IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 3, 2018, accepted August 9, 2018, date of publication August 13, 2018, date of current version September 5, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2865137

Caterpillar RLNC With Feedback (CRLNC-FB): Reducing Delay in Selective Repeat ARQ Through Coding

FRANK GABRIEL[1], SIMON WUNDERLICH[1], SREEKRISHNA PANDI[1], FRANK H. P. FITZEK[1], AND MARTIN REISSLEIN[2], (Fellow, IEEE)

[1]Deutsche Telekom Chair of Communication Networks, 5G Lab Germany, Technical University Dresden, 01062 Dresden, Germany
[2]School of Electrical, Computer, and Energy Engineering, Arizona State University, Tempe, AZ 85287-5706, USA

Corresponding author: Martin Reisslein (reisslein@asu.edu)

This work was supported in part by Ericsson, in part by Deutsche Telekom, and in part by the Free State of Saxony through the European Commission for the Atto3D Project.

ABSTRACT Wireless networks typically employ some form of forward error correction (FEC) coding and some automatic repeat request (ARQ) protocol to ensure reliable data transmission over lossy channels. We propose to integrate FEC and ARQ in the context of random linear network coding (RLNC). In particular, we develop Caterpillar RLNC with feedback (CRLNC-FB), an RLNC approach with a finite sliding packet transmission window in conjunction with feedback-based selective repeat ARQ. CRLNC-FB employs a novel RLNC decoding method based on a band-form of Gaussian elimination. In response to lost packets, CRLNC-FB retransmits lost packets in systematic (uncoded) form to aid fast in-order packet delivery at the receiver. Extensive performance evaluations indicate that CRLNC-FB gives higher throughput-delay performance than the preceding RLNC approaches with feedback. In particular, CRLNC-FB with its sliding window achieves lower delays than block-based RLNC. Also, the retransmission of uncoded source packets in CRLNC-FB contributes to a significantly higher throughput-delay performance than loss recovery through coded packets interspersed among future source packets at a prescribed code rate.

INDEX TERMS Automatic repeat request (ARQ) protocol, random linear network coding (RLNC), reliable data transfer, packet delay, throughput-delay tradeoff.

I. INTRODUCTION

A. MOTIVATION

Increasing transmission bitrates and coverage distances in modern wireless systems lead to large bandwidth-delay products. On the other hand, the burst error characteristics of wireless systems cause losses of several successive packets that need to be recovered for applications requiring reliable data transfer. Reliable data transfer is conventionally achieved through combinations of automatic repeat request (ARQ) protocols and forward error correction (FEC) coding. Feedback based ARQ protocols typically either achieve only low throughput or incur relatively high delays in networks with large bandwidth-delay products. FEC can mitigate delays by recovering lost packets without retransmissions. In particular, FEC through random linear network coding (RLNC) has many attractive features for complex wireless networks [1]–[11].

The first practical RLNC approaches were based on blocks (generations) of multiple successive source packets [12], [13]. Block based RLNC incurs relatively high delays as the encoding at the sender and the loss recovery (decoding) at the receiver operate on the basis of multiple source packets. Recent sliding window RLNC approaches advance the encoding window at the sender and the decoding window at the receiver by individual source packets and thus have the potential to reduce delays compared to block based RLNC [14], [15]. As detailed in Section II, initial sliding window RLNC studies considered an infinite sliding window that covered all packets of a stream. The infinite sliding window is impractical as it becomes computationally prohibitive.

Practical finite sliding window RLNC has received relatively little research attention so far, and we are aware of only one prior approach, namely Tetrys [16]–[19], that examined finite sliding window RLNC with feedback in the context of burst error wireless networks.

B. CONTRIBUTION

We develop and evaluate a novel sliding window RLNC approach that exploits feedback based ARQ to overcome bursty packet losses. In particular, we introduce Caterpillar RLNC with Feedback (CRLNC-FB). CRLNC-FB combines RLNC over a finite sliding window with feedback based retransmission of lost source packets to quickly recover from bursts of lost packets. CRLNC-FB employs a novel form of RLNC decoding with a band-form matrix Gaussian elimination. CRLNC-FB quickly recovers from loss bursts through retransmissions of lost source packets in uncoded (systematic) form. We conduct extensive evaluations of CRLNC-FB for burst-error wireless channels with large bandwidth-delay products. We find that CRLNC-FB provides favorable throughput-delay performance for a wide range of channel conditions and can be flexibly tuned to emphasize low delay or high throughput by setting the RLNC code rate.

II. BACKGROUND AND RELATED WORK

A. RLNC OVERVIEW

RLNC creates coded packets by linearly combining payload source packets (also referred to as source symbols) over a Galois field GF(2^q). Block based RLNC operates on non-overlapping sets of W, $W \geq 1$, successive source symbols. That is, the encoding window encompasses W successive source symbols and advances in steps of W source symbols. The systematic variant of block based RLNC sends the W source symbols of a block in uncoded form, i.e., systematically [20], followed by $W(1/R - 1)$ coded packets, whereby R, $R \leq 1$, denotes the code rate, i.e., the ratio of the number of source packets to the total number of transmitted packets (systematic + coded packets).

In contrast, sliding window RLNC advances the encoding window in steps of individual source symbols [14], [21]–[24]. Sliding window RLNC with systematic source symbol transmission enforces a prescribed code rate R by sending $S = R/(1-R)$ successive systematic source packets followed by one coded packet. Feedback free RLNC has been studied in several contexts, see e.g., [25]–[31].

B. FEEDBACK BASED AUTOMATIC REPEAT REQUEST (ARQ)

A classic ARQ scheme is the stop-and-wait ARQ protocol, also referred to as send-and-wait protocol, which was originally employed in IEEE 802.11 W-LANs [32]. The sender sends a single packet and then stops to wait for a feedback acknowledgement from the receiver. The receiver sends a feedback upon every received packet. The sender only transmits another packet after receiving an acknowledgement for the previous packet or after a timeout. Stop-and-wait ARQ is simple and performs very well if the propagation delay between sender and receiver is negligible compared to the packet transmission time. However, increasing wireless network coverage distances increase the propagation time and improvements in modulation (e.g., OFDMA and MIMO) decrease the packet transmission time, increasing the bandwidth-delay product of wireless networks. Stop-and-wait ARQ achieves only low utilization, i.e., low throughput, in networks with a large bandwidth-delay product.

Modern wireless protocols therefore implement window based ARQ schemes. They allow a window of W, $W \geq 1$, packets to be sent without waiting after each transmission. With the selective repeat form of window based ARQ, the receiver acknowledges individual received packets or selectively requests retransmissions of lost packets. Selective repeat ARQ can achieve high utilization, i.e., high throughput, if the transmission time of the W packets is larger than the round-trip (propagation delay) time (RTT). Selective repeat ARQ is implemented in IEEE 802.11 W-LANs since the 802.11n amendment [32] as well as many cellular networks, such as WCDMA (also known as UMTS) [33].

In order to further enhance the wireless network reliability, some wireless networks combine FEC and ARQ. For instance, Long Term Evolution (LTE) cellular networks use a combination of selective repeat ARQ on an upper layer and a lower layer Hybrid ARQ scheme (H-ARQ) [34]–[36]. Hybrid ARQ schemes specifically address bit errors in received packets by resending (partial) packet data using FEC codes. In contrast, we consider erasure channels, where complete packets either arrive without bit errors or complete packets are dropped. Also, hybrid ARQ schemes typically do not combine multiple packets as we do through network coding in CRLNC-FB-FB.

C. FEEDBACK ENHANCED RLNC

The enhancement of network coding with feedback has been considered in relatively few studies to date [37]. The general feedback structure for supporting RLNC has been studied in [38]. General principles for enhancing multicast and broadcast network coded transmissions with feedback have been examined in [39]–[48]. Block based RLNC has been enhanced with feedback in [49] to trigger the transmission of additional coded packets for a generation in order to recover from packet losses.

Sliding window RLNC studies have commonly exploited receiver feedback to advance the lower end of the encoding window, which covers the oldest source packets [50]–[56]. Thus, these studies essentially utilize the feedback to turn infinite sliding window RLNC [14], [21]–[24], which considers the complete set of source symbols since the beginning of a stream, into finite sliding window RLNC that considers a subset of the stream's source symbols. Prior finite sliding window RLNC studies have mainly focused on the interactions with the Transmission Control

Protocol (TCP) [50], [51], [55], on maximizing the throughput [52], or on information theoretic characterizations [53], [54], [56]. In [57] the feedback is exploited to control the transmission rate at the sender in order to avoid overwhelming the channel.

More closely related to our study emphasis on delay reduction, the Tetrys protocol [16]–[19] exploits feedback to reduce delay for reliable data transfer by selectively removing source symbols that have been "seen" at the receiver from the encoding window at the sender (whereby the definition of "seen" follows from [54], [56]). Importantly, the sliding window approaches that advance the lower end of the encoding window based on feedback cover a set of consecutive source symbols within the encoding window; in contrast, the Tetrys encoding window covers only the "unseen" source symbols, which may be non-consecutive. However, Tetrys does not retransmit lost packets or transmit additional coded packets in response to losses. Instead, Tetrys continues sending new systematic source symbols and coded packets at the prescribed code rate R, whereby the coded packets are linear combinations of only the "unseen" packets. In contrast to Tetrys, our proposed CRLNC-FB retransmits lost systematic source packets that cannot be recovered from the coded packets so as to actively address bursts of lost packets.

The recent analysis in [58] has modeled the decision problem of sending a systematic source packet or a coded packet for a packet erasure channel with independent packet losses in each slot with a fixed loss probability and with a lossless feedback channel. In contrast, we consider a more realistic burst error channel where successive slots are correlated and feedback may be lost.

D. BANDWIDTH-DELAY CHARACTERISTICS IN WIRELESS SYSTEMS

ARQ protocols in modern wireless systems have to accommodate increasing bandwidth-delay products mainly due to increased transmission bitrates. While early 802.11b WiFi could work well with stop-and-wait ARQ due to low transmission bitrates and short propagation distances between stations, today's cellular and satellite systems have large bandwidth-delay products. RLNC is generally well suited as a basis for reliable data transfer protocols in these challenging wireless communication settings [59]–[64].

LTE network measurements in 2012 [65] found approximately 5 ms one-way downlink delay and 15 ms one-way uplink delay on the radio access layer, which incorporates selective repeat ARQ and hybrid ARQ [66]. These delay values are also seen at the 5th percentile; thus, these delay values are reasonable approximations for the "good" cases where the ARQ does not add latency. These measurements were conducted at a distance of 130 m from the base station, corresponding to a negligible propagation delay of less than a microsecond. 5G cellular networks will likely have similar characteristics. Considering the 1 Gbit/s user peak transmission bitrate for high mobility [67], a 1 ms RTT according to the tactile Internet delay target [68], and a 1500 byte MTU, the round-trip bandwidth delay product is about 90 packets.

Broadband communication satellites operate in different orbits. The popular Inmarsat satellite fleet is positioned in the geostationary equatorial orbit (GEO) of 35,786 km above the Earth's equator. The one way latency is mostly governed by the speed of light propagation delay, which contributes at least 120 ms delay. Inmarsat typically provides a maximum throughput of 25 Mbit/s to end users. Thus, there can be up to 262 packets in flight in a direction at a time.

OneWeb [69] and SpaceX [70] plan to deploy low earth orbit (LEO) satellite constellations [71] at an altitude of 1,200 km, resulting in RTTs of approximately 8 ms. Other delay components will likely contribute larger delays, but we consider the propagation delays as lower delay bounds. SpaceX advertises up to 1 Gbps per user, and 17–23 Gbps per satellite. OneWeb targets 6 Gbps per satellite. Conservatively considering 300 Mbps results in a lower bound for the bandwidth-delay product around 210 packets.

III. CATERPILLAR RLNC WITH FEEDBACK (CRLNC-FB) PROTOCOL

This section introduces the CRLNC-FB protocol. First, the encoder operation and packet format are introduced, followed by the operation of the decoder and the feedback format. The main notations are summarized in Table 1.

TABLE 1. Summary of main notations.

CRLNC-FB Encoding Parameters	
s_e	Source symbol sequence number at encoder
p_e	Packet count tracks number of transmitted (systematic and coded) packets at encoder
S	Number of consecutive uncoded (systematic) source symbols preceding a coded symbol
W	Window size [in number of source symbols] for ARQ and RLNC
R	Code rate; $R = S/(S+1)$
T	Retransmission limit, $T = 0$ means no retransm.
CRLNC-FB Decoding Parameters	
s_d	Highest source symbol sequence number received at decoder, irrespective of whether decoded or not
p_d	Highest packet count received at decoder, irrespective of whether decoded or not

A. ENCODING AND PACKET FORMAT

The CRLNC-FB protocol employs the CRLNC encoding mechanism and extends the CRLNC packet format to support the decoding with feedback. We briefly review the CRLNC encoding and packet format [30] and describe the modifications due to the operation with feedback. CRLNC encoding follows the general principles of systematic RLNC with a finite sliding window covering W successive source symbols. In particular, S, $1 \leq S \leq W$, successive source symbols are sent uncoded (systematically), followed by one coded packet so as to achieve a code rate of $R = S/(S+1)$. (CRLNC-FB with $R = 1$ corresponds to conventional selective repeat ARQ without network coding.) Each source symbol is sent only once, unless a retransmission is requested through feedback. A coded packet is constructed with RLNC considering all

unacknowledged source symbols within the window. Following the principles of conventional window based reliable data transfer protocols, the encoder can send at most W source symbols. Suppose that s_e^b denotes the highest sequence number up to which all source symbols have been acknowledged, i.e., the receiver has acknowledged all preceding source symbols up to and including the source symbol with sequence number s_e^b to the encoder. Then, we refer to the contiguous range of source symbol sequence numbers from $s_e^b + 1$ to $s_e^b + W$ as the encoding window. The removal of the oldest source symbol (with lowest sequence number s_e^b) from the window is commonly referred to as *closing* the window on source symbol s_e^b.

The CRLNC packet format consisting of source packet sequence number (source symbol sequence number s_e for systematic source packet; highest considered source symbol sequence number s_e for coded packet), the coding coefficients, and the payload (systematic source packet; or coded packet) [30] is augmented by a packet count p_e. The packet count p_e is incremented for each sent packet, i.e., for each sent systematic source packet and for each sent coded packet. This packet count is considered when interpreting the feedback, see Section III-C. Following the principles of window-based network transport protocols [72]–[74], the packet count needs to uniquely identify the packet transmissions. For each source packet s_e in the encoding window, the encoder memorizes the corresponding packet count $p_e(s_e)$ of the packet transmission that contained source packet s_e. For instance, in the example in Fig. 1, the source symbol $s_e = 7$ is contained in the packet transmission with packet count $p_e(s_e = 7) = 10$.

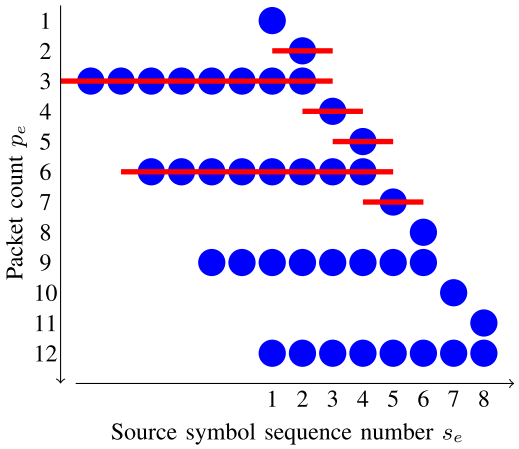


FIGURE 1. Example of a packet stream with sliding window network coding with window size $W = 8$ and code rate $R = 2/3$, i.e., a coded packet after every $S = 2$ systematic source packets.

B. DECODING

1) PRINCIPLES FOR RLNC DECODING WITH FEEDBACK

The CRLNC-FB decoder performs a novel variant of Gaussian elimination to recover the original source symbols from the coded symbols. Additionally, the decoder sends feedback to the encoder. The feedback indicates the missing packets at the decoder to give the encoder an opportunity to retransmit them. Before introducing the details of the CRLNC-FB decoding, we briefly discuss basic considerations for sliding window RLNC decoding in networks with feedback.

a: DECODING WINDOW AND DECODING MATRIX

As in conventional finite sliding window RLNC [12], [75], [76], the decoder maintains a decoding matrix with W rows corresponding to W successive source symbols and W columns corresponding to the coding coefficients related to the source symbols. We refer to the range of source symbol sequence numbers covered by the rows of the decoding matrix as the decoding window. The decoding window at the receiver is advanced by the arrival of a packet with a higher source symbol sequence number s_e than the current highest source symbol sequence number s_d at the decoder. Specifically, upon the arrival of a packet with source symbol sequence number $s_e > s_d$, the decoding window is advanced to cover the source symbol sequence numbers $s_e - (W - 1), \ldots, s_e - 1, s_e$, moreover s_d is updated to s_e, and the decoding window is closed on any source symbols with sequence numbers $s_e - W$ and lower.

b: SYSTEMATIC (UNCODED) SOURCE SYMBOLS TO SUPPORT LOW DELAY

For low-latency communication, we want to move the encoding window forward, i.e., slide the encoding window up to the next source symbols as early as possible. For this reason, we adopted systematic RLNC for CRLNC-FB, i.e., we send source symbols in their original form (uncoded) and additionally send coded packets to aid recovery. With systematic RLNC, the decoder (receiver) can immediately pass a received uncoded source symbol to the next higher protocol layer, if all preceding symbols have been decoded (i.e., the in-order source symbol delivery is ensured). Thus, the decoder does not need to wait until the decoding matrix reaches full rank, if the preceding source symbols have been decoded [77]–[81]. However, a lost symbol (in conjunction with the considered in-order delivery requirement) forces the decoder to wait until enough coded packets have arrived to recover all losses, or the symbol is considered permanently lost. A symbol is considered permanently lost when a timeout occurs, as detailed in Section III-B.1.d, e.g., when the encoder has reached the retransmission limit and moved on to a higher sequence number, thus evicting an old symbol from the decoding window.

The CRLNC-FB decoder reduces the wait times and occurrences of permanent losses by explicitly requesting the retransmission of lost packets (that cannot be recovered from the coded packets). The CRLNC-FB encoder retransmits requested systematic packets, while maintaining the RLNC code rate for the retransmissions.

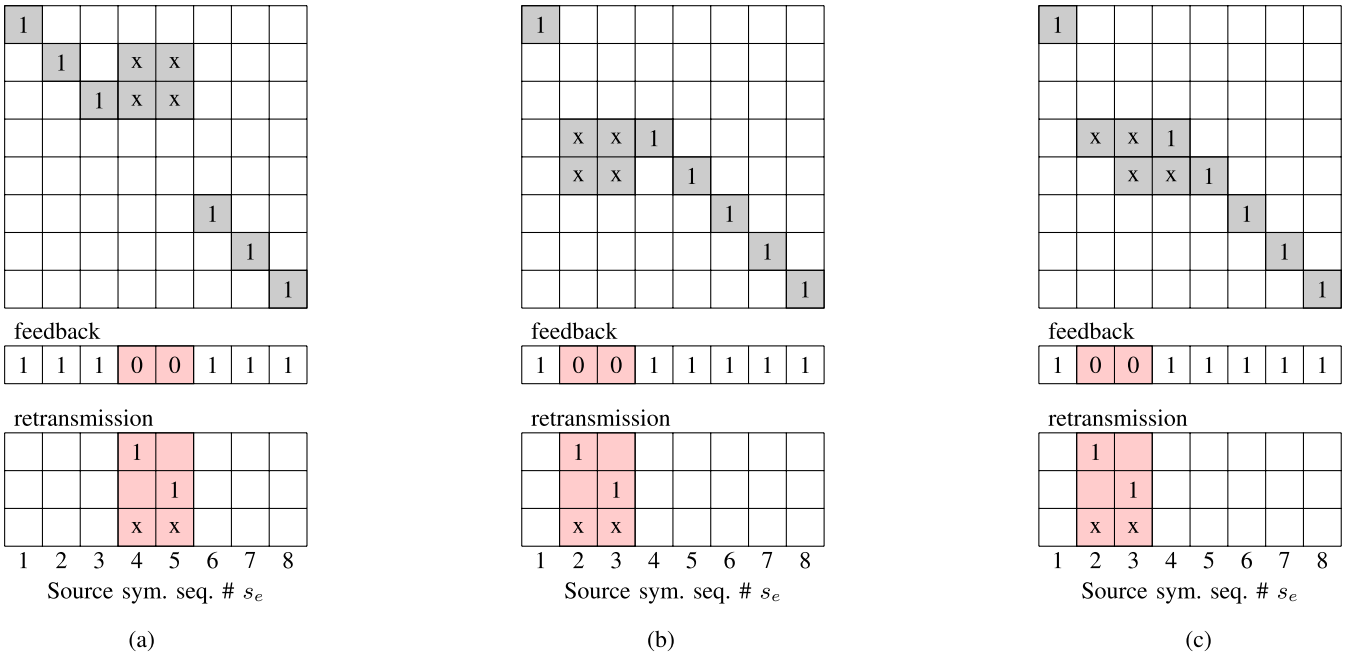


FIGURE 2. Illustration of three different forms of Gaussian elimination for the example stream in 1. The novel band matrix RLNC decoding performs right-to-left pivot selection and left-to-right elimination so that at most one row is removed when closing decoding window by one symbol. (a) Forward Gaussian elim. (b) Reverse Gaussian elim. (c) Band Matrix.

c: DEFINITIONS OF SOURCE SYMBOL TYPES FOR RLNC DECODING

Generally, in block (generation) based RLNC, the decoder recovers all symbols in a coded block when the decoding matrix achieves full rank. In RLNC protocols it therefore suffices to feed back the number of missing rows, without explicitly identifying the missing rows [49]. However, there are still good reasons to explicitly identify the missing rows in the feedback. More specifically, inspired by [54] and [56], we define two main types of source symbols at the decoder. A source symbol s_e has been "seen" at the decoder if the decoder has a pivot (i.e., a value of one) in the position for source symbol s_e in the decoding matrix. In particular, either the systematic source symbol s_e has been received at the decoder or the systematic source symbol s_e was dropped by the channel and successive coded packets were received and enabled the decoder to normalize the pivot of the decoding matrix row corresponding to source symbol s_e. In the example in Fig. 2(a), source symbols $s_e = 1, 2, 3, 6, 7$, and 8 have been seen.

We further define the complement of the seen source symbols as the "unseen" source symbols. The decoder does not have any entries for the unseen source symbols in the decoding matrix and requests the retransmission of the unseen source symbols in the feedback to the encoder. In the example in Fig. 2(a), source symbols $s_e = 4$ and 5 are unseen. Indicating the requested unseen source symbols in the feedback in turn allows the encoder to close the encoding window on the source symbols that have been seen at the decoder.

d: TIMEOUT

Timeout mechanisms can be employed in CRLNC-FB to mainly reduce delays at the expense of occasional permanent source symbol losses, i.e., by giving up on the delivery of a source symbol. A timeout can be implemented through a retransmission limit at the encoder through specifying the maximum allowed integer number of retransmissions. If the retransmission limit has been exhausted for a source symbol s_e, then the encoder closes the encoding window on the source symbol s_e, i.e., shifts up the encoding window to cover source symbols $s_e + 1$ to $s_e + W$, allowing the transmission of a new source symbol $s_e + W$ (which previously was outside the window). The arrival of the new source symbol $s_e + W$ will advance the decoding window (decoding matrix) to cover source symbols s_e+1 to s_e+W and any coefficients related to source symbol s_e will be removed from the decoding matrix.

Alternatively, a timeout could be implemented at the decoder. Specifically, the decoder can close the decoding window on source symbols that have exceeded a prescribed delay limit so as to meet latency requirements of the application. Upon such a timeout, the decoder sends a feedback message indicating that the affected source symbols have been seen (essentially pretending that they have arrived) so that the encoder can advance its window.

Moreover, following the principles of the transmission control protocol (TCP) [82], the encoder maintains a timer for the oldest unacknowledged source symbol in the window. This timeout addresses scenarios with long loss bursts on the encoder-to-decoder channel and/or the decoder-to-encoder channel, where the encoder may transmit many

packets and not receive any feedback packets. Without the receipt of feedback packets, the encoder may be blocked from further transmissions. This encoder blocking occurs if the window is full, but the oldest symbol is not acknowledged. To prevent encoder blocking, the CRLNC-FB encoder waits for feedback for the oldest unacknowledged source symbol for a prescribed timeout duration. The timeout duration may be set similar to the timeouts in reliable data transfer protocols [83], [84]. If this timeout expires, the encoder sends an additional coded packet that combines all presently unacknowledged packets.

2) BAND-FORM GAUSSIAN ELIMINATION

This section introduces a novel variant of the Gaussian elimination for RLNC decoding, namely band-form Gaussian elimination. Band-form Gaussian elimination facilitates the timely recovery of the source symbols with the aid of feedback.

Fig. 1 shows an example of a packet stream while Fig. 2(a) illustrates the corresponding decoding matrix. In the illustrated example, a packet stream with an encoding window of $W = 8$ and $S = 2$ systematic source symbols between successive coded packets, i.e., code rate $R = 2/3$, has been ongoing for a while without losses. We focus now on the excerpt indicated by packet count values $p_e = 1, 2, \ldots, 12$ (on the y-axis) and the corresponding source symbols $s_e = 1, 2, \ldots, 8$ (indicated by the x-axis position). In the example, the packets with packet counts $p_e = 2, 3, \ldots, 7$ are lost during network transport.

a: FORWARD GAUSSIAN ELIMINATION

The conventional forward Gaussian elimination for a matrix with incomplete rank results in a matrix in reduced row echelon form, as shown in Fig. 2(a), covering source symbols $s_e = 1, 2, \ldots, 8$. The conventional Gaussian elimination proceeds in a ''forward'' manner from the top left towards the bottom right when forming the reduced row echelon form. With this forward Gaussian elimination, the decoder must wait with decoding source symbol $s_e = 2$ until the matrix reaches full rank and can be fully decoded. In particular, the feedback acknowledges all seen source symbols, i.e., source symbols $s_e = 1, 2, 3, 6, 7$, and 8, as illustrated by the feedback message in the middle part of Fig. 2(a). More specifically, in the depicted example in Fig. 1, source symbols $s_e = 2, 3, 4$, and 5 were lost. However, the two received coded packets $p_e = 9$ and 12 included all these lost source symbols. These two coded packets are entered into the decoding matrix using Gaussian elimination, see Fig. 2(a). In particular, the coding coefficients of the already decoded source symbols $s_e = 1, 6, 7$, and 8 are eliminated. Then, conventional forward Gaussian elimination will normalize the remaining left-most coefficients. The elimination process first normalizes the left-most coefficient corresponding to the pivot for source packet $s_e = 2$, and then the left-most coefficient corresponding to the pivot for source packet $s_e = 3$.

As illustrated in the decoding matrix in Fig. 2(a), source symbols $s_e = 2$ and 3 have thus been ''seen'', but cannot be decoded yet since their packets contains data from other source symbols. That is, there are still non-zero (non-yet-eliminated) coding coefficients related to other source symbols in rows 2 and 3 of the decoding matrix. Source symbols $s_e = 4$ and 5 are unseen at the decoder and the decoder thus requests their retransmission. Note that source symbols $s_e = 4$ and 5 are contained in the received coded packets $p_e = 9$ and 12. However, these two coded packets have been ''used up'', to obtain the rows with pivots for source symbols $s_e = 2$ and 3.

Thus, the feedback message requests the retransmission of source symbols $s_e = 4$ and 5. (For forward Gaussian elimination, the feedback message also needs to indicate the highest source symbol sequence number s_d^e up to which all source symbols have been decoded without any gaps, whereby $s_d^e = 1$ in the example in Fig. 2(a). Upon receipt of the feedback at the encoder, the encoding window can be closed on this source symbol s_d^e and any preceding source symbols.) Once these source symbols $s_e = 4$ and 5 have been received, the matrix reaches full rank and source symbols $s_e = 2$ and 3 can be decoded (recovered) and delivered along with source symbols 4–8 to the higher layers. Alternatively, the matrix reaches full rank if the decoder receives one of these retransmitted systematic source symbols and the coded packet that is sent along with the retransmitted source symbols, adhering to the prescribed code rate.

b: REVERSE GAUSSIAN ELIMINATION

An alternative Gaussian elimination strategy is to proceed in a ''reverse'' manner from the bottom right towards the top left with forming the reduced row echelon form, as illustrated in Fig. 2(b). By selecting the pivots in the reverse manner, i.e., starting from the right side of the matrix, the selection of the seen packets changes slightly, as illustrated in Fig. 2(b).

However, decoding with this backward Gaussian elimination is problematic for CRLNC-FB in case of a timeout (Section III-B.1.d). When the retransmission fails after the specified timeout, then the sliding window will be closed, and the symbol whose retransmission failed is removed from the window. This symbol removal results in a permanent symbol loss, if the symbol has not been decoded prior to the removal. Moreover, all rows in the decoding matrix with a nonzero coefficient for this symbol are removed.

For example, for the forward Gaussian elimination in Fig. 2(a), closing the decoding window by one symbol will only remove the one corresponding row from the matrix. For instance, closing the decoding window on source symbol 2 in Fig. 2(a) would only remove row 2 from the decoding matrix. In contrast, for the backward Gaussian elimination in Fig. 2(b), multiple rows need to be removed if the symbol for the second column of the decoding matrix is removed. In particular, closing the receiver window on source symbol 2 in Fig. 2(b) implies the removal of all rows

that have a nonzero coefficient in column 2; thus, the rows corresponding to source symbols 4 and 5 would need to be removed. In the worst case, closing the receiver window by one symbol removes all coded symbols from the decoding matrix.

c: BAND MATRIX GAUSSIAN ELIMINATION

In order to exploit the strengths, yet to avoid the shortcomings of the forward and backward Gaussian elimination for RLNC decoding, we developed the novel band-matrix Gaussian elimination for CRLNC-FB decoding. The band-matrix Gaussian elimination illustrated in Fig. 2(c) brings the matrix in a band-like form, and we refer to the approach therefore also as band-form Gaussian elimination. We select the pivots in reverse order, from bottom right to top left, but we do not bring the matrix into row-echelon form. Instead, after a pivot is selected in the right to left order, the elimination process proceeds from left to right. This right-to-left pivot selection and left-to-right elimination proceeds in an iterative manner processing one received packet at time. The resulting matrix has the same pivots as reverse Gaussian elimination, as observed when comparing Fig. 2(c) with Fig. 2(b). However, the following left-to-right elimination additionally ensures that for every source symbol the matrix contains only one row where the corresponding coefficient is the left-most nonzero coefficient. For a matrix with this property, at most one row needs to be removed when the decoding window is closed by one symbol. This preserves as much information as possible in the decoding matrix.

C. FEEDBACK FORMAT AND FEEDBACK PROCESSING AT ENCODER

1) BITMASK FEEDBACK FORMAT

The decoder sends a feedback packet in response to each received packet (or after a receiver timeout, see Section III-B.1.d). A feedback packet contains the highest received packet count at the decoder p_d, the highest received source symbol sequence number of the decoder s_d, and a bit mask. The bit mask marks all packets that have been seen by the decoder (irrespective of whether they have been decoded or not), as illustrated in Fig. 2.

In case of reordering of feedback messages during network transport, the encoder can identify the most recent feedback information by the packet count in the feedback packet. In particular, the encoder ignores feedback packets with a lower packet count p_d than previously received feedback packets.

2) BASIC RETRANSMISSION POLICY

Upon reception of a feedback packet, all packets that are indicated by the bit mask as seen at the decoder are marked by the encoder as received. These received symbols will never be used again to generate a coded symbol. This decreases the computational complexity of encoding and decoding. If the oldest symbol in the window is acknowledged in this way, then the window can be closed on the oldest symbol and a new symbol can be added to the window. In other words, with the band matrix Gaussian elimination, the encoding window is closed according to the source symbols "seen" at the decoder; whereas with the conventional forward Gaussian elimination, the encoding window is closed according to the source symbols that have been decoded. With either approach, the encoding window is closed on the highest indexed source symbol (and all preceding symbols) up to which all source symbols have been "seen" (decoded) without any gaps in the band matrix (forward) Gaussian elimination.

All unacknowledged packets are scheduled for retransmission if the following retransmission criterion is met. A source symbol s_e that was previously transmitted with packet count $p_e(s_e)$ is retransmitted only if the packet count p_d of the decoder in the feedback packet is higher than $p_e(s_e)$, i.e, an unacknowledged source symbol s_e is retransmitted only if $p_d > p_e(s_e)$. This retransmission criterion prevents repetitive retransmissions of a given source symbol until the feedback corresponding to a retransmission arrives at the encoder.

The encoder injects the retransmissions of the unacknowledged packets into its output stream. During the retransmission process, the encoder adheres to the prescribed code rate $R = S/(S+1)$ by adding in one coded packet after S transmitted source symbols (irrespective of whether a source symbol is transmitted for the first time or retransmitted).

3) REFINED (OPTIMISTIC) RETRANSMISSION POLICY

The symbols marked by the bit mask are potentially lost and need to be retransmitted. But a retransmission is not always necessary. Since the feedback arrives with a delay, it does not reflect the current state of the decoder. The packets that were in flight when the feedback was generated and the packets that were newly transmitted up to the time instant when the feedback arrives at the encoder can still influence the state of the decoder. For example, a coded packet that was received during that time could have recovered a lost packet. Therefore, to enhance the operational efficiency, the encoder can estimate the state of the decoder at the time when the feedback is received.

The encoder checks if the lost source packets indicated by the feedback with decoder packet count p_d could have been recovered by the packets in flight, i.e., the packets that have been transmitted since the packet transmission with packet count p_d. Each lost source packet marked in the feedback is matched to a coded packet that includes the lost source packet. If there is no matching coded packet, then recovery is impossible and a retransmission is necessary, as illustrated for an example with three lost source packets and two in-flight coded packets in Fig. 3.

For each source symbol s_e in the encoding window, the encoder tracks the packet count $p_e(s_e)$ of the transmitted packet that contained the source symbol s_e. The encoder also tracks the packet counts p_e of the μ, $\mu \geq 0$, most recently transmitted coded packets. Ideally, μ is selected

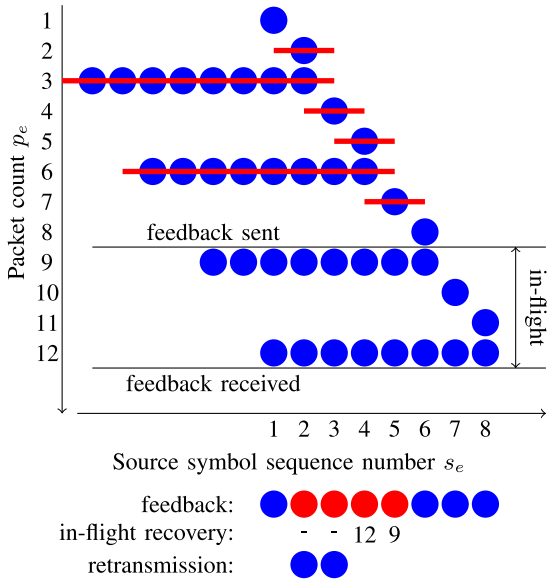


FIGURE 3. Example of matching lost source packets to coded packets. The feedback message with decoder packet count $p_d = 8$ indicating three lost source symbols is received at the encoder during or at the conclusion of the transmission with packet count $p_e = 12$. Thus, two coded packets are in flight which can recover two of the lost source packets. Hence, only one source symbol has to be retransmitted.

large enough to track all coded packets sent during an RTT. The RTT can either be acquired from external sources (e.g., TCP RTT), or from the encoder and decoder interactions by memorizing timestamps and packets and comparing them to packet numbers from the feedback. The parameter μ provides a tuning knob for the encoder behavior. A pessimistic encoder with $\mu = 0$ retransmits every lost source packet. With increasing μ, the encoder becomes more optimistic and triggers retransmissions only if the number of unseen packets at the decoder, as indicated by the received feedback, exceeds μ. Retransmissions follow the procedures of the basic retransmission policy in Section III-C.2, i.e., need to meet the retransmission criterion and comply with the prescribed code rate R.

IV. PERFORMANCE EVALUATION

A. EVALUATION SETUP

1) NETWORKING SCENARIO

We have implemented a time-slotted stochastic discrete event simulator to model the CRLNC-FB sender-receiver process with a binary erasure channel. Specifically, the channel allows for the transmission of one packet (one systematic source packet or one coded packet) per slot. The channel follows the two-state (good-bad) Gilbert-Elliott model [85], [86] with a prescribed average packet loss probability π_B and a prescribed expected number of consecutive packet erasures $E[L]$. We consider a range of common packet loss probabilities π_B in the range from 0.05 to 0.2 and common expected numbers of consecutive packet erasures $E[L]$ in the range from 16 to 256 slots. For 5G terrestrial wireless communication, $E[L]$ can be derived from the Rayleigh fading envelope, e.g., for a mobile client speed of 20 m/s and typical cellular system parameters (1.885 GHz carrier frequency, 10 dB fade margin) the average sojourn time in the bad (fading) channel state is on the order of 0.3 ms [87]–[90]. Thus, with a 1 Gbps bitrate, on the order of 25 packets of the 1500 Byte MTU size are lost on average in the bad state. Depending on the mobile client speed and cellular system parameters, a wide range of average sojourn times in the bad channel state can be expected for 5G systems. Also, satellite systems tend to have longer bad channel state sojourn times than cellular systems [91]–[97].

We model the channel propagation delays according to the review in Section II-D. Specifically, we consider 5G with 90 packet slots RTT, as well as LEO and GEO satellite communication with 210 and 524 packet slots RTT, respectively. The encoder-to-decoder channel and the decoder-to-encoder channel have identical independent Gilbert-Elliott models and the same one-way propagation delay.

A given simulation replication simulates the transmission of 128,000 source symbols utilizing the Kodo RLNC libraries [98] with a Galois field GF(2^8). All source symbols are available at the start of the simulation in the next higher protocol layer. The encoder pulls the source symbols from the next higher layer and processes them so that the encoder transmits one packet (systematic source symbol or coded packet) in each slot. We require that the decoder delivers packets in-order to the next higher protocol layer.

2) OPERATION OF CRLNC-FB AND BENCHMARK PROTOCOLS

We operate CRLNC-FB as specified in Section III with a window size of $W = 1024$. We do not limit the number of retransmissions. We implement a timeout to prevent the encoder from being blocked due to long bursts of lost packets or long bursts of lost feedback packets, see Section III-B.1.d. This timeout triggers a coded packet transmission when the encoding window is completely full and an RTT has expired since the last packet transmission. We evaluate the refined retransmission policy from Section III-C.3 with $\mu = 0$ and with $\mu = (1 - \pi_b)$ RTT$/S$. Note that RTT$/S$ coded packets are transmitted in an RTT, i.e., are in flight, when a feedback packet arrives at the encoder, and in the long run a proportion of $(1 - \pi_b)$ of these coded packets will arrive at the decoder.

We compare the introduced CRLNC-FB with conventional stop-and-wait ARQ, selective repeat ARQ, RLNC coded selective repeat ARQ [49], and Tetrys [16]–[19]. We operate all protocols (except stop-and-wait ARQ, which operates on individual packets) with a window size of $W = 1024$ source symbols; which is larger than the round-trip bandwidth-delay product of the considered communication channels. All benchmark approaches employ the same feedback frequency and detail as CRLNC-FB. For the block based RLNC coded selective repeat ARQ [49], we set the block size to 32 source

symbols and the window size to 32 blocks. For Tetrys we modified the timeout for preventing encoder blocking (see Section III-B.1.d) to transmit as many additional coded packets as there are currently unacknowledged source symbols in the encoding window.

3) PERFORMANCE METRICS

We define the in-order delay D as the time duration from the time instant when a packet would have been delivered without any losses to the time instant when the packet is actually delivered to the next higher protocol layer. The total delay of a source packet from the time instant when it is pulled by the encoder from the next higher protocol layer to the time instant when the packet is delivered by the decoder to the next higher protocol layer is then the considered in-order delay D plus the transmission time of one slot, plus the one-way channel propagation delay. We neglect the RLNC encoding and decoding computation time [99]–[101]. We report the 99 percentile of the in-order delay D. We define the throughput as the ratio of the total number of source symbols to be transmitted to the total number time slots required to complete the delivery to the next higher protocol layer at the decoder.

We define efficiency as the ratio of the total number of innovative packets (in the conventional sense of RLNC coding [102], [103]) received by the decoder to the total number of packets received by the decoder. Ideally, every packet received by the decoder should be innovative which corresponds to an efficiency of 1.0.

For each given evaluation scenario, we simulate 100 independent replications of the transmission of 128,000 source symbols. The resulting 95% confidence intervals are smaller than 2% of the sample means and are omitted from the plots to avoid clutter.

B. CRLNC-FB VARIANTS

Fig. 4 compares the throughput-delay performance of variants of the CRLNC-FB protocol. In particular, we compare CRLNC-FB utilizing the novel band-form Gaussian elimination RLNC decoding with the conventional forward Gaussian elimination (FGE) RLNC decoding in the context of the CRLNC-FB protocol. These two CRLNC-FB variants consider $\mu = 0$ in-flight packets and are compared with CRLNC-FB-Opt with band-form Gaussian elimination with $\mu = (1 - \pi_b)\text{RTT}/S$ considered in-flight packets (see Section III-C.3). Moreover, we compare with a CRLNC-FB variant (with band-form Gaussian elimination and $\mu = 0$) that retransmits only coded packets (CRLNC-FB-Cod). That is, the coded variant transmits a coded packet (generated from all presently unacknowledged source symbols) for each lost source symbol while enforcing the retransmission criterion from Section III-C.2 for coded packets transmitted in response to lost source symbols. The throughput-delay curves are obtained by evaluating the throughput and 99 percentile delay metrics for the code rates $R = S/(S + 1)$ corresponding to $S = 1, 2, 4, 8, 16$, and 32.

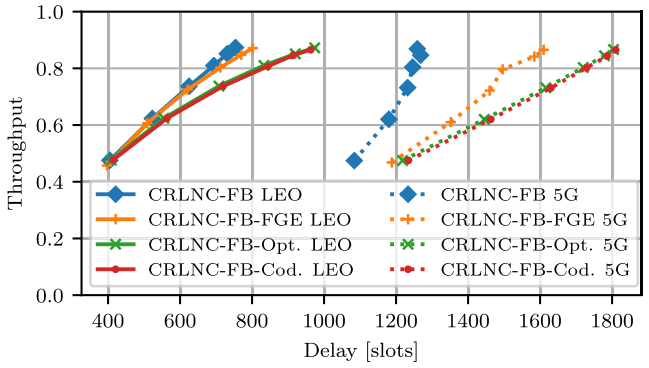


FIGURE 4. Throughput and 99%tile of packet delay of different CRLNC-FB variants for range of code rates $R = S/(S + 1)$ corresponding to $S = 1, 2, 4, 8, 16, 32$ source packets between coded packets for LEO (RTT = 210 slots) with $E[L] = 64$ successive packet losses and for 5G (RTT = 90 slots) with $E[L] = 256$ successive packet losses; Fixed packet loss probability $\pi_B = 10\%$.

1) BAND MATRIX CRLNC-FB VS. FORWARD GAUSSIAN ELIMINATION

We observe from Fig. 4 that for the $E[L] = 256$ (long loss burst) channel scenario, the FGE variant gives higher 99 percentile packet delays than CRLNC-FB with band-form Gaussian elimination. This difference is most pronounced for high code rates that approach one and give throughputs above 0.8. We observe from Fig. 4 that for the $E[L] = 64$ scenario the band-form decoding gives essentially the same delays as conventional forward Gaussian elimination decoding; we have confirmed this result also for $E[L] = 16$ (short loss burst) channel scenarios in additional evaluations that are not included here to avoid clutter. Long loss bursts $E[L]$ drop many successive source symbols that can typically not be recovered from the coded packets when the code rate R is high, i.e., when there are relatively few coded packets interspersed among the source packets. Thus, retransmissions are needed to recover the lost source packets. As illustrated in Fig. 2(c), the band-form decoding approach requests the retransmission of the systematic source packets at the beginning of a loss burst. Upon the receipt of the retransmitted systematic source packets, the decoder can immediately forward the source packets (in-order) to the higher layers without waiting for the decoding matrix to achieve full rank. In contrast, the forward Gaussian elimination decoding, as illustrated in Fig. 2(a), requests the lost source symbols from the tail end of a loss burst, and requires that the decoding matrix achieves full rank in order to recover the lost source packets from the beginning of the loss burst. For long loss bursts and correspondingly numerous lost source packets, achieving full rank requires the receipt of numerous retransmitted packets, which adds delays as observed in Fig. 4.

Moreover, upon the receipt of the retransmitted systematic source packets with the band-form approach, the decoding and encoding windows can be closed on the received packets, advancing the windows and enabling the transmission of new source packets. In particular, with the band-form approach, the windows advance at the granularity of individual source

symbols, whereas with the forward Gaussian elimination, the windows advance later and in steps of multiple source symbols (according to the loss pattern). In the example in Fig. 2(c) the receipt of the retransmitted systematic source symbol with sequence number 2 allows the decoder to deliver this source symbol to the next higher layer and to close the decoding window on this source symbol. The corresponding feedback packet acknowledges that the decoder has seen source symbol 2. Upon receipt of this feedback packet, the encoder can close its window on source symbol 2. Upon the receipt of the retransmitted systematic source symbol 3, the decoding matrix in Fig. 2(c) reaches full rank, hence source symbols 3–8 can be delivered to the next higher layer and the decoding window and the encoding window (after receipt of the corresponding feedback) can be closed on these symbols. In contrast, with forward Gaussian elimination, the receipt of retransmitted systematic source symbol 4 does not result in the delivery of any source symbols to the next higher layer, neither in an advance of the encoding window. Rather, the encoding window in forward Gaussian elimination only closes on the decoded source symbols (see Section III-B.2.a), i.e., in the example in Fig. 2(a), the encoding window still covers source symbols 2–9 after the receipt of the feedback corresponding to the receipt of retransmitted source symbol 4. Only after the receipt of the retransmitted source symbol 5 does the decoding matrix reach full rank, allowing for the recovery of source symbols 2 and 3, the delivery of source symbols 2–8 to the higher layer, the closing of the decoding window on source symbols 2–8, and the closing of the encoding window on source symbols 2–8 after receipt of the corresponding feedback packet. Thus, the band-form Gaussian elimination delivers some source symbols earlier than the forward Gaussian elimination and reduces the instances of the encoder being blocked from further transmissions due to a full window.

2) CRLNC-FB WITH OPTIMISTIC RETRANSMISSION POLICY

We also observe from Fig. 4 that optimistically considering the in-flight packets for the recovery of the lost source packets in CRLNC-FB-Opt generally reduces the throughput-delay performance. This is mainly because the optimistic consideration of the in-flight packets delays retransmissions. This effect of delaying retransmissions outweighs the advantages that come from sending more new source packets (instead of retransmissions); especially for practical window sizes that limit the number of in-flight packets to less than a few times the bandwidth-delay product. However, we observe from Fig. 5 that considering the in-flight packets very slightly improves the efficiency. In additional evaluations that are not included to avoid clutter, we found that high packet loss probabilities, long propagation delays, and large windows W generally increase the efficiency gain achieved by optimistically considering the in-flight packets (CRLNC-FB-Opt) over the basic CRLNC-FB that ignores the in-flight packets when deciding on retransmissions. For instance, for the GEO channel with a 20% packet loss probability

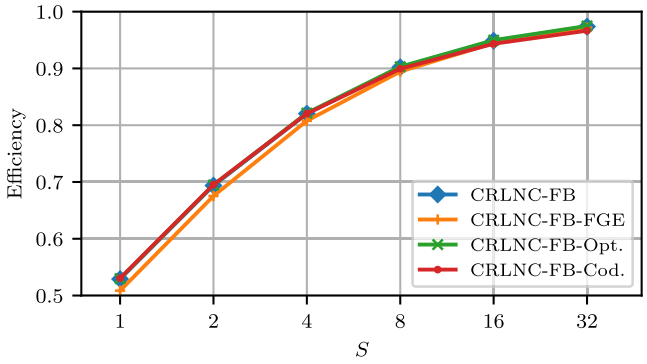


FIGURE 5. Efficiency of different CRLNC-FB variants as a function of number S of successive source symbols between coded packets for LEO (RTT = 210 slots) with a mean of $E[L] = 64$ successive packet losses for $\pi_B = 10\%$ loss.

(independently for each slot) and a window of $W = 4096$, the efficiencies are 0.784 vs. 0.748 for $R = 2/3$, and 0.885 vs. 0.870 for $R = 4/5$ for CRLNC-FB-Opt vs. CRLNC-FB. Thus, the efficiency gains (which correspond to reduced energy consumption) from the optimistic consideration of the in-flight packets are small to modest and need to be carefully weighed against the delay increases.

3) CRLNC-FB WITH CODED PACKET RETRANSMISSIONS

We observe from Fig. 4 that transmitting coded packets in the band-form Gaussian elimination (CRLNC-FB-Cod) instead of retransmitting systematic source packets gives slightly lower throughput-delay performance than the forward Gaussian elimination approach with systematic packet retransmissions (CRLNC-FB-FGE). In scenarios with one packet loss burst in the window, CRLNC-FB-Cod and CRLNC-FB-FGE have the same dynamics. In the example in Fig. 2(c), transmitting two coded packets instead of retransmitting the systematic source symbols 2 and 3, requires the receipt of both coded packets before the decoding matrix reaches full rank and before any symbols can be delivered and windows advanced. Specifically, after the receipt of the first coded packet, source symbol 3 is considered as seen, while source symbol 2 is still unseen.

CRLNC-FB-FGE can have an advantage over CRLNC-FB-Cod when there are two or more packet loss bursts in the window. In particular, imagine the scenario in Fig. 2(a) being duplicated in a doubled window covering source symbols 1–16, whereby source symbols 4 and 5 as well 12 and 13 are unseen, while the other source symbols 1–3, 6–11, and 14–16 are seen. Then, the receipt of the retransmitted source symbols 4 and 5 allows for the recovery of source symbols 2 and 3 in CRLNC-FB-FGE. In contrast, in CRLNC-FB-Cod, the receipt of two coded packets does not allow for the recovery of any source symbols, rather all four coded packets would need to be received and the entire decoding matrix would need to achieve full rank before any source symbols can be recovered.

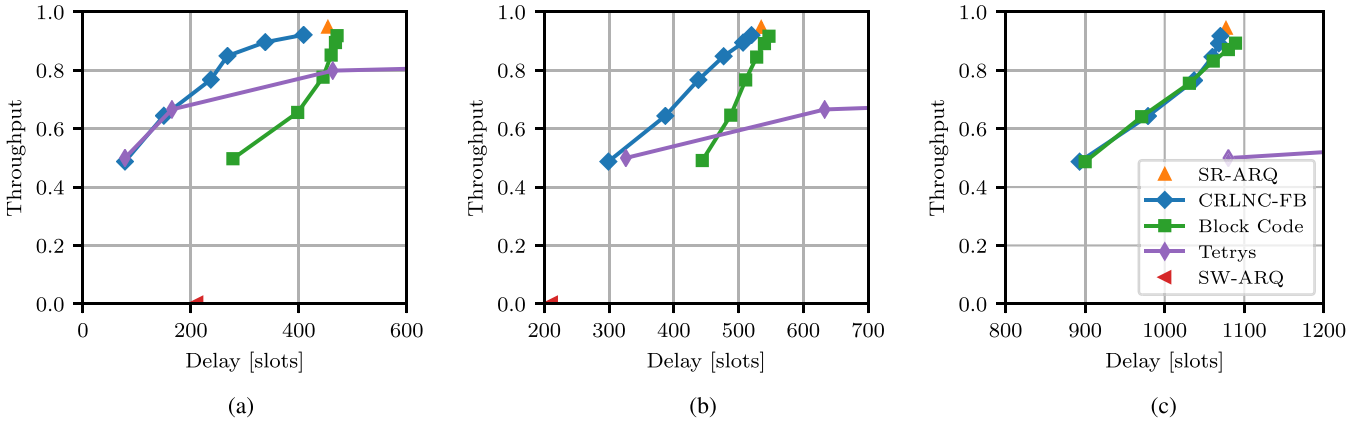


FIGURE 6. Throughput and 99%tile packet delay for LEO (RTT = 210 slots) with $\pi_B = 5\%$ loss for range of mean number of successive packet losses $E[L]$ for code rate $R = S/(S+1)$ corresponding to $S = 1, 2, 4, 8, 16, 32$. Results outside the plotted range are summarized in captions. (a) $E[L] = 16$; Tetrys reaches 0.88 throughput for 2400 slots delay. (b) $E[L] = 64$; Tetrys reaches 0.87 throughput for 7400 slots delay. (c) $E[L] = 256$; SW-ARQ has 420 slots delay; Tetrys reaches 0.86 throughput for delays exceeding 10000 slots.

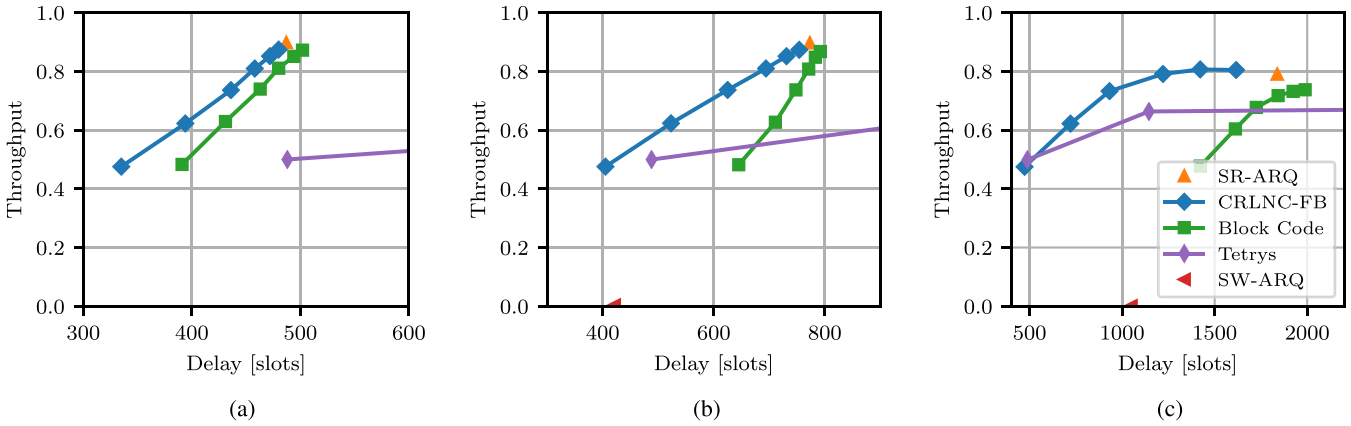


FIGURE 7. Throughput and 99%tile delay for $\pi_B = 10\%$ loss and a mean of $E[L] = 64$ successive packet losses. (a) 5G (RTT = 90 slots); SW-ARQ has 180 slots delay; Tetrys reaches 0.79 throughput for 3800 slots delay. (b) LEO (RTT = 210 slots); Tetrys reaches 0.78 throughput for 4100 slots delay. (c) GEO (RTT = 524 slots); Tetrys reaches 0.75 throughput delays exceeding 10000 slots.

For the remainder of this study we consider CRLNC-FB with the band-form Gaussian elimination RLNC decoding and $\mu = 0$ considered in-flight packets.

C. THROUGHPUT-DELAY RESULTS

1) CRLNC-FB VS. STOP-AND-WAIT ARQ

In Figs. 6 and 7 we plot the throughput-delay curves for CRLNC-FB and its benchmarks. Stop-and-wait ARQ (SW-ARQ) achieves low delay, but also very low throughput as only one packet at a time is pulled from the next higher layer at the sender. One RTT after a packet transmission, either an acknowledgement arrives at the sender to confirm the packet reception at the receiver (if there are no losses), or the time-out (see Section III-B.1.d) triggers a retransmission. Thus, the throughput is upper bounded by one source symbol per one RTT, while the efficiency is one. Network coding with a low code rate $R = 1/2$, which corresponds to the left-most point on the throughput-delay curves, sends one coded packet for each systematic source symbol. Accordingly, the throughput is upper bounded by one half. For short loss bursts ($E[L] = 16$), see Fig. 6(a), the delay of network coding with $R = 1/2$ is slightly below the delay for stop-and-wait ARQ as the coded packets recover lost packets at the decoder if the number of received coded packets exceeds the number of lost source symbols in the window.

On the other hand, for long loss bursts ($E[L] = 64$ and 256), see Figs. 6(b) and (c), stop-and-wait ARQ has a lower delay than the RLNC approaches with $R = 1/2$ as explained in the following. According to the recommendations for the transmission control protocol (TCP) [82], we consider a single retransmit timer, see Section III-B.1.d. Consider a source packet transmitted for the first time when a long loss burst (on the encoder-to-decoder channel) starts (while the decoder-to-encoder channel is lossfree and there have been no prior losses on the encoder-to-decoder channel). Stop-and-wait ARQ will repeatedly retransmit this packet until it is successfully delivered and acknowledged when the loss burst ends. In contrast, the encoder in the window-based

RLNC approaches will keep pulling source packets from the next higher layer and transmit the packets (until the window is full). The single timer for the oldest unacknowledged packet triggers retransmissions of the oldest unacknowledged packet until this packet is successfully received and the corresponding feedback packet reaches the encoder. The encoder then retransmits the lost source symbols. Thus, these subsequent source packets in the window (after the oldest unacknowledged packet) are delayed by the loss burst (whereas with stop-and-wait ARQ these subsequent packets are pulled from the next higher layer after the end of the loss burst).

2) CRLNC-FB VS. SELECTIVE REPEAT ARQ

Generally, we observe for all network coding approaches in Fig. 6 increasing delays for increasing code rate R. A higher code rate R implies fewer coded packets that could be used to recover lost source packets. Thus, more retransmissions are needed, increasing the delays. We observe from Fig. 6 that for very high codes rates R that approach one, the CRLNC-FB throughput-delay approaches the throughput-delay of selective repeat ARQ. For increasing code rates $R = S/(S+1)$ there are more and more systematic source symbols S between two coded packets. Thus, for very large S, the coded packets become negligible and CRLNC-FB degenerates to selective repeat ARQ. Selective repeat ARQ achieves the maximum possible efficiency by minimizing the number of unnecessary packet transmissions. Unnecessary packet transmissions occur with selective repeat ARQ only if feedback packets are lost and trigger a timeout to prevent encoder blocking, see Section III-B.1.d. With a perfect feedback channel, selective repeat ARQ would achieve an efficiency of one. We also observe that selective repeat ARQ achieves a fairly high throughput, although the RLNC approaches can achieve a higher throughput. In particular, for the long propagation delay GEO channel [see Fig. 7(c)], retransmissions are very time consuming, reducing the throughput. Thus, recovery of lost packets through RLNC can achieve higher throughput than relying only on retransmissions, as selective repeat ARQ does.

Nevertheless, in these cases where the RLNC throughput is higher than the selective repeat ARQ throughput, CRLNC-FB still converges to selective repeat ARQ as the code rate R increases. Thus, in these cases, CRLNC-FB exhibits decreasing throughput and increasing delay as the code rate R increases towards one. Intuitively, for high code rates R there are too few coded packets to recover the lost packets at the decoder. Hence, with increasing code rate R, more and more retransmissions are requested through feedback. These retransmissions add an additional RTT (or several RTTs for repeated retransmissions), increasing the delay. These delay increases accumulate to increase the overall time duration for delivering a set of packets and thus reduce the throughput. Moreover, for channels with high RTT, the more frequent retransmissions increase the likelihood of old unacknowledged source symbols preventing the encoding window from advancing. When the encoding window cannot advance due to old unacknowledged packets, then additional time slots elapse (without being utilized for transmissions), increasing the total number of required slots, and hence reducing the throughput.

3) CRLNC-FB VS. BLOCK BASED RLNC

We observe from Fig. 6 that block based RLNC coded selective repeat ARQ [49], gives significantly lower throughput delay performance than CRLNC-FB. Block based RLNC generally increases the delay compared to sliding window RLNC, i.e., the block based RLNC throughput delay curves are to the right (towards higher delays) compared to CRLNC-FB. These delay increases are mainly due to the block-by-block granularity of the RLNC encoding and decoding, i.e., a full block of source symbols is encoded or decoded at a time. In contrast, sliding window RLNC can advance the encoding and decoding windows at a granularity of a single source symbol, and thus inherently achieve lower delays. The delay introduced due to the block granularity can be reduced by reducing the block size. However, for a prescribed code rate $R = S/(S+1)$, the block size cannot be smaller than S.

4) CRLNC-FB VS. TETRYS

We observe from Fig. 6 that CRLNC-FB gives higher throughput-delay performance than Tetrys, especially for long loss bursts $E[L]$. Tetrys conforms to the prescribed code rate R throughout the retransmission process. If multiple successive source packets, say ℓ packets, are lost, then Tetrys needs to transmit ℓS new systematic (uncoded) source symbols along with the ℓ coded packets required to recover the ℓ lost source packets. For moderate to large S, i.e., for moderate to high code rates R, this transmission of the coded packets interspersed with the transmission of new source packets adds significant delays. Importantly, due to the in-order delivery requirement, the receiver needs to buffer the ℓS systematic source packets until the preceding ℓ lost packets can be recovered from the ℓ received coded packets. CRLNC-FB avoids these additional delays by directly retransmitting the lost source symbols, along with the corresponding coded packets according to the prescribed code rate R.

A related important aspect for understanding the Tetrys dynamics is that the Tetrys encoding window is allowed to cover non-consecutive source symbols. Thus, Tetrys does not require that the ℓ lost source symbols along with the ℓS source symbols (required for generating enough coded packets for recovering the ℓ lost symbols) fit into the encoding window W. Rather, Tetrys continues loss-free operation as long as the ℓ lost source symbols and the currently unacknowledged in-flight source symbols (up to RTT source symbols) fit into the encoding window. Several RTTs worth of source symbols may be required to generate enough coded packets to recover the ℓ lost symbols. In contrast, the CRLNC-FB encoding window covers up to W consecutive source symbols.

When packet losses occur towards the end of a file transmission there may not be enough source symbols left in the file to generate enough coded packets to recover from the losses. In such a case, Tetrys can rely on the timeout for preventing encoder blocking (see Sections III-B.1.d and IV-A.2) to generate additional coded packets.

Overall, the Tetrys throughput-delay performance is dominated by the mean duration $E[L]$ of loss bursts, as observed from Fig. 6. The Tetrys recovery of lost packets with coded packets interspersed at a fixed code rate $R = S/(S+1)$ among new source packets essentially amplifies the delay effects of a burst of ℓ lost packets by a factor of S. This delay effect could be mitigated by adapting the code rate R, e.g., reduce $R = S/(S+1)$ by reducing S, at the expense of the complexity of an adaptation algorithm [18]. An adaptive form of Tetrys could thus operate at the most suitable point along the throughput-delay curve. On the other hand, we observe from Fig. 7 that Tetrys is relatively insensitive to the RTT, while the throughput-delay performance of the other approaches is sensitive to both the RTT and the loss burst duration.

V. CONCLUSION

We have developed and evaluated Caterpillar RLNC with Feedback (CRLNC-FB). CRLNC-FB combines finite sliding window RLNC for forward error correction with feedback based selective repeat Automatic Repeat Request (ARQ) retransmission of lost source symbols. Extensive performance evaluations for erasure channels with bursty packet loss patterns have indicated that CRLNC-FB has higher throughput-delay performance than the prior approaches combining finite sliding window RLNC with feedback based retransmissions. In particular, CRLNC-FB achieves lower delays than the block based RLNC with feedback based transmission of coded packets [49] through the fine-granular advancement of the finite sliding coding window by individual source symbols. CRLNC-FB achieves higher throughput-delay performance than the Tetrys approach, which transmits new source symbols and coded packets according to a prescribed code rate in response to feedback [16]–[19], by retransmitting the lost source symbols in uncoded (systematic) form to promote fast in-order delivery at the receiver.

There are several important directions for future research related to CRLNC-FB. While this study considered a single wireless hop, an important future research direction is to examine recoding in multi-hop wireless networks, e.g., wireless mesh networks. Recoding at intermediate nodes could strengthen the FEC for subsequent highly lossy wireless hops. Recoding could also enhance mesh network transport where a given stream may send packets over multiple paths. Another direction is to examine finite sliding window RLNC with feedback in the context of multiple traffic flows. More specifically, this study considered the intra-coding of a single traffic flow. Future research could examine the inter-coding of multiple traffic flows to enhance the overall transport capability of lossy networks.

REFERENCES

[1] H. Alshaheen and H. Takruri-Rizk, "Energy saving and reliability for wireless body sensor networks (WBSN)," *IEEE Access*, vol. 6, pp. 16678–16695, 2018.

[2] I. Achour, T. Bejaoui, A. Busson, and S. Tabbane, "Network coding scheme behavior in a vehicle-to-vehicle safety message dissemination," in *Proc. IEEE ICC Workshops*, May 2017, pp. 441–446.

[3] H. Kang, H. Yoo, D. Kim, and Y.-S. Chung, "CANCORE: Context-aware network coded repetition for VANETs," *IEEE Access*, vol. 5, pp. 3504–3512, 2017.

[4] J.-S. Liu, C.-H. R. Lin, and J. Tsai, "Delay and energy tradeoff in energy harvesting multi-hop wireless networks with inter-session network coding and successive interference cancellation," *IEEE Access*, vol. 5, pp. 544–564, 2016.

[5] F. A. Monteiro *et al.*, "Special issue on network coding," *EURASIP J. Adv. Signal Process.*, vol. 2017, no. 29, pp. 29-1–29-3, Apr. 2017.

[6] A. Nessa, M. Kadoch, and B. Rong, "Fountain coded cooperative communications for LTE-A connected heterogeneous M2M network," *IEEE Access*, vol. 4, pp. 5280–5292, 2016.

[7] H. V. Nguyen, S. X. Ng, W. Liang, P. Xiao, and L. Hanzo, "A network-coding aided road-map of large-scale near-capacity cooperative communications," *IEEE Access*, vol. 6, pp. 21592–21620, 2018.

[8] B. Tang and S. Yang, "An LDPC approach for chunked network codes," *IEEE/ACM Trans. Netw.*, vol. 26, no. 1, pp. 605–617, Feb. 2018.

[9] R. Torrea-Duran, M. M. Céspedes, J. Plata-Chaves, L. Vandendorpe, and M. Moonen, "Topology-aware space-time network coding in cellular networks," *IEEE Access*, vol. 6, pp. 7565–7578, 2018.

[10] Y. Yang, W. Chen, O. Li, and L. Hanzo, "Joint rate and power adaptation for amplify-and-forward two-way relaying relying on analog network coding," *IEEE Access*, vol. 4, no. 1, pp. 2465–2478, 2016.

[11] Y. Yang, W. Chen, O. Li, Q. Liu, and L. Hanzo, "Truncated-ARQ aided adaptive network coding for cooperative two-way relaying networks: Cross-layer design and analysis," *IEEE Access*, vol. 4, pp. 9361–9376, 2016.

[12] R. Bassoli, H. Marques, J. Rodriguez, K. W. Shum, and R. Tafazolli, "Network coding theory: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 5, pp. 1950–1978, 4th Quart., 2013.

[13] M. Z. Farooqi, S. M. Tabassum, M. H. Rehmani, and Y. Saleem, "A survey on network coding: From traditional wireless networks to emerging cognitive radio networks," *J. Netw. Comput. Appl.*, vol. 46, pp. 166–181, Nov. 2014.

[14] J. Cloud and M. Médard, "Network coding over SATCOM: Lessons learned," in *Proc. Int. Conf. Wireless Satell. Syst.* Cham, Switzerland: Springer, 2015, pp. 272–285.

[15] X. Li, Q. Chang, and Y. Xu, "Queueing characteristics of the best effort network coding strategy," *IEEE Access*, vol. 4, pp. 5990–5997, 2016.

[16] J. Detchart, E. Lochin, J. Lacan, and V. Roca, *Tetrys, An On-the-Fly Network Coding Protocol*, document draft-detchart-nwcrg-tetrys-04, IETF, Internet-Draft, Mar. 2018.

[17] T. T. Thai, E. Lochin, and J. Lacan, "Online multipath convolutional coding for real-time transmission," in *Proc. IEEE Int. Packet Video Workshop (PV)*, May 2012, pp. 41–46.

[18] T. T. Thai, J. Lacan, and E. Lochin, "Joint on-the-fly network coding/video quality adaptation for real-time delivery," *Signal Process., Image Commun.*, vol. 29, no. 4, pp. 449–461, Apr. 2014.

[19] P. U. Tournoux, E. Lochin, J. Lacan, A. Bouabdallah, and V. Roca, "On-the-fly erasure coding for real-time video applications," *IEEE Trans. Multimedia*, vol. 13, no. 4, pp. 797–812, Aug. 2011.

[20] J. Barros, R. A. Costa, D. Munaretto, and J. Widmer, "Effective delay control in online network coding," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 208–216.

[21] J. Cloud and M. Médard, "Multi-path low delay network codes," in *Proc. IEEE GLOBECOM*, Dec. 2016, pp. 1–7.

[22] A. Garcia-Saavedra, M. Karzand, and D. J. Leith, "Low delay random linear coding and scheduling over multiple interfaces," *IEEE Trans. Mobile Comput.*, vol. 16, no. 11, pp. 3100–3114, Nov. 2017.

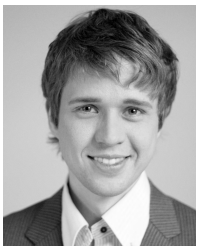
[23] M. Karzand and D. J. Leith, "Low delay random linear coding over a stream," in *Proc. Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2014, pp. 521–528.

[24] M. Karzand, D. J. Leith, J. Cloud, and M. Médard, "Design of FEC for low delay in 5G," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 8, pp. 1783–1793, Aug. 2017.

[25] M. Esmaeilzadeh, P. Sadeghi, and N. Aboutorab, "Random linear network coding for wireless layered video broadcast: General design methods for adaptive feedback-free transmission," *IEEE Trans. Commun.*, vol. 65, no. 2, pp. 790–805, Feb. 2017.
[26] P. Ostovari, J. Wu, and A. Khreishah, "Cooperative Internet access using helper nodes and opportunistic scheduling," *IEEE Trans. Veh. Technol.*, vol. 66, no. 7, pp. 6439–6448, Jul. 2017.
[27] V. Roca, B. Teibi, C. Burdinat, T. Tran-Thai, and C. Thienot, "Block or convolutional AL-FEC codes? A performance comparison for robust low-latency communications," Inria, Rocquencourt, France, Tech. Rep. hal-01395937v2, 2017.
[28] V. Roca, B. Teibi, C. Burdinat, T. Tran-Thai, and C. Thienot, "Less latency and better protection with AL-FEC sliding window codes: A robust multimedia CBR broadcast case study," in *Proc. IEEE Int. Conf. Wireless Mobile Comput., Netw. Commun. (WiMob)*, Oct. 2017, pp. 1–8.
[29] B. Sun, C. Gui, Y. Song, H. Chen, and X. Zhu, "Performance analysis of sliding window network coding in MANET," in *Proc. Adv. Comput. Archit. Conf.* Singapore: Springer, 2016, pp. 174–183.
[30] S. Wunderlich, F. Gabriel, S. Pandi, F. H. P. Fitzek, and M. Reisslein, "Caterpillar RLNC (CRLNC): A practical finite sliding window RLNC approach," *IEEE Access*, vol. 5, pp. 20183–20197, 2017.
[31] Y. Zeng, X. Xu, and R. Zhang, "Trajectory design for completion time minimization in UAV-enabled multicasting," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2233–2246, Apr. 2018.
[32] *IEEE Standard for Information Technology—Telecommunications and Information Exchange Between Systems—Local and Metropolitan Area Networks—Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard, IEEE 802.11 Working Group, 2012.
[33] H. Inamura, O. Takahashi, H. Nakano, T. Ishikawa, and H. Shigeno, "Impact of layer two ARQ on TCP performance in W-CDMA networks," in *Proc. Int. Conf. Distrib. Comput. Syst.*, 2004, pp. 284–291.
[34] E. Cabrera, G. Fang, and R. Vesilo, "Adaptive hybrid ARQ (A-HARQ) for ultra-reliable communication in 5G," in *Proc. IEEE Veh. Technol. Conf. (VTC Spring)*, Jun. 2017, pp. 1–6.
[35] S. Lin and D. J. Costello, *Error Control Coding*. New York, NY, USA: Pearson, 2004.
[36] E. Malkamaki, D. Mathew, and S. Hamalainen, "Performance of hybrid ARQ techniques for WCDMA high data rates," in *Proc. IEEE Veh. Technol. Conf. (VTC Spring)*, vol. 4, May 2001, pp. 2720–2724.
[37] V. Shah-Mansouri and S. Srinivasan, "Retransmission scheme for intra-session linear network coding in wireless networks," *Int. J. Adv. Intell. Paradigms*, vol. 9, no. 4, pp. 326–346, 2017.
[38] I. Tsokalo, F. Gabriel, S. Pandi, F. H. P. Fitzek, and R. Lehnert, "Reliable feedback mechanisms for routing protocols with network coding," in *Proc. IEEE Int. Symp. Power Line Commun. Appl.*, Apr. 2018, pp. 1–7.
[39] Y.-J. Chen, L.-C. Wang, K. Wang, and W.-L. Ho, "Topology-aware network coding for wireless multicast," *IEEE Syst. J.*, to be published.
[40] B. Chen *et al.*, "Packet multicast in cognitive radio ad hoc networks: A method based on random network coding," *IEEE Access*, vol. 6, pp. 8768–8781, 2018.
[41] E. Drinea, L. Keller, and C. Fragouli, "Real-time delay with network coding and feedback," *Phys. Commun.*, vol. 6, pp. 100–113, Mar. 2013.
[42] T. Niu, Z. Chen, D. Zhang, and K. Xu, "On minimizing average packets decoding delay based on B-DLNC for wireless broadcasting," in *Proc. IEEE Int. Conf. Commun. Technol. (ICCT)*, Oct. 2017, pp. 815–820.
[43] M. Yu, A. Sprintson, and P. Sadeghi. (2018). "On the packet decoding delay of linear network coded wireless broadcast." [Online]. Available: https://arxiv.org/abs/1802.02727
[44] G. Hu, K. Xu, and Y. Xu, "ARNC multicasting of HDCP data for cooperative mobile devices with dual interfaces," *IEEE Commun. Lett.*, vol. 21, no. 11, pp. 2504–2507, Nov. 2017.
[45] N. Papanikos and E. Papapetrou, "Deterministic broadcasting and random linear network coding in mobile ad hoc networks," *IEEE/ACM Trans. Netw.*, vol. 25, no. 3, pp. 1540–1554, Jun. 2017.
[46] M. E. Migabo, T. O. Olwal, K. Djouani, and A. M. Kurien, "Cooperative and adaptive network coding for gradient based routing in wireless sensor networks with multiple sinks," *J. Comput. Netw. Commun.*, vol. 2017, Oct. 2017, Art. no. 5301462-1–5301462-10.
[47] X. Xu, Y. L. Guan, Y. Zeng, and C.-C. Chui, "Spatial-temporal network coding based on BATS code," *IEEE Commun. Lett.*, vol. 21, no. 3, pp. 620–623, Mar. 2017.
[48] F. Wu, C. Hua, H. Shan, and A. Huang, "Reliable network coding for minimizing decoding delay and feedback overhead in wireless broadcasting," in *Proc. IEEE Int. Symp. Pers. Indoor Mobile Radio Commun. (PIMRC)*, Sep. 2012, pp. 796–801.
[49] J. Cloud, D. Leith, and M. Médard, "A coded generalization of selective repeat ARQ," in *Proc. IEEE INFOCOM*, Apr./May 2015, pp. 2155–2163.
[50] S. Gheorghiu, A. L. Toledo, and P. Rodriguez, "Multipath TCP with network coding for wireless mesh networks," in *Proc. IEEE ICC*, May 2010, pp. 1–5.
[51] P. Karafillis, K. Fouli, A. ParandehGheibi, and M. Médard, "An algorithm for improving sliding window network coding in TCP," in *Proc. IEEE Conf. Inf. Sci. Syst.*, Mar. 2013, pp. 1–5.
[52] Y. Lin, B. Liang, and B. Li, "SlideOR: Online opportunistic network coding in wireless mesh networks," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–5.
[53] D. Malak, M. Médard, and E. M. Yeh. (2018). "Analysis of coded selective-repeat ARQ via matrix signal-flow graphs." [Online]. Available: https://arxiv.org/abs/1801.10500
[54] J. K. Sundararajan, D. Shah, and M. Médard, "ARQ for network coding," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2008, pp. 1651–1655.
[55] J. K. Sundararajan, D. Shah, M. Médard, S. Jakubczak, M. Mitzenmacher, and J. Barros, "Network coding meets TCP: Theory and implementation," *Proc. IEEE*, vol. 99, no. 3, pp. 490–512, Mar. 2011.
[56] J. K. Sundararajan, D. Shah, M. Médard, and P. Sadeghi, "Feedback-based online network coding," *IEEE Trans. Inf. Theory*, vol. 63, no. 10, pp. 6628–6649, Oct. 2017.
[57] A. Fu and P. Sadeghi, "Queue-based rate control for low feedback RLNC," in *Proc. IEEE ICC*, Jun. 2014, pp. 2841–2847.
[58] P. Garrido, D. Leith, and R. Aguero. (2018). "Joint scheduling and coding over lossy paths with delayed feedback." [Online]. Available: https://arxiv.org/abs/1804.04921
[59] R. Alegre-Godoy and M. Á. Vázquez-Castro, "Network coding for system-level throughput improvement in satellite systems," *Int. J. Satel. Commun. Netw.*, vol. 35, no. 6, pp. 551–570, Nov./Dec. 2017.
[60] M. Bacco and A. Gotta, "RLNC in satellite networks: A cooperative scenario for delivering M2M traffic," *Int. J. Satell. Commun. Netw.*, vol. 35, no. 6, pp. 605–620, Nov./Dec. 2017.
[61] A. Engelmann, W. Bziuk, A. Jukan, and M. Médard. (2017). "Exploiting parallelism in optical network systems: A case study of random linear network coding (RLNC) in Ethernet-over-optical networks." [Online]. Available: https://arxiv.org/abs/1707.02789
[62] G. Giambene, D. K. Luong, V. A. Le, T. de Cola, and M. Muhammad, "Transport layer performance combining multipath and network coding in mobile satellite networks," *Int. J. Satell. Commun. Netw.*, vol. 35, no. 6, pp. 583–603, Nov./Dec. 2017.
[63] Y. Li, J. Wang, S. Zhang, Z. Bao, and J. Wang, "Efficient coastal communications with sparse network coding," *IEEE Netw.*, vol. 32, no. 4, pp. 122–128, Jul./Aug. 2018.
[64] F. Vieira and J. Barros, "Network coding multicast in satellite networks," in *Proc. IEEE Next Gen. Internet Netw. (NGI)*, Jul. 2009, pp. 1–6.
[65] M. Laner, P. Svoboda, P. Romirer-Maierhofer, N. Nikaein, F. Ricciato, and M. Rupp, "A comparison between one-way delays in operating HSPA and LTE networks," in *Proc. IEEE Int. Symp. Modeling Optim. Mobile, Ad Hoc Wireless Netw. (WiOpt)*, May 2012, pp. 286–292.
[66] A. Larmo, M. Lindström, M. Meyer, G. Pelletier, J. Torsner, and H. Wiemann, "The LTE link-layer design," *IEEE Commun. Mag.*, vol. 47, no. 4, pp. 52–59, Apr. 2009.
[67] C.-X. Wang *et al.*, "Cellular architecture and key technologies for 5G wireless communication networks," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 122–130, Feb. 2014.
[68] G. Fettweis and S. Alamouti, "5G: Personal mobile Internet beyond what cellular did to telephony," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 140–145, Feb. 2014.
[69] *Home—Oneweb|OneWorld*. Accessed: Aug. 17, 2017. [Online]. Available: http://oneweb.world/
[70] *SpaceX Satellite Constellation*. Accessed: Aug. 17, 2017. [Online]. Available: https://en.wikipedia.org/wiki/-SpaceX_satellite_constellation
[71] Q. Chen, Y. Bai, L. Chen, and Z. Pang, "Design of LEO constellations providing Internet services based on SOC method," in *Proc. MATEC Web Conf.*, vol. 114, 2017, pp. 1–10.
[72] L. Lockefeer, D. M. Williams, and W. J. Fokkink, "Formal specification and verification of TCP extended with the window scale option," *Sci. Comput. Program.*, vol. 118, pp. 3–23, Mar. 2016.

[73] A. U. Shankar, "Verified data transfer protocols with variable flow control," *ACM Trans. Comput. Syst.*, vol. 7, no. 3, pp. 281–316, 1989.
[74] N. V. Stenning, "A data transfer protocol," *Comput. Netw.*, vol. 1, no. 2, pp. 99–110, Sep. 1976.
[75] S. Ding, X. He, J. Wang, and J. Liu, "Pre-decoding recovery mechanism for network coding opportunistic routing in delay tolerant networks," *IEEE Access*, vol. 6, pp. 14130–14140, 2018.
[76] J. Wang, K. Xu, Y. Xu, and D. Zhang, "Pseudo-systematic decoding of hybrid instantly decodable network code for wireless broadcasting," *IEEE Wireless Commun. Lett.*, to be published.
[77] J. Heide, M. V. Pedersen, F. H. P. Fitzek, and T. Larsen, "Network coding for mobile devices—Systematic binary random rateless codes," in *Proc. IEEE ICC Workshops*, Jun. 2009, pp. 1–6.
[78] M. Kwon and H. Park, "Analysis on decoding error rate of systematic network coding," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Jan. 2017, pp. 258–259.
[79] D. E. Lucani, M. Médard, and M. Stojanovic, "On coding for delay—Network coding for time-division duplexing," *IEEE Trans. Inf. Theory*, vol. 58, no. 4, pp. 2330–2348, Apr. 2012.
[80] S. Pandi, F. Gabriel, J. A. Cabrera, S. Wunderlich, M. Reisslein, and F. H. P. Fitzek, "PACE: Redundancy engineering in RLNC for low-latency communication," *IEEE Access*, vol. 5, pp. 20477–20493, 2017.
[81] R. Prior and A. Rodrigues, "Systematic network coding for packet loss concealment in broadcast distribution," in *Proc. IEEE Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2011, pp. 245–250.
[82] K. R. Fall and W. R. Stevens, *TCP/IP Illustrated: The Protocols*, vol. 1. Boston, MA, USA: Addison-Wesley, 2011.
[83] A. Kesselman and Y. Mansour, "Optimizing TCP retransmission timeout," in *Proc. Int. Conf. Netw.* Berlin, Germany: Springer, 2005, pp. 133–140.
[84] L. Ma, G. R. Arce, and K. E. Barner, "TCP retransmission timeout algorithm using weighted medians," *IEEE Signal Process. Lett.*, vol. 11, no. 6, pp. 569–572, Jun. 2004.
[85] E. N. Gilbert, "Capacity of a burst-noise channel," *Bell Syst. Tech. J.*, vol. 39, no. 5, pp. 1253–1265, Sep. 1960.
[86] E. O. Elliott, "Estimates of error rates for codes on burst-noise channels," *Bell System Tech. J.*, vol. 42, no. 5, pp. 1977–1997, Sep. 1963.
[87] C. Chien, M. B. Srivastava, R. Jain, P. Lettieri, V. Aggarwal, and R. Sternowski, "Adaptive radio for multimedia wireless links," *IEEE J. Sel. Areas Commun.*, vol. 17, no. 5, pp. 793–813, May 1999.
[88] F. H. P. Fitzek and M. Reisslein, "A prefetching protocol for continuous media streaming in wireless environments," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 10, pp. 2015–2028, Oct. 2001.
[89] C.-Y. Hsu, A. Ortega, and M. Khansari, "Rate control for robust video transmission over burst-error wireless channels," *IEEE J. Sel. Areas Commun.*, vol. 17, no. 5, pp. 756–773, May 1999.
[90] G. L. Stüber, *Principles of Mobile Communication*, 4th ed. Cham, Switzerland: Springer, 2017.
[91] J. M. Garcia-Rubia, J. M. Riera, P. Garcia-del-Pino, D. Pimienta-del-Valle, and G. A. Siles, "Fade and interfade duration characteristics in a slant-path Ka-band link," *IEEE Trans. Antennas Propag.*, vol. 65, no. 12, pp. 7198–7206, Dec. 2017.
[92] J. Goldhirsh and W. J. Vogel, "Mobile satellite system fade statistics for shadowing and multipath from roadside trees at UHF and L-band," *IEEE Trans. Antennas Propag.*, vol. 37, no. 4, pp. 489–498, Apr. 1989.
[93] J. Goldhirsh and W. J. Vogel, "Propagation effects for land mobile satellite systems: Overview of experimental and modeling results," NASA, Washington, DC, USA, Tech. Rep. NASA-RP-1274, Feb. 1992.
[94] M. S. Karaliopoulos and F. N. Pavlidou, "Modelling the land mobile satellite channel: A review," *Electron. Commun. Eng. J.*, vol. 11, no. 5, pp. 235–248, Oct. 1999.
[95] A. D. Panagopoulos, P.-D. M. Arapoglou, and P. G. Cottis, "Satellite communications at Ku, Ka, and V bands: Propagation impairments and mitigation techniques," *IEEE Commun. Surveys Tuts.*, vol. 6, no. 3, pp. 2–14, 3rd Quart., 2004.
[96] M. Rytir, M. Cheffena, P. A. Grotthing, L. E. Bráten, and T. Tjelta, "Three-site diversity at Ka-band satellite links in Norway: Gain, fade duration, and the impact of switching schemes," *IEEE Trans. Antennas Propag.*, vol. 65, no. 11, pp. 5992–6001, Nov. 2017.
[97] B. Vucetic and J. Du, "Channel modeling and simulation in satellite mobile communication systems," *IEEE J. Sel. Areas Commun.*, vol. 10, no. 8, pp. 1209–1218, Oct. 1992.
[98] M. V. Pedersen, J. Heide, and F. H. Fitzek, "Kodo: An open and research oriented network coding library," in *NETWORKING* (Lecture Notes in Computer Science), vol. 6827. Berlin, Germany: Springer, 2011, pp. 145–152.
[99] L. Nielsen, R. R. Hansen, and D. E. Lucani, "Latency performance of encoding with random linear network coding," in *Proc. IEEE Eur. Wireless Conf.*, May 2018, pp. 1–6.
[100] H. Shin and J.-S. Park, "Optimizing random network coding for multimedia content distribution over smartphones," *Multimedia Tools Appl.*, vol. 76, no. 19, pp. 19379–19395, Oct. 2017.
[101] S. Wunderlich, J. A. Cabrera, F. H. P. Fitzek, and M. Reisslein, "Network coding in heterogeneous multicore IoT nodes with DAG scheduling of parallel matrix block operations," *IEEE Internet Things J.*, vol. 4, no. 4, pp. 917–933, Aug. 2017.
[102] A. Fu, P. Sadeghi, and M. Médard, "Dynamic rate adaptation for improved throughput and delay in wireless network coded broadcast," *IEEE/ACM Trans. Netw.*, vol. 22, no. 6, pp. 1715–1728, Dec. 2014.
[103] H. Wang, S. Xiao, and C.-C. J. Kuo, "Random linear network coding with ladder-shaped global coding matrix for robust video transmission," *J. Vis. Commun. Image Represent.*, vol. 22, no. 3, pp. 203–212, Apr. 2011.

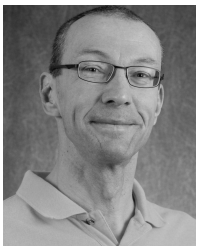
FRANK GABRIEL received the Dipl.-Inf. degree in computer science from the Technical University Chemnitz, Germany, in 2011. He is currently a Ph.D. Researcher with the Deutsche Telekom Chair of Communication Networks, Technical University Dresden.



SIMON WUNDERLICH received the Dipl.-Inf. degree in computer science from Chemnitz Technical University, Germany, in 2009. He is currently pursuing the Ph.D. degree in electrical engineering with the Technical University Dresden, Germany. He has co-authored *Wi-Fi Mesh software B.A.T.M.A.N. Advanced*.

SREEKRISHNA PANDI was born in Chennai, India. He received the B.E. degree in electronics and instrumentation engineering from Anna University in 2013, and the master's degree in nano-electronic systems from the Technical University Dresden (TU Dresden), Dresden, Germany, in 2015. Since 2015, he has been with the Deutsche Telekom Chair of Communication Networks, TU Dresden, where he is currently a Ph.D. Researcher.

FRANK H. P. FITZEK received the Diploma (Dipl.-Ing.) degree in electrical engineering from the University of Technology – Rheinisch-Westfälische Technische Hochschule (RWTH), Aachen, Germany, in 1997, the Ph.D. (Dr.-Ing.) degree in electrical engineering from the Technical University Berlin, Germany, in 2002, and the Honorary degree "Doctor Honoris Causa" from the Budapest University of Technology and Economy in 2015. He was an Adjunct Professor with the University of Ferrara, Italy, in 2002. In 2003, he joined Aalborg University as an Associate Professor, where he became a Professor. He co-founded several start-up companies, starting with Acticom GmbH, Berlin, in 1999. He is currently a Professor and also the Head of the Deutsche Telekom Chair of Communication Networks, Technical University Dresden, Germany, where he coordinates the 5G Lab Germany. His current research interests are in the areas of wireless and mobile 5G communication networks, mobile phone programming, network coding, cross layer, as well as energy efficient protocol design and cooperative networking. He was a recipient of the NOKIA Champion Award several times in a row from 2007 to 2011. In 2008, he received the Nokia Achievement Award for his work on cooperative networks. In 2011, he received the SAPERE AUDE Research Grant from the Danish Government. In 2012, he received the Vodafone Innovation prize.



MARTIN REISSLEIN (S'96–A'97–M'98–SM'03–F'14) received the Ph.D. degree in systems engineering from the University of Pennsylvania in 1998. He is currently a Professor with the School of Electrical, Computer, and Energy Engineering, Arizona State University, Tempe, AZ, USA. He chairs the Steering Committee of the IEEE TRANSACTIONS ON MULTIMEDIA. He currently serves as an Associate Editor of the IEEE TRANSACTIONS ON MOBILE COMPUTING, the IEEE TRANSACTIONS ON EDUCATION, the IEEE ACCESS, and *Computer Networks*. He is an Associate Editor-in-Chief of the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS and a Co-Editor-in-Chief of *Optical Switching and Networking*.