for the analysis of SR ARQ.

In the current paper, the nodes of the flow graphs correspond to the states of the transmitter. Upon the initial state that a new packet is transmitted (input node, I), the transmitter goes from one state to the other. The output node (O) represents correct reception of ACK by the sender, and other nodes are hidden states. Upon the start of transmission, the transmitter goes from one state to the other. A certain value for the random variable X, that for example models the transmission or delay time for ARQ protocols as in [7], [8], [23], [24], corresponds to a state transition. A state transition is accompanied with the value for X, and its probability p, which appear in the branch gain as pz^X. Hence, the input-output gain of the graph is a polynomial in z, whose coefficients are the probabilities of corresponding values of X. This polynomial denotes the probability-generating function (PGF) for X, i.e., $\mathbb{E}[z^X]$. Flow graphs with vector node values and branches labeled with observation probability matrices are called matrix signal-flow graphs[4] (MSFGs) [7]. The graph can be simplified using the basic equivalence operations, i.e. parallel, series, and self-loop. Then, the input-output relationship is given by the matrix-generating function (MGF) $\mathbf{\Phi}(z)$.

C. Distributions of the Transmission Time and Delay

In this paper, we derive the MGFs for the transmission time and the delay of Coded ARQ protocol (Sect. IV). The transmission time τ is defined as the number of packets transmitted per successful packet, while the delay D is the time from when a packet is first transmitted to when its ACK is successfully received at the sender. Both τ and D are random variables with positive integer outcomes. The PGFs $\Phi_\tau(z)$ and $\Phi_D(z)$ of τ and D are derived using their MGFs by pre- and postmultiplications of row and column vectors, respectively. We will discuss how to obtain the MSFGs and the MGFs for the transmission time and delay in Sect. IV. We now discuss in detail how to obtain $\Phi(z)$'s from $\mathbf{\Phi}(z)$'s.

For the GE channel model, the probability vector of transmitting a new packet depends on the channel state. We assume that there are no erasures when the channel state is G, i.e. $\epsilon_G = 0$. Hence, the probability of transmitting a new packet in state G is $\pi_G(1-q)+\pi_B r$. Similarly, the probability of transmitting a new packet in state B is $(\pi_G q+\pi_B(1-r))(1-\epsilon_B)$. Hence, the probability vector of transmitting a new packet is $\pi_I = \pi\mathbf{P}_0 = [\pi_G(1-q)+\pi_B r,\ (\pi_G q+\pi_B(1-r))(1-\epsilon_B)]$.

a) Throughput Analysis: The MGF of the transmission time τ is calculated by left- and right-multiplying the matrix-generating function of τ, i.e., $\mathbf{\Phi}_\tau(z)$ with π_I and the column vector of ones:

$$\phi_\tau(z) = \frac{\pi_I \mathbf{\Phi}_\tau(z)\mathbf{1}}{\pi_I \mathbf{1}} = \frac{1}{1-\epsilon}\pi\mathbf{P}_0\mathbf{\Phi}_\tau(z)\mathbf{1}. \quad (2)$$

The average transmission time $\bar{\tau}$ is found by evaluating the derivative of $\phi_\tau(z)$ at $z=1$, as $\bar{\tau} = \phi'_\tau(1)$. The throughput is the reciprocal of the average transmission time, thus $\eta = 1/\bar{\tau}$.

b) Delay Analysis: The MGF of the delay D is given as

$$\phi_D(z) = \frac{\pi_{\mathbf{I}}\mathbf{\Phi}_D(z)\mathbf{1}}{\pi_{\mathbf{I}}\mathbf{1}}, \quad (3)$$

where $\mathbf{\Phi}_D(z)$ is the matrix-generating function of the delay. The average delay time $\bar{D}$ is found by evaluating the derivative of $\phi_D(z)$ at $z=1$, as $\bar{D} = \phi'_D(1)$.

[4]MSFGs have been extensively used in the state-space formulation of feedback theory [9]. They can also be used to model channel erasures, incorporating unreliable feedback.

In reality, the feedback is lossy and delayed, burst errors occur, and the fluctuations in the RTT can cause a high variability in the delay. To understand these effects, we exploit the three-sigma rule[5]. The 3σ heuristic is justifiable when the distribution of the delay is sub-Gaussian. A sub-Gaussian distribution has strong tail decay property since the tails decay at least as fast as the tails of a Gaussian. In [26], we show via numerical simulations that the tails of the delay distribution are dominated by the tails of a Gaussian distribution. In this case, the guaranteeable delay $\hat{D}$ of a protocol is upper bounded by the guaranteeable delay of a Gaussian distribution with the same mean and variance as the distribution of D.

We define the guaranteeable delay of the ARQ protocol – given that the distribution of the delay is sub-Gaussian – as

$$\hat{D} = \bar{D} + 3\sigma_D, \tag{4}$$

where $\bar{D}$ is the average delay, and σ_D^2 is the variance of the delay, which is evaluated as $\sigma_D^2 = \phi_D''(1) + \bar{D} - \bar{D}^2$, where $\phi_D''(1)$ is the second derivative of $\phi_D(z)$ at $z = 1$.

We denote by $\hat{D}_{\mathsf{ARQ}}(3\sigma)$, $\hat{D}_{\mathsf{C-ARQ}}(3\sigma)$ the guaranteeable delays of the uncoded and Coded ARQ protocols, respectively.

IV. Coded ARQ

In this section, we propose a Coded ARQ scheme, where the transmitted packets are coded. The coding scheme is similar to the generation-based random linear network coding in [27]. The sender has a coding bucket, and when it is ready to send a packet to the receiver, it produces a coded packet by forming a random linear combination of all the packets in the bucket. The encoded packet is then transmitted to the receiver. The receiver sends a cumulative feedback to indicate the set of successfully received encoded packets in the coding bucket. If the receiver successfully collects a sufficient number of encoded packets to decode all packets in the coding bucket, and the sender successfully receives the cumulative ACK message, it then purges the successfully ACK'ed encoded packets in the coding bucket and partially updates the coding bucket by moving new packets. While the performance of the reverse link can also be boosted with coding, the analysis of the protocol becomes more complex. We also expect that the gain of coding would be incremental compared to the gain of the cumulative feedback of the seen packets. The extension of the model to include coded feedback is left as future work.

We consider minimum coding, i.e., with a sliding window of size $M = 2$. Different from uncoded ARQ, HARQ with soft combining, and CF ARQ, the transmission scheme is adaptive, i.e. the transmission rate is adjusted based on the cumulative feedback for $M = 2$ MDS coded packets in the transmitted packet stream. The receiver needs both coded packets to reconstruct the transmitted packet stream, i.e., the DoFs required at the receiver is $N = 2$. We do not assume in-order packet delivery. Therefore, the transmitted packets in the bucket will be successfully decoded when both of the coded transmitted packets are successfully received and ACK'ed by the receiver. While the model can be extended to $M > 2$ using a recursion, the state space scales exponentially, and the analysis becomes prohibitively complicated without any additional insights. Therefore, it is left as future work.

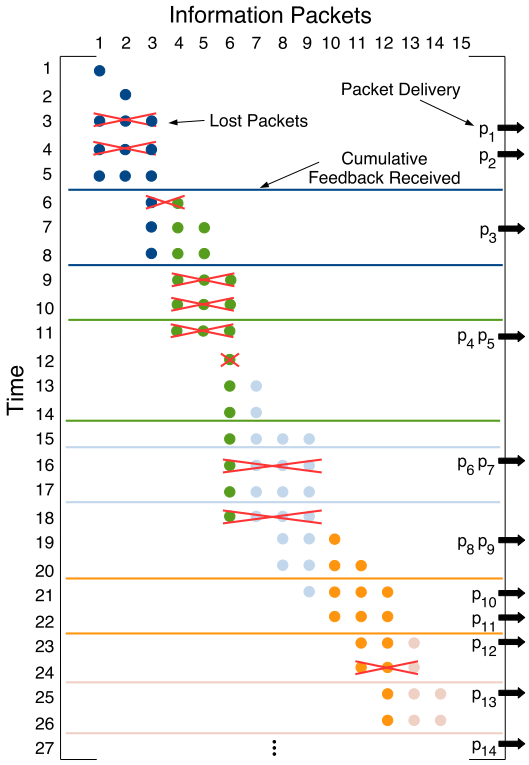

Fig. 2: Coding matrix for Coded ARQ scheme with 2 packets assuming a feedback delay of 2 time slots.

We illustrate the coding matrix for Coded ARQ for $M = 2$ in Fig. 2. The rows and columns of the matrix indicate the time and the specific information (uncoded) packets, p_i, to be transmitted. The dots in each row show the composition of the transmitted packet, and different colors indicate specific generations, horizontal lines show the time when feedback about a specific generation is obtained, and the red crosses show lost packets. Furthermore, the double arrows on the right-hand show the in-order delivery times of the packets to the client application. There are 2 main properties of Coded ARQ. Different from the uncoded ARQ, i) the ACK in Coded ARQ is cumulative, and ii) the rate of the Coded ARQ is adapted based on the cumulative feedback.

The combined observation set for Coded ARQ with $M = 2$ packets is $\mathcal{X}^{(c)} = \mathbb{Z}_2^3$. For example, $X_t^{(c)} = 001$ means that the forward channel is good for both packets and the reverse channel is erroneous, i.e., the ACK for $M = 2$ packets is lost. The HMM for the delay of Coded ARQ is shown in Fig. 3. Similar to previous models, I_1 and O are the input and output nodes, and other nodes are the hidden states, and I_1 and I_2 represent transmission of the first new packet and the second packet one time slot later, respectively. The possibilities upon

[5]Even for non-normally distributed variables, at least 88.8% of cases should fall within properly calculated three-sigma intervals, which follows from Chebyshev's Inequality. For unimodal distributions, the probability of being within the interval is at least 95% [25].

the transmission of the 2 packets are:

- **Transition to state A_2.** After sending the new packets ($M = 2$), the transmitter receives a feedback message $k-1$ time slots later. This state is denoted by node A_2.
- **Transition to state O.** If the feedback is an error-free ACK (with probability $\mathbf{P}_{000} = \mathbf{P}_{0x}\mathbf{P}_{00}$), then the system transits to state O.
- **Transition to state B_2.** If the feedback is a successful NACK for both packets (with probability $\mathbf{P}_{110} = \mathbf{P}_{1x}\mathbf{P}_{10}$), then the system transits to state B_2, where both packets have to be retransmitted.
- **Transition to state C_2.** If the feedback is an erroneous ACK (with probability $\mathbf{P}_{001} = \mathbf{P}_{0x}\mathbf{P}_{01}$) and the timer expires before receiving any error-free ACKs/NACKs, the system will transit to state C_2, the packets will be retransmitted, and the timeout will be reset. The packets will then be acknowledged when a succeeding ACK/NACK is correctly received.
- **Transition to state A_1.** If only one of the packets is successfully transmitted and the feedback is an error-free ACK (with probability $\mathbf{P}_{100} + \mathbf{P}_{010} = \mathbf{P}_{1x}\mathbf{P}_{00} + \mathbf{P}_{0x}\mathbf{P}_{10}$), the system goes to state A_1. This state is equivalent to the state A for the uncoded ARQ model as shown in [7, Figure 4]. Hence, the rest of the analysis follows from the uncoded ARQ analysis in [7].
- **Transition to state G_2.** Node G_2 indicates that a NACK is lost (with probability $\mathbf{P}_{111} = \mathbf{P}_{1x}\mathbf{P}_{11}$), both packets are lost, and the transmitter waits for timeout (node B_2). We refer the reader to Fig. 3 for the detailed description of these states. For the simplified matrix-flow diagrams for throughput and delay analysis of uncoded ARQ in unreliable feedback, see also [7, Figures 2, 4].
- **Transition to state G_3.** Node G_3 indicates that a NACK is lost, but only one of the packets is successfully transmitted and the other is lost (with probability $\mathbf{P}_{011} + \mathbf{P}_{101} = \mathbf{P}_{0x}\mathbf{P}_{11} + \mathbf{P}_{1x}\mathbf{P}_{01}$), and the sender waits for timeout (node B_3). Node A_3 denotes the retransmission of both packets, and the receiver only needs one of them. Hence, if the system goes to state A_3, the rest of the analysis follows from the uncoded ARQ analysis in [7].

In this paper, as we use tiny codes, i.e. sliding window by coding with just 2 packets, the available redundancy rate in terms of the packets in the encoding window is 50%. However, we do a finer-grained control over the redundancy rate via the feedback which is cumulative. This can be observed from Fig. 3. For example, if the CF acknowledges the successful reception of 1 packet only, i.e., the system transits to state A_1, then the rate is adaptively adjusted to retransmit 1 packet only. On the other hand, if only 1 packet is successfully transmitted and the CF is lost, the system has a transition to G_3, and then to A_3 that represents the retransmission of 2 packets while the receiver only needs one of the packets. In this case, upon the successful reception of the CF in the succeeding time slots, the system either transits to state A_1, i.e. 1 packet has to be retransmitted again, or to state 0, i.e. no retransmission is required. Therefore, the redundancy rate is not always 50%, and a finer-grained control is provided via the feedback.

The matrix gain of the graph in Fig. 3 can be calculated using the basic simplification rules. For general channel models (including the GE channel model), the analytical derivation of the MGFs of the transmission time (2) and delay time (3), hence characterization of the throughput and delay performance of Coded ARQ, have been provided in the full version of the paper [26, Appendices I, J]. Later in Sect. V, we demonstrate the throughput and delay performance for the GE channel under different burst erasure rates.

We next present closed form expressions for throughput and delay of memoryless channels.

Proposition 1. *The throughput for Coded ARQ for memoryless channels is given by*

$$\eta_{\mathsf{C\text{-}ARQ}} = \frac{(1-\epsilon)}{\alpha_C(\epsilon) + \epsilon^{d+1}(1-\epsilon)\beta_C(\epsilon)/(1-\epsilon^T)}, \quad (5)$$

where $\alpha_C(\epsilon) = (1+\epsilon+7\epsilon^2/2-\epsilon^3/2-3\epsilon^4+\epsilon^6)/(1+\epsilon)^2$, $\beta_C(\epsilon) = 1/2+\epsilon^2(1-\epsilon)$, *and* $d = T-k$.

Proof. See [26, Appendix I]. □

In (5), it is easy to show that $3/4 \leq \alpha_C(\epsilon) \leq 1$, and $1/2 \leq \beta_C(\epsilon) < 13/20$. Therefore, we can conclude that $\eta_{\mathsf{C\text{-}ARQ}}$ is always higher than η_{ARQ} for any given T, d, as determined in [7]. Furthermore, $\eta_{\mathsf{C\text{-}ARQ}}$ is upper bounded by $(1-\epsilon)/\alpha_C(\epsilon)$ as $T, d \to \infty$. This implies that for memoryless systems, with minimum coding, it is possible to achieve a gain of more than 30% compared with uncoded ARQ. The gain becomes higher if the channel has memory, as demonstrated in Sect. V.

Proposition 2. *The average delay of the Coded ARQ for memoryless channels is given by*

$$\begin{aligned}\overline{\mathrm{D}}_{\mathsf{C\text{-}ARQ}} &= k+1+3\epsilon+(2T+5k+4)\epsilon^2 \\ &\quad + (T-7k+5)\epsilon^3 + \mathcal{O}(\epsilon^4), \quad \epsilon \to 0. \end{aligned} \quad (6)$$

Proof. See [26, Appendix J]. □

The variance of delay for Coded ARQ is derived next.

Proposition 3. *The variance of delay for Coded ARQ for memoryless channels is*

$$\begin{aligned}\sigma^2_{\mathrm{D}_{\mathsf{C\text{-}ARQ}}} &= k^2\epsilon^2(1+\epsilon-16\epsilon^2) + k\epsilon^2(5-31\epsilon+43\epsilon^2 \\ &\quad - T\epsilon^2(4-6\epsilon+2\epsilon^2)) + \mathcal{O}(1), \quad \epsilon \to 0. \end{aligned} \quad (7)$$

Proof. See [26, Appendix K]. □

Comparing (6) with the average delay of uncoded ARQ, we observe that $\overline{\mathrm{D}}_{\mathsf{C\text{-}ARQ}} - \overline{\mathrm{D}}_{\mathsf{ARQ}} = 1 + (3-k-1)\epsilon + (T+5k+3)\epsilon^2 + \mathcal{O}(\epsilon^3)$ as $\epsilon \to 0$, which is due to the cumulative feedback. On the other hand, when ϵ is large, $\overline{\mathrm{D}}_{\mathsf{C\text{-}ARQ}}$ becomes comparable to $\overline{\mathrm{D}}_{\mathsf{ARQ}}$, as we demonstrate in Sect. V. However, Coded ARQ always provides better delay guarantees than uncoded ARQ. This provides insights in designing systems that are robust to the RTT fluctuations.

We next numerically evaluate the performance of the different ARQ protocols and outline the advantages of cumulative feedback and Coded ARQ over uncoded ARQ protocols.

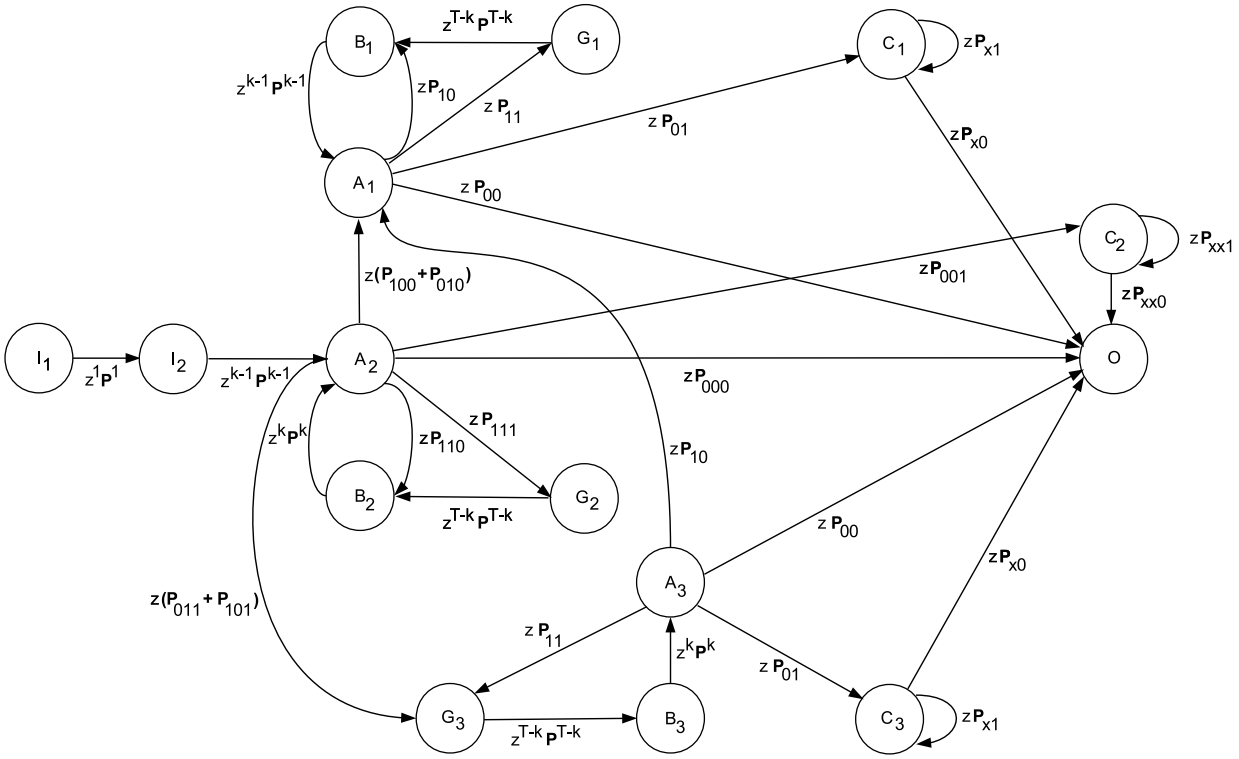

Fig. 3: Matrix-flow graph for delay analysis of SR ARQ in unreliable feedback with coding.

V. Numerical Simulations

We evaluate the performance of the Coded ARQ scheme outlined in Sect. IV by computing the MGFs of transmission and delay times via the MSFG approach detailed in Sect. III, and provide a numerical comparison of uncoded ARQ in [7] and Coded ARQ schemes with feedback erasures. We also include the simulation results[6] to validate our analytical models. The parameters for the numerical results are selected as follows. The RTT[7] is $k = 5$ time slots, timeout is $T = \{8, 15\}$ slots, and we have the same $r = \{0.1, 0.3\}$ for the forward and reverse GE channels with the same parameters $\epsilon_B = 1$ and $\epsilon_G = 0$, hence the same erasure rate. We also assume that $M = 2$ packets are transmitted, and $N = 2$ DoFs are required at the receiver. The performance metrics are the throughput η, the average per packet delay $\bar{D}$, i.e. the per packet delay for uncoded ARQ, and the delay corresponding to the transmission of $M = 2$ packets in Coded ARQ, and the guaranteeable delay $\hat{D}$ versus ϵ for varying timeout T and r. Unless otherwise specified, solid (Coded ARQ), and dotted (ARQ) curves denote the analytical results, and unfilled circles denote the simulation results.

[6]The source code for simulation and analysis is available at github.com/deryam/TinyCodesforDelayGuarantees.

[7]The slot duration should be adjusted according to the transmission protocol. For example, if the transmission rate is 10 Mbits/s, it takes 1ms to transmit 10^4 bits over the channel. In that case, the RTT equals to 1 ms.

The throughput of the baseline uncoded ARQ of [7], and the Coded ARQ protocols in the Markov channel for $r = 0.3$ is shown in Fig. 4, for $k = 5$ and $k = 10$, for different values of T. In the Coded ARQ, more packets can be reliably transmitted even when the packet loss rate ϵ is large. As ϵ increases, throughput of Coded ARQ scheme decays slower than the other schemes because coding can compensate the packet losses. Hence, less number of retransmissions is required. For Coded ARQ, the throughput is always higher than the throughput of the uncoded ARQ because the feedback is cumulative, which decreases the number of packets being transmitted per a successful packet. Furthermore, the transmission rate is adapted based on the feedback received.

We next illustrate the average delay $\bar{D}$ and the guaranteeable delay $\hat{D}$ with respect to erasure rate, ϵ, for $T = 8$, in Fig. 5, for $r = 0.3$ and $r = 0.1$, respectively. Since the minimum time required to transmit 2 packets is $k + 1$ given an RTT k, the delay gap between the coded and uncoded schemes at $\epsilon = 0$ is 1. Although the gap is indeed very small for small coding bucket sizes M, it also means that when the erasure rate is low, Coded ARQ has higher average delays compared to the uncoded ARQ. We observe that both $\bar{D}$ and $\hat{D}$ for both uncoded and Coded ARQ increase in T. Although Coded ARQ might have a higher $\bar{D}$ than uncoded ARQ, its $\hat{D}$ is lower than the value of uncoded ARQ. By increasing the timeout, the gap between the guaranteeable delays for both models can

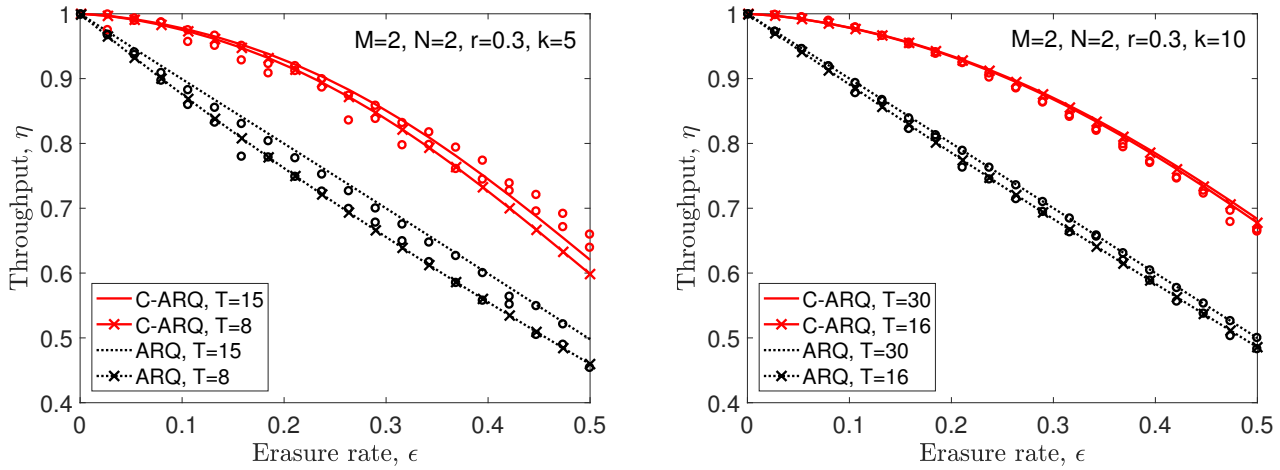

Fig. 4: Throughput η vs. erasure rate ϵ, in Markov erasures for burst parameter $r = 0.3$, and RTT $k = 5$ (L), RTT $k = 10$ (R).

be made smaller. Comparing the different models in Fig. 5, we observe that under burst errors ($r = 0.1$), the gap between the guaranteeable delays is larger compared to the case of $r = 0.3$ for small ϵ. When $r = 0.3$, the gap between the guaranteeable delays increases in ϵ, which means that Coded ARQ is more predictable under high erasure rates.

We can observe that the higher the error burst ($r = 0.1$), the lower the uncoded ARQ throughput and the higher is the delay in noisy feedback [7]. As the burst rate increases, the average delay is also higher for Coded ARQ. For both uncoded ARQ and Coded ARQ, when the timeout T increases, both throughput and delay are higher. For uncoded ARQ, the sensitivity of throughput to timeout T increases as r decreases, hence the throughput becomes very low when the timeout T is very small. When $r = 0.1$, for Coded ARQ, throughput becomes more sensitive to T, and it is possible to achieve significantly higher throughputs by increasing T. However, for both schemes, the sensitivity of delay to timeout T decreases as r decreases, hence the variability of delay with timeout T becomes less important under burst errors.

Our findings on Coded ARQ scheme suggest that the following design insights should enable more robust design for two-way erasure channels for wireless networks:

- Sensitivity of throughput and delay to timeout and RTT increases under burst errors.
- Uncoded ARQ is very sensitive to erasure bursts. Hence, the higher the burst rate, the lower its throughput and the higher its guaranteeable delay is.
- Coded ARQ can provide gains up to 37% in terms of throughput than the baseline uncoded ARQ.
- Coded ARQ has high delay but low variability. Furthermore, it has lower guaranteeable delay than uncoded ARQ given the unreliability of the feedback.
- Coding has benefits under imperfect feedback, burst erasures or higher erasure rates. Coded ARQ is more predictable across statistics, and hence is more stable. This can help design robust systems when RTT is unreliable.

VI. Conclusions

We leveraged discrete-time queuing and coding theory to enhance the performance of SR ARQ schemes by exploiting a matrix signal-flow graph approach. We proposed a Coded ARQ scheme and computed the MGFs of transmission and delay times. Contrasting the performance of Coded ARQ with the uncoded ARQ scheme, we demonstrated its gain in terms of throughput and delay. For the given parameter setting with a sliding window of size 2, Coded ARQ can provide gains up to 37% in terms of throughput, and its guaranteeable delay is lower than the one for uncoded ARQ.

Extensions include the optimization of the erasure coded schemes with minimal encoding and decoding complexity and their code rate, and the development of more sophisticated coding schemes, such as sequential MDS and convolutional codes, Reed-Solomon codes, and the study of convolutional codes for better FEC. This will pave the way for protocol design and state space representations. Possible future directions also include the extension of the minimum coding scheme to long codes. Extending Coded ARQ to bucket sizes to $M > 2$, we can investigate the scaling between the bucket size M, the RTT k and the DoFs required at the receiver N.

Appendix

A. Transition Probability Matrices

The state-transition matrix both for the forward and $\mathbf{P}$ for the reverse channels is denoted by $\mathbf{P}$. Since $\mathbf{P}$ is a stochastic matrix, $\mathbf{P}^n\mathbf{1} = \mathbf{1}$ for $n \geq 1$. The stationary vector of the state-transition matrix π satisfies $\pi\mathbf{1} = 1$ and $\pi\mathbf{P} = \pi$. The combined state-transition matrix for the symmetric GE channels equals the Kronecker product of $\mathbf{P}$ with itself, i.e., $\mathbf{P}^{(c)} = \mathbf{P} \otimes \mathbf{P}$, which is given by

$$\mathbf{P}^{(c)} = \begin{bmatrix} (1-q)^2 & q(1-q) & q(1-q) & q^2 \\ r(1-q) & (1-q)(1-r) & qr & q(1-r) \\ r(1-q) & qr & (1-q)(1-r) & q(1-r) \\ r^2 & r(1-r) & r(1-r) & (1-r)^2 \end{bmatrix}.$$

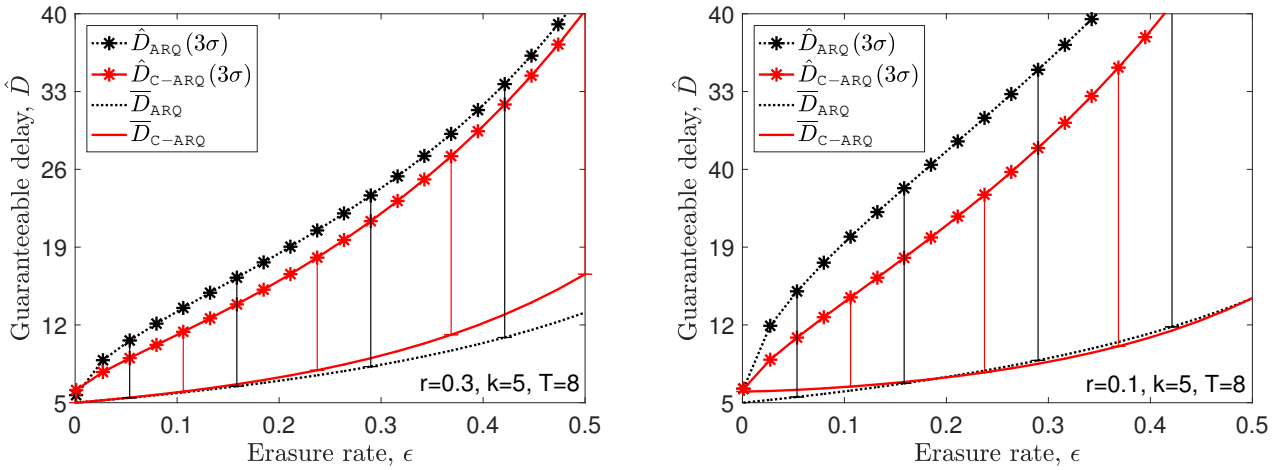

Fig. 5: Guaranteeable delay $\hat{D}$ vs. erasure rate ϵ, in Markov erasures for RTT $k=5$, timeout $T=8$, burst parameter $r=0.3$ (L), $r=0.1$ (R).

Let $\mathbf{P}_0$ and $\mathbf{P}_1$, respectively, be the success and the error probability matrices of an HMM. Forward and reverse links satisfy $\mathbf{P}_0^{(f)} = \mathbf{P}_0^{(r)} = \mathbf{P}_0$ and $\mathbf{P}_1^{(f)} = \mathbf{P}_1^{(r)} = \mathbf{P}_1$, where

$$\mathbf{P}_0 = \mathbf{P} \cdot \text{diag}\{\mathbf{1} - \boldsymbol{\epsilon}\} = \begin{bmatrix} 1-q & q \\ r & 1-r \end{bmatrix} \begin{bmatrix} 1-\epsilon_G & 0 \\ 0 & 1-\epsilon_B \end{bmatrix}$$
$$= \begin{bmatrix} (1-q)(1-\epsilon_G) & q(1-\epsilon_B) \\ r(1-\epsilon_G) & (1-r)(1-\epsilon_B) \end{bmatrix},$$
$$\mathbf{P}_1 = \mathbf{P} \cdot \text{diag}\{\boldsymbol{\epsilon}\} = \begin{bmatrix} 1-q & q \\ r & 1-r \end{bmatrix} \begin{bmatrix} \epsilon_G & 0 \\ 0 & \epsilon_B \end{bmatrix}$$
$$= \begin{bmatrix} (1-q)\epsilon_G & q\epsilon_B \\ r\epsilon_G & (1-r)\epsilon_B \end{bmatrix}.$$

The probability vector of transmitting a new packet is $\pi_I = \pi\mathbf{P}_0$. Given the erasure rates $\boldsymbol{\epsilon} = [\epsilon_G, \epsilon_B]$, and $\epsilon = \pi\boldsymbol{\epsilon}^\intercal$, we have $\pi_I\mathbf{1} = \pi\mathbf{P}_0\mathbf{1} = 1 - \pi\mathbf{P}\boldsymbol{\epsilon}^\intercal = 1 - \epsilon$, and $(\pi - \pi_I)\mathbf{1} = \pi\mathbf{P}_1\mathbf{1} = \pi\mathbf{P}\boldsymbol{\epsilon}^\intercal = \epsilon$. The combined observation probabilities are given by the following 4×4 matrices: $\mathbf{P}_{00}^{(c)} = \mathbf{P}_0 \otimes \mathbf{P}_0$, $\mathbf{P}_{01}^{(c)} = \mathbf{P}_0 \otimes \mathbf{P}_1$, $\mathbf{P}_{10}^{(c)} = \mathbf{P}_1 \otimes \mathbf{P}_0$, and $\mathbf{P}_{11}^{(c)} = \mathbf{P}_1 \otimes \mathbf{P}_1$.

Transition Probability Matrices for the Symmetric Memoryless Channel. Since the memoryless channel has only one state, $\mathbf{P} = 1$ and its combined state-transition matrix is $\mathbf{P}^{(c)} = \mathbf{P} \otimes \mathbf{P} = 1$. Hence, for memoryless channels with a symmetric erasure rate ϵ, we have $\mathbf{P}_0 = (1-\epsilon)$, and $\mathbf{P}_1 = \epsilon$. Thus, the observation probabilities are $\mathbf{P}_{00} = (1-\epsilon)^2$, $\mathbf{P}_{01} = (1-\epsilon)\epsilon$, $\mathbf{P}_{10} = \epsilon(1-\epsilon)$, and $\mathbf{P}_{11} = \epsilon^2$.

References

[1] 3GPP, "Service requirements for the 5G system," 3rd Generation Partnership Project (3GPP), TS 22.261, Jun. 2017.

[2] S. R. Khosravirad and H. Viswanathan, "Analysis of feedback error in Automatic Repeat reQuest," *arXiv preprint:1710.00649*, Oct. 2017.

[3] K. R. Sachin, S. Nikhil, S. K. Apeksha, C. Nisarga, and M. S. Usha, "A review of Hybrid ARQ in 4G LTE," *Int. J. Adv. Res. Innov. Ideas Educ.*, vol. 1, no. 3, pp. 160–165, 2015.

[4] J. Lu, C. K. Wu, S. Xiao, and J. C. Du, "A network coding based hybrid ARQ algorithm for wireless video broadcast," *Science China Information Sciences*, vol. 54, no. 6, pp. 1327–1332, Jun. 2011.

[5] "3GPP TS 36.212; evolved universal terrestrial radio access (E-UTRA); multiplexing and channel coding," Tech. Rep., Dec. 2016.

[6] "3GPP TR 36.877; LTE device to device (D2D) proximity services (ProSe)," 3GPP, Tech. Rep., Mar. 2015.

[7] K. Ausavapattanakun and A. Nosratinia, "Analysis of selective-repeat ARQ via matrix signal-flow graphs," *IEEE Trans. Commun.*, vol. 55, no. 1, pp. 198–204, Jan. 2007.

[8] Y. J. Cho and C. K. Un, "Performance analysis of ARQ error controls under Markovian block error pattern," *IEEE Trans. Commun.*, vol. 42, no. 2-4, pp. 2051–2061, Feb. - Apr. 1994.

[9] W.-K. Chen, "The use of matrix signal-flow graph in state-space formulation of feedback theory," in *Proc., IEEE Int. Symp. Circuits and Systems*, Jun. 1991, pp. 1005–1008.

[10] M. Karzand and D. J. Leith, "Low delay random linear coding over a stream," in *Proc., IEEE Annu. Allerton Conf. Commun., Control, and Computing*, Sep. 2014, pp. 521–528.

[11] J. Lieb, "Complete MDP convolutional codes," *arXiv preprint arXiv:1712.08767*, Dec. 2017.

[12] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann, "Practical loss-resilient codes," in *Proc., ACM Symp. Theory of Comput.*, New York, NY, USA, 1997, pp. 150–159.

[13] M. Tömösközi, F. H. Fitzek, D. E. Lucani, M. V. Pedersen, and P. Seeling, "On the delay characteristics for point-to-point links using random linear network coding with on-the-fly coding capabilities," in *Proc., European Wireless*, May 2014.

[14] G. Joshi, "On playback delay in streaming communication," Master's thesis, Cambridge, MA, USA, Jun. 2012.

[15] E. Martinian, "Dynamic information and constraints in source and channel coding," Ph.D. dissertation, Cambridge, MA, USA, Sep. 2004.

[16] E. O. Elliott, "Estimates of error rates for codes on burst-noise channels," *Bell System Technical Journal*, vol. 42, no. 5, pp. 1977–1997, Sep. 1963.

[17] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state markov chains," *The Annals of Mathematical Statistics*, vol. 37, no. 6, pp. 1554–1563, Dec. 1966.

[18] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, and J. Crowcroft, "Forward error correction (FEC) building block," Tech. Rep. RFC 3452, 2002.

[19] D. Malak, M. Médard, and E. Yeh, "ARQ with cumulative feedback to compensate for burst errors," in *Proc., IEEE Globecom*, Dec. 2018.

[20] J. K. Sundararajan, D. Shah, and M. Médard, "ARQ for network coding," in *Proc., IEEE Intl. Symp. Inf. Theory*, Jul. 2008.

[21] S. J. Mason and H. J. Zimmermann, *Electronic, Circuits, Signals, and Systems*. New York: Wiley, 1960.

[22] R. A. Howard, *Dynamic Probabilistic Systems*. Courier Corp., 1971.

[23] D. L. Lu and J. F. Chang, "Analysis of ARQ protocols via signal flow graphs," *IEEE Trans. Commun.*, vol. 37, no. 3, pp. 245–51, Mar. 1989.

[24] ——, "Performance of ARQ protocols in nonindependent channel errors," *IEEE Trans. Commun.*, vol. 41, pp. 721–30, May 1993.

[25] F. Pukelsheim, "The three sigma rule," *American Statistician*, vol. 48, p. 8891, 1994.

[26] D. Malak, M. Médard, and E. M. Yeh, "Tiny codes for guaranteeable delay," *IEEE J. Sel. Areas Commun.*, vol. 37, pp. 809–825, Apr. 2019.

[27] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, pp. 4413–30, Oct. 2006.