Fulcrum

Flexible Network Coding for Heterogeneous Devices

Lucani, Daniel E.; Pedersen, Morten V.; Ruano Benito, Diego; Sørensen, Chres W.; Fitzek, Frank H. P. ; Heide, Janus; Geil, Hans Olav; Nguyen, Vu; Reisslein, Martin

Published in:
IEEE Access

DOI (link to publication from Publisher):
10.1109/ACCESS.2018.2884408

Publication date:
2018

Document Version
Publisher's PDF, also known as Version of record

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 8, 2018, accepted November 26, 2018, date of publication November 30, 2018,
date of current version December 31, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2884408

Fulcrum: Flexible Network Coding for Heterogeneous Devices

DANIEL E. LUCANI[1], (Senior Member, IEEE), MORTEN VIDEBÆK PEDERSEN[2], DIEGO RUANO[3,4], CHRES W. SØRENSEN[5], FRANK H. P. FITZEK[6], JANUS HEIDE[2], OLAV GEIL[7], VU NGUYEN[6], AND MARTIN REISSLEIN[8], (Fellow, IEEE)

[1]DIGIT Centre, Department of Engineering, Aarhus University, 8200 Aarhus, Denmark
[2]Steinwurf ApS, 9220 Aalborg, Denmark
[3]IMUVA-Mathematics Research Institute, University of Valladolid, 47011 Valladolid, Spain
[4]Department of Mathematical Sciences, Aalborg University, 9200 Aalborg, Denmark
[5]Chocolate Cloud ApS, 9220 Aalborg, Denmark
[6]5G Lab Germany, Deutsche Telekom Chair of Communication Networks, Technische Universitaet Dresden, 01069 Dresden, Germany
[7]Department of Mathematical Sciences, Aalborg University, 9200 Aalborg, Denmark
[8]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 the Green Mobile Cloud project under Grant DFF-0602-01372B, in part by the Colorcast project under Grant DFF-0602-02661B, in part by the TuneSCode Project under Grant DFF - 1335-00125, and project Grant DFF-4002-00367 granted by the Danish Council for Independent Research, in part by the Spanish MINECO/FEDER Project under Grants MTM2012-36917-C03-03, Grant MTM2015-65764-C3-2-P, and Grant MTM2015-69138-REDT, in part by the RYC-2016-20208 (AEI/FSE/UE), and in part by the Aarhus Universitets Forskningsfond Starting Grant Project AUFF-2017-FLS-7-1.

ABSTRACT We introduce Fulcrum, a network coding framework that achieves three seemingly conflicting objectives: 1) to reduce the coding coefficient overhead down to nearly n bits per packet in a generation of n packets; 2) to conduct the network coding using only Galois field $GF(2)$ operations at intermediate nodes if necessary, dramatically reducing computing complexity in the network; and 3) to deliver an end-to-end performance that is close to that of a high-field network coding system for high-end receivers, while simultaneously catering to low-end receivers that decode in $GF(2)$. As a consequence of 1) and 3), Fulcrum has a unique trait missing so far in the network coding literature: providing the network with the flexibility to distribute computational complexity over different devices depending on their current load, network conditions, or energy constraints. At the core of our framework lies the idea of precoding at the sources using an expansion field $GF(2^h)$, $h > 1$, to increase the number of dimensions seen by the network. Fulcrum can use any high-field linear code for precoding, e.g., Reed-Solomon or Random Linear Network Coding (RLNC). Our analysis shows that the number of additional dimensions created during precoding controls the trade-off between delay, overhead, and computing complexity. Our implementation and measurements show that Fulcrum achieves similar decoding probabilities as high field RLNC but with encoders and decoders that are an order of magnitude faster.

INDEX TERMS Decoding probability, random linear network coding (RLNC), resource-constrained devices, throughput.

I. INTRODUCTION

A. NETWORK CODING OVERVIEW

Ahlswede *et al.* [2] proposed network coding (NC) as a means to achieve network capacity of multicast sessions as determined by the min-cut max-flow theorem [3], a feat that was provably unattainable using standard store-and-forwarding of packets (routing). NC breaks with this store-and-forward packet routing paradigm, encouraging intermediate network nodes to mix (recode) data packets. Thus, network coding proposed a store-code-forward paradigm to network operation, essentially extending the set of functions assigned to intermediate network nodes to include coding [4], [5]. Linear network codes were shown to be sufficient to achieve multicast capacity [6]. RLNC provides an asymptotically optimal and distributed approach to create linear combinations using random coefficients at intermediate network nodes [7].

Network coding has shown significant gains in a multitude of settings, from wireless networks [8]–[22] and

multimedia transmission [23]–[27], to distributed storage and content distribution [28]–[33] and Peer-to-Peer (P2P) networks [34]–[36]. Practical implementations have also confirmed the gains and capabilities of NC [37]–[39]. The reason behind these gains lies in two facts. First, the network does not need to transport each packet without modification, which opens more opportunities and freedom to deliver the data to the receivers and increases the impact of each transmitted coded packet (a linear combination of the original packets). Second, receivers no longer need to track individual packets, but instead accumulate enough independent linear combinations in order to recover (decode) the original packets. These relaxations have a profound impact on system designs and achievable gains.

B. MOTIVATION: HETEROGENEOUS NETWORKING SCENARIOS AND DEVICES

After more than a decade of research and in spite of NC's theoretical gains in throughput, delay, and energy performance, its widespread assimilation remains elusive. One, if not the most, critical weakness of the NC technology is the inherent complexity that NC introduces into network devices. This complexity is driven by two factors. First, devices must perform additional processing, which may limit the energy efficiency gains or even become a bottleneck in the system's overall throughput if processing is slower than the incoming/outgoing data rates [40]–[42]. This additional effort can be particularly onerous if we consider that the conventional wisdom dictates that large field sizes *are needed* to provide high reliability, throughput, and delay performance [43], [44]. In addition to the computational burden, the use of high field sizes comes at the cost of a high signaling overhead to communicate the coefficients used for coding the data packets. Other alternatives, e.g., sending a seed for a pseudo-random number generator, are relevant for end-to-end communication, but do not allow for a simple recoding mechanism in intermediate network nodes. Interestingly, [43] showed that using small to moderate field sizes, e.g., $GF(2)$ (whereby we denote $GF(q)$ and $\mathbb{F}_q$ for finite fields of size q), is key to achieving a reasonable trade-off among computational complexity, throughput performance, and total overhead, especially when recoding data packets in intermediate network nodes. This is encouraging since $GF(2)$ encoding/decoding could be as fast as 160 Mbps and 9600 Mbps in a 2009 mobile phone and laptop [45], respectively, while in 2013 the speeds increased by five-fold in high-end phones [46]. Even limited sensors, e.g., TelosB motes, can generate packets in $GF(2)$ at up to 500 kbps [47]. The $GF(2)$ decoding goodput can reach up to 100 Mbps and over 1000 Mbps on the Samsung S3 and S5, respectively [48], while 20 Mbps and 110 Mbps can be reached with the Raspberry Pi 1 and 2, respectively [49].

Second, devices must support different configurations, e.g., different field sizes, for each application or data flow, to achieve a prescribed target performance. Supporting disparate configurations translates into high costs in hardware, firmware, or software. In computationally constrained devices, e.g., sensors, the support for encoding, recoding, or decoding in higher fields is prohibitive due to the required processing complexity. On the other end of the spectrum, computationally powerful devices may also be unable to support multiple configurations. For example, high-load, high-speed Internet routers would require deep packet inspection to determine the coding configuration, followed by a different treatment of each incoming packet. This translates into additional expensive hardware to provide high processing speeds. Additionally, intermediate network nodes are typically heterogeneous, which limits the system's viable configurations.

A separate, yet related practical issue is that receivers interested in the same data flow may have widely differing computational, display, and battery capabilities as well as different network conditions. This end-device heterogeneity may restrict service quality at high-end devices when support is required for low-end devices, may deny service to low-end devices for the benefit of high-end devices, or may require the system to invest additional resources supporting parallel data flows, each with distinct characteristics matching different sets of end devices.

A straightforward option to solve the compatibility and complexity challenges is to limit sources, intermediate nodes, and receivers to use only $GF(2)$. However, using only $GF(2)$ may prevent high-end devices from achieving high reliability and throughput performance. Is it possible to develop a single, easily implementable, and compatible network coding framework that supports flows with different end-to-end requirements?

C. CONTRIBUTION: FULCRUM NETWORK CODING FRAMEWORK

We address the need for a flexible network coding framework by developing and evaluating the Fulcrum network coding framework. The name Fulcrum is derived from the meaning of ''fulcrum'' as the point on which a mechanical lever pivots in a simple mechanical machine. The Fulcrum network coding framework combines an outer and an inner network code and achieves its flexibilities by pivoting between different operational modes of the outer and inner network codes.

More specifically, Fulcrum uses only $GF(2)$ operations (inner network coding) in the network (see Fig. 1), to reduce overhead and computational cost, as well as to ensure compatibility for heterogeneous devices and data flows. At the same time, Fulcrum provides the opportunity to employ higher fields end-to-end via a tunable and straightforward precoding (outer network coding) mechanism for higher performance. Fig. 1 shows a Fulcrum example, where two sources operate using different fields $GF(2^h)$ and $GF(2^b)$ for source 1 and 2, respectively. The intermediate nodes in the network use only $GF(2)$ operations. With Fulcrum network codes, the left-most receiver of flow 2 (the nature image from source 2) can choose to decode using $GF(2)$ only as it has limited computation capabilities. Since the left-most receiver of flow 1 (the Lena image from source 1) has a better

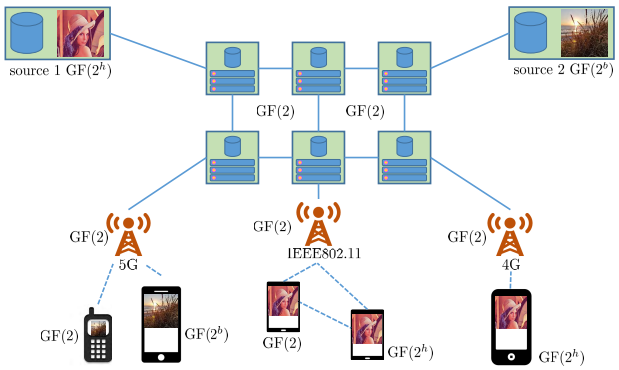


FIGURE 1. Fulcrum network codes allow sources and receivers to operate at higher field sizes to achieve high performance but maintain compatibility with the $GF(2)$-only network. Receivers can choose to trade off delay with decoding effort by choosing to decode with $GF(2)$ or in higher fields.

channel than the other devices and the router may have to broadcast for a longer time due to the other receiver, this left-most receiver can choose to save energy on computation by accumulating additional packets and decoding using $GF(2)$. Furthermore, this left-most receiver can also recode packets and send them to a neighbor interested in the same content, thus increasing the coverage of the system and reducing the number of transmissions needed to deliver the content.

II. SPECIFICATION OF FULCRUM NETWORK CODING FRAMEWORK

The key goals of the Fulcrum network coding framework are:

1) Reduce the overall overhead of network coding (a) by reducing the overhead due to coding coefficients per packet, and (b) by reducing the overhead due to transmission of linearly dependent packets.
2) Provide simple operations at the routers/devices in the network. The key is to make recoding at these devices as simple as possible, without compromising network coding capabilities.
3) Enable a simple and adaptive trade-off between performance and complexity.
4) Support compatibility with any end-to-end linear erasure code in $GF(2^h)$, $h > 1$.
5) Control and choose desired performance and effort for a variety of applications in end devices, while providing intermediate nodes with simple compatible network coding.

A. GENERAL UNDERLYING PRINCIPLES

The key technical idea of Fulcrum is to introduce a dimension expansion step. In particular, a batch of n source packets, typically called a generation, from the original file or stream is expanded into $n + r$ coded packets, whereby the r coded packets contain redundant information and are called *expansion packets*. After the expansion, each resulting coded packet is treated as a new packet that will be coded in $GF(2)$ and sent through the network, see Fig. 2.

Since addition in any field of the type $GF(2^k)$, $k \geq 1$, is simply a bit-wise Exclusive OR (XOR) operation, the underlying linear mapping in higher fields can be reverted at the receivers. The reason for the expansion is related to the performance of $GF(2)$, which can introduce non-negligible overhead in some settings [15], [44]. More specifically, $GF(2)$ coded packets have a relatively high probability of being linearly dependent when large data sets are available at the receiver. Increasing dimensions addresses this linear dependency problem by mapping back to the high field representation after receiving n linearly independent coded packets and decoding before the probability of receiving independent combinations in $GF(2)$ becomes prohibitively low, as analyzed in detail in Section III. The number r of additional dimensions (expansion packets) controls the decoding probability. The larger r, the higher the decoding probabilities achieved by the receivers while still using $GF(2)$ in the network.

Our approach naturally divides the problem into the design of inner and outer codes, using the nomenclature of concatenated codes [50]. Concatenating codes is a common strategy in coding theory, but has typically only been used for increasing throughput performance point-to-point [50] or end-to-end, e.g., Raptor codes [51]. Some recent NC studies have considered the idea of using concatenation (i) to create overlapping generations (with the same field size in the inner and outer code) so as to make the system more robust to time-dependent losses [52], [53]; (ii) to decompose the network into small sub-networks in order to simplify cooperative relaying [54]; (iii) to connect NC and error correcting channel coding, e.g., [55]; or (iv) to design subspace codes for noncoherent network coding [56]. Fulcrum is fundamentally disruptive in two important ways. First, we allow the outer code to be agreed upon by the sources and receivers (dimension expansion), while the inner code is created in the network by recoding packets. Thus, we provide a flexible code structure with controllable throughput performance. Second, Fulcrum provides a conversion from higher field ($GF(2^h)$, $h > 1$) arithmetic to $GF(2)$ to reduce computational complexity.

The division into two separate codes has an added advantage, not envisioned in previous approaches. This advantage comes from the fact that the senders can control the outer code structure to accommodate heterogeneous receivers. The simplest way to achieve this is by using a systematic structure in the outer code. This systematic outer coding structure provides the receivers with the alternative to decode in $GF(2)$ after receiving $n+r$ coded packets instead of mapping back to higher fields after receiving n coded packets. This decoding in $GF(2)$ has low decoding complexity, as $GF(2)$ operations are computationally simple, but incurs higher delay since r additional packets must be received.

If the precoding uses a systematic structure, the system can support three main types of receivers, see Fig. 2. First, a computationally powerful receiver can decode in $GF(2^h)$ by mapping back from the received $GF(2)$ combinations. We call

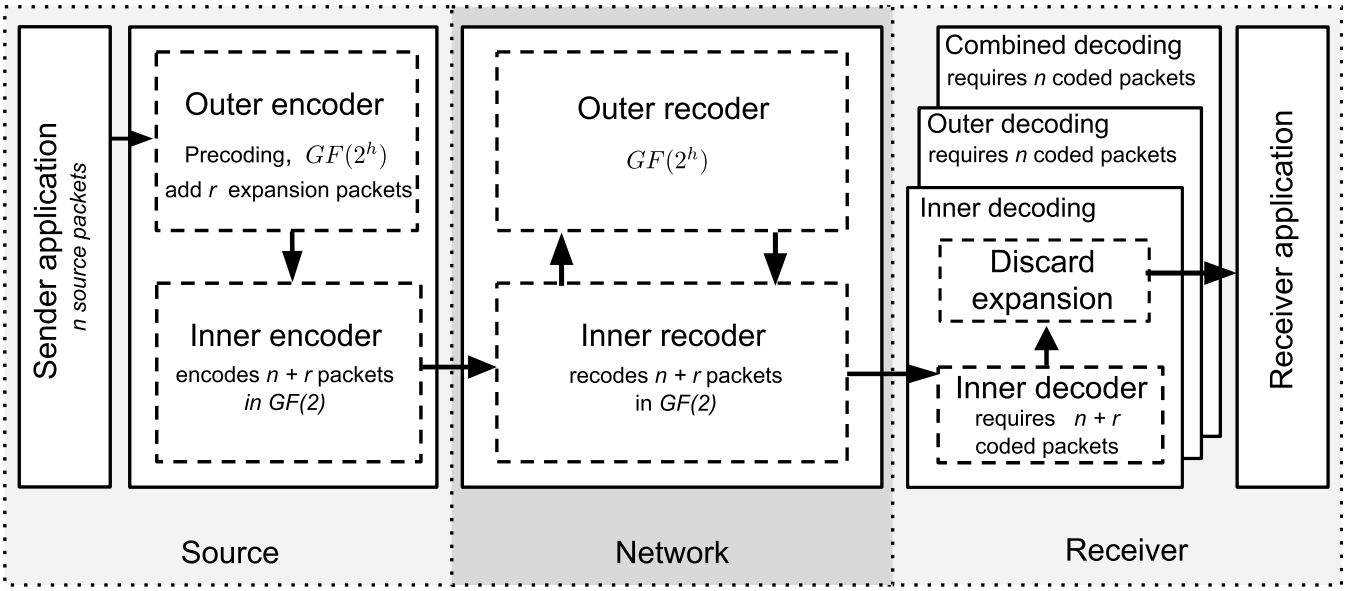


FIGURE 2. Illustration of Fulcrum network coding framework with outer $GF(2^h)$, $h > 1$, and inner $GF(2)$ code structures. The outer code is typically established end-to-end. The inner $GF(2)$ recoder has low computational complexity and supports a wide range of functionalities. (Some applications could use outer recoders at intermediate nodes for higher efficiency.) The sinks can choose from three main types of decoders: the inner, the outer, and the combined decoders. The outer decoder can operate with any configuration of outer/inner codes, while the inner and combined decoders require a specific structure of the outer code, i.e., a systematic code.

this the outer decoder. The mapping back procedure is simple because the addition in any extension field $GF(2^h), h \geq 1$, is the same as that in $GF(2)$, namely, a bit-by-bit XOR. We show that accumulating n linearly independent $GF(2)$ coded packets is sufficient to decode in the higher field. A receiver that decodes in $GF(2)$ reduces its decoding computing complexity but needs to gather $n + r$ independent linear combinations. Finally, we show that a hybrid decoder is possible, which can maintain the high decoding probability when receiving n coded packets as in the high-field decoder, while having similar decoding computing complexity to that of the inner decoder. We call this hybrid decoder the combined decoder.

Our work is inspired in part by Thomos and Frossard [57], who attempted to limit the overhead to a single symbol per packet. Thomos and Frossard carefully designed the packet coding at the source; however, only a small number of packets could be transmitted while maintaining the overhead at one symbol per packet. In contrast, we argue that the careful code construction is not really needed. Through our Fulcrum network coding framework, we break free from the constraint of a single symbol overhead and open up the potential (i) to reduce the overhead per packet in the network to roughly that of an end-to-end $GF(2)$ RLNC system (which is equivalent to the overhead reported in [57]), (ii) to trade off performance in the presence of heterogeneous receivers exploiting a family of precoders, and (iii) to exploit any generation size without introducing a synthetic constraint due to the field size at the precoder. Thus, the approach in [57] is a special subcase of our general Fulcrum framework.

B. FULCRUM ENCODING AT THE SOURCE

1) ENCODING SPECIFICATION

a: OUTER ENCODING

Using n original (input) source packets $P_1, P_2, \ldots, P_i, \ldots, P_n$, the source generates $n+r$ coded packets $\Omega_1, \Omega_2, \ldots, \Omega_j, \ldots, \Omega_{n+r}$ using $GF(2^h)$ operations, see Fig. 2-Source [7]. We refer to the additional r coded packets as *expansion packets*. We denote the outer coding coefficients as $\omega_{j,i}$, $j = 1, 2, \ldots, n + r; i = 1, 2, \ldots, n$. The outer encoding linearly combines the n source packets P_i, $i = 1, 2, \ldots, n$, "weighed" by the outer coding coefficients $\omega_{j,i}$ to form the outer coded packet

$$\Omega_j = \sum_{i=1}^{n} \omega_{j,i} P_i, \quad j = 1, 2, \ldots, n + r. \tag{1}$$

We refer to the vector of outer coding coefficients $\omega_{j,i}$, $i = 1, 2, \ldots, n$, denoted by $\{\omega_{j,i}\}_{i=1,2,\ldots,n}$, that are utilized to generate encoded packet Ω_j, $j = 1, 2, \ldots, n+r$, as the outer encoding vector of packet Ω_j.

For systematic outer encoding, the outer encoding vectors for coded packets Ω_j, $j = 1, 2, \ldots, n$, form an identity matrix of dimension $n \times n$, i.e., coded packets $C_j, j = 1, 2, \ldots, n$, are identical to the source packets P_j. However, the outer coding coefficients $\omega_{j,i}, j = n + 1, n + 2, \ldots, n + r; i = 1, 2, \ldots, n$ for the r expansion packets are randomly selected from $GF(2^h)$, i.e., the r expansion packets are linear combinations of the n source packets formed with random $GF(2^h)$ coding coefficients.

For non-systematic outer coding, all outer coding coefficients $\omega_{j,i}$, $j = 1, 2, \ldots, n+r; i = 1, 2, \ldots, n$, are randomly

selected from $GF(2^h)$, i.e., all $n+r$ outer coded packets are linear combinations of the n source packets.

b: INNER ENCODING

The source takes the outer coded packets Ω_j, $j = 1, 2, \ldots, n+r$, as input for the inner encoding, which is conducted in $GF(2)$. The source randomly selects inner coding coefficients $\iota_{k,j}$, $k = 1, 2, \ldots; j = 1, 2, \ldots, n+r$, from $GF(2)$ and forms inner coded packet C_k as

$$C_k = \sum_{j=1}^{n+r} \iota_{k,j} C_j. \tag{2}$$

Generally, the coding over $GF(2)$ is performed in accordance with the network's supported inner code. For example, if the network nodes support RLNC, then the source generates $GF(2)$ RLNC inner coded packets. We refer to the set of inner coding coefficients $\iota_{k,j}$, $j = 1, 2, \ldots, n+r$, that are utilized to generate the inner coded packet C_k, $k = 1, 2, \ldots$, as the inner encoding vector of packet C_k and denote this vector by $\{\iota_{k,j}\}_{j=1,2,\ldots,n+r}$. This inner encoding vector is included in the header of the encoded inner packet so as to enable recoding in intermediate network nodes. Note that for a Fulcrum coding structure with n source packets followed by r expansion packets, the first n coding coefficients $\iota_{k,j}$, $j = 1, 2, \ldots, n$ correspond to the n source packets (in uncoded form for a systematic outer encoding or coded form for a non-systematic outer coding). The next r coefficients correspond to the expansion packets.

Our main design constraint is that the receivers should (i) be able decode with the $n+r$ coded packets, and more importantly (ii) be able to decode with high probability after the reception of n coded packets. Given that the source controls the structure of the outer coding (and the positioning of the expansion packets), we could use a Reed-Solomon (RS) outer code, which is known end-to-end, or we could send the seed that was used to generate the random outer coding coefficients $\omega_{j,i}$, $j = 1, 2, \ldots, n+r; i = 1, 2, \ldots, n$ to the receivers. The Fulcrum coding framework assumes that the parameters generation size n, number of expansion packets r, and the outer coding coefficients $\omega_{j,i}$, $j = 1, 2, \ldots, n+r$; $i = 1, 2, \ldots, n$ are exchanged between source and receivers during the flow setup.

In order to cater to the capabilities of heterogeneous receivers, we recommend the use of a systematic outer encoding, which guarantees condition (i), but also provides interesting advantages for computationally constrained receivers as explained in more detail in Section II-D. With systematic outer encoding, the source and receivers only need to exchange outer coding coefficients $\omega_{j,i}$, $j = n+1, n+2, \ldots, n+r; i = 1, 2, \ldots, n$, during flow setup.

2) ENCODING IMPLEMENTATION

In this section we describe our actual implementation of the Fulcrum encoder in the Kodo network coding library [58]. For our initial implementation, we utilized two RLNC codes with the outer code operating in $GF(2^8)$ or $GF(2^{16})$ and the inner code operating in $GF(2)$. The encoder implementation in the Fulcrum framework is quite simple. Essentially, the two encoders can be implemented independently, whereby the outer encoder uses the n original source symbols to produce $n+r$ input symbols for the inner encoder. In general, the inner encoder can be oblivious to the fact that the input symbols may already contain encoded data.

For the initial implementation we required all source symbols to be available before any encoding could take place. This is however not necessary in cases where both encoders support systematic encoding. In such cases, it would be possible to push the initial n symbols directly through both encoders without any coding operations or adding additional delay. An illustration of this systematic encoding is shown in Fig. 3(a), where $n = 8$ original symbols are sent with the outer encoder configured to build an expansion of $r = 2$.

As both encoders in Fig. 3(a) are systematic, no coding takes place until steps 9 and 10, when the outer encoder produces the first encoded symbols. At this point, the inner encoder is still in the systematic phase and therefore passes the two symbols directly through to the network. In step 11, the inner encoder also exits the systematic phase and starts to produce encoded symbols. At this stage, the inner encoder is fully initialized and no additional symbols are needed from the outer encoder, all following encoding operations therefore take place in the inner encoder.

As shown in this simple example, the systematic structure in both encoders can be very beneficial for low delay applications because packets can be sent as they arrive at the encoder [59]–[63]. Systematic encoding is not always required for attaining this low delay. Alternatively, the inner encoder could be a standard RLNC encoder, only generating non-zero coefficients for the available symbols, i.e., using an on-the-fly encoding mechanism.

In the case of a non-systematic inner code, this low delay performance is typically not possible. However, there are several applications where non-systematic encoding may be more beneficial, e.g., for security, multiple-source and/or multi-hop networks. For data confidentiality, a systematic outer code can become a system vulnerability. A dense high field outer code is key to providing high levels of confidentiality.

As an example, Fig. 3(b) shows the use of a non-systematic outer encoder. Assuming the outer mapping is kept secret, only nodes with knowledge of the secret would be able to decode the actual content. Whereas all other nodes would still be able to operate on the inner code. Fig. 3(b) shows that it is also possible to use a non-systematic inner encoder. A non-systematic inner encoder can minimize the risk of transmitting linear dependent information in networks which may contain multiple sources for the same data, e.g., in peer-to-peer systems, or if the state of the sinks is unknown.

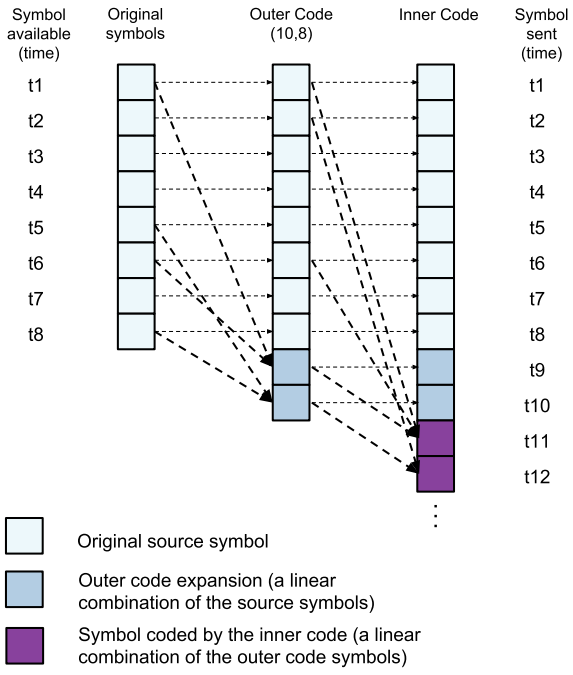


FIGURE 3. **Illustrative examples of outer and inner encoders in Fulcrum framework with $n = 8$ source packets and $r = 2$ expansion packets generated by the outer encoder. (a) Systematic outer encoder and systematic inner encoder. (b) Non-systematic outer encoder and non-systematic inner encoder.**

C. RECODING AT INTERMEDIATE NETWORK NODES

The operations at the intermediate network nodes are quite simple, see Fig. 2-Network. Essentially, the network nodes receive coded packets in $GF(2)$ of the form $\sum_{j=1}^{n+r} \iota_{k,j}\Omega_j$, store them in their buffers, and send recoded versions to the next hops, typically implementing an inner recoder as described in the following. The recoding mechanism defines the structure of the inner code of our Fulcrum system. Recoding can be done as a standard $GF(2)$ RLNC system would do, i.e., each packet in the buffer has a probability of $1/2$ to be XORed with the others to generate the recoded packet. However, the network can also support other recoding mechanisms, such as recoding for perpetual network codes [64] and for tunable sparse network coding [65], [66], or even no recoding.

In some scenarios, it may be possible to allow intermediate network nodes to know and to exploit the outer code, see Fig. 2-Network. In particular, when an intermediate node gathers n linearly independent coded packets in the inner code, then the node can map back to the higher field in order to decode the data and improve the quality of the recoded packets. The rationale is that, at that point, the node could recreate the original code structure and generate the additional dimensions r that are missing in the inner code, thus speeding up the transmission process. Although not required for the operation of the system, this optional mechanism can be useful if the network nodes are allowed to trade off throughput performance with complexity. This option is not examined in detail in this paper and is left for future research.

D. FULCRUM DECODING AT THE RECEIVERS

We initially assume a systematic outer code, which allows for three main types of receivers. In particular, the outer decoder can operate with arbitrary outer encoding, while the inner and combined decoder require a systematic outer encoding.

1) INNER DECODER

a: SPECIFICATION

Receivers using an inner decoder decode $n + r$ received $GF(2)$ coded packets using $GF(2)$ operations, see Fig. 4-left. The $GF(2)$ decoding is computationally fast, although there is some additional cost for decoding an $(n + r) \times (n + r)$ matrix. If the outer encoding had used a systematic structure, then the inner decoding provides the original packets without additional decoding in $GF(2^h)$. The penalty for this reduced computational effort is the additional delay incurred by waiting for $n+r$ independent linear combinations in $GF(2)$. Thus, there is no benefit over standard $GF(2)$; however, Fulcrum provides compatibility with other receivers that utilize higher $GF(2^h)$ field sizes.

b: IMPLEMENTATION

The inner decoder's implementation is very similar to a standard RLNC $GF(2)$ decoder configured to receive $n + r$ symbols, see Fig. 5. The only difference is that only n of the decoded symbols contain the original encoded source data.

2) OUTER DECODER

a: SPECIFICATION

Receivers using an outer decoder map back to the original linear combination in $GF(2^h)$, see Fig. 4-middle. This means

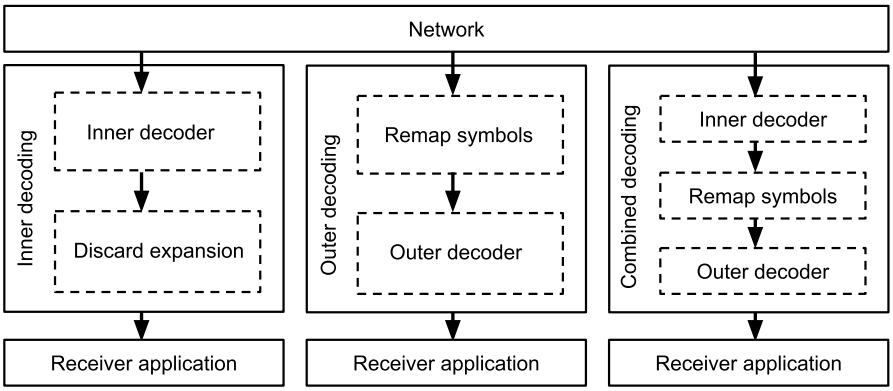


FIGURE 4. Overview of components of the three types of Fulcrum decoders. Each decoder operates on the same data stream coming from the network. This makes it possible to support heterogeneous receivers using some mix of the three decoding types.

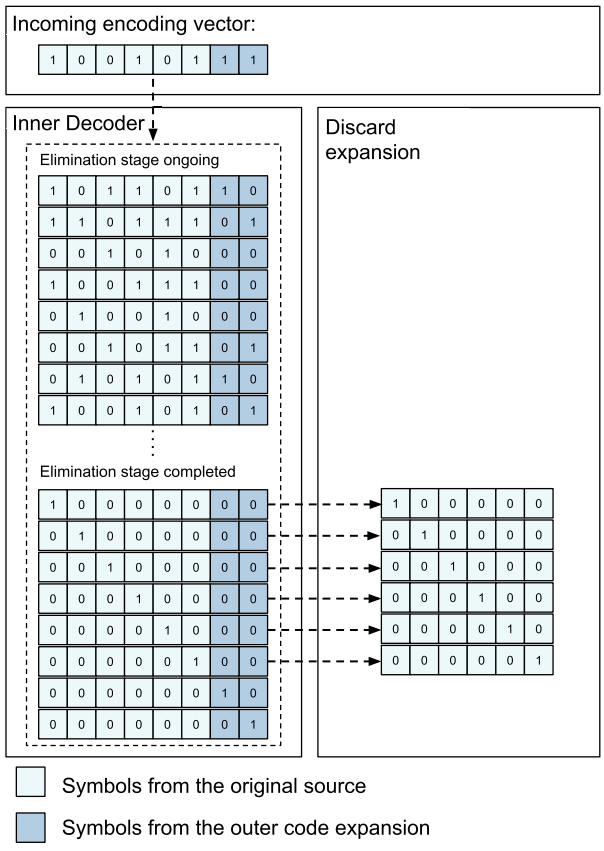


FIGURE 5. Illustration of coding coefficient vector processing for Fulcrum inner decoder: $n + r = 6 + 2$ linearly independent $GF(2)$ coded packets are decoded using $GF(2)$ decoding and the $r = 2$ (right-most) coding coefficient columns corresponding to the expansion packets are discarded to obtain the original source data (provided the outer encoding was systematic).

that after receiving n independent coded packets in $GF(2)$, the receiver can decode an $n \times n$ matrix in the original field $GF(2^h)$ with high probability. More specifically, coded packets in $GF(2)$ arrive to the receivers. Utilizing their knowledge of the positioning of the r expansion packets relative to the n source packets in the outer encoding, as well as the outer coding coefficients $\omega_{i,j}$, each receiver converts (maps back) the received $GF(2)$ coded packets to $GF(2^h)$.

For ease of understanding, we specify this mapping back in the context of a concrete example with $n = 4$ source packets P_i, $i = 1, 2, 3, 4$, and $r = 2$ expansion packets. Suppose the outer coding is systematic $GF(2^8)$ RLNC and the outer coding coefficient vector (outer encoding vector) for the first expansion packet is $\{\omega_{n+1,i}\}_{i=1,2,3,4} = \{\omega_{5,i}\}_{i=1,2,3,4} = \{192, 0, 95, 148\}$, while the outer encoding vector for the second expansion packet is $\{\omega_{n+2,i}\}_{i=1,2,3,4} = \{\omega_{6,i}\}_{i=1,2,3,4} = \{116, 0, 1, 86\}$. Suppose that the inner coding linearly combines the outer coded packets Ω_j, $j = 1, 2, \ldots, 6$ with $GF(2)$ RLNC with inner encoding vectors

$$\{\iota_{k=1,j}\}_{j=1,2,\ldots,6} = \{1, 0, 0, 0, 1, 1\}, \quad (3)$$
$$\{\iota_{k=2,j}\}_{j=1,2,\ldots,6} = \{1, 1, 0, 1, 0, 0\}, \quad (4)$$
$$\{\iota_{k=3,j}\}_{j=1,2,\ldots,6} = \{0, 0, 1, 1, 1, 0\}, \quad (5)$$
$$\{\iota_{k=4,j}\}_{j=1,2,\ldots,6} = \{1, 0, 1, 1, 0, 0\}. \quad (6)$$

The decoder receives the inner coded packet $k = 1$, including the inner encoding vector $\{\iota_{k=1,j}\}_{j=1,2,\ldots,6}$, and notes that both inner coding coefficients corresponding to the two expansion packets are one, i.e., $\iota_{k=1,j=5} = 1$ and $\iota_{k=1,j=6} = 1$. Accordingly, the decoder maps back to the outer code by XORing (denoted by $\oplus$) the inner coding vector corresponding to the n source packets as well as the two outer encoding vectors:

$$\{\iota_{k=1,j}\}_{j=1,2,3,4} \oplus \{\omega_{5,i}\}_{i=1,2,3,4} \oplus \{\omega_{6,i}\}_{i=1,2,3,4} = \{181, 0, 94, 194\}. \quad (7)$$

In the encoding vector for the second inner coded packet, both inner coding coefficients corresponding to the $r = 2$ expansion packets are zero, i.e., $\iota_{k=2,j=5} = 0$ and $\iota_{k=1,j=6} = 0$. Accordingly, the mapping back simply takes the first $n = 4$ coefficients, i.e., gives

$$\{1, 1, 0, 1\}. \quad (8)$$

For the third inner coded packet the mapping back gives:

$$\{\iota_{k=3,j}\}_{j=1,2,3,4} \oplus \{\omega_{5,i}\}_{i=1,2,3,4} = \{192, 0, 94, 148\}, \quad (9)$$

while for the fourth packet ($k = 4$), the mapping back gives

$$\{1, 0, 1, 1\}. \quad (10)$$

Thus, the full decoding matrix after the mapping back is an $n \times n = 4 \times 4$ matrix consisting of the vectors given in Eqns. (7), (8), (9), and (10), which can be decoded with standard $GF(2^8)$ RLNC decoding techniques to a 4×4 identity matrix so as to recover the original source packets.

Generally, the mapping back to $GF(2^h)$ for the outer decoder from the decoding matrix from the received inner encoding vectors $\{\iota_{k,j}\}_{j=1,2,\ldots,n+r;k\geq 1}$, as well as the expansion packet encoding vectors $\{\omega_{j,i}\}_{j=n+1,\ldots,n+r;i=1,2,\ldots,n}$, computes the mapped back encoding vectors for received $GF(2)$ encoded packet k, $k \geq 1$, as

$$\{\iota_{k,i}\}_{i=1,2,\ldots,n} \oplus_{j=n+1}^{n+r} \iota_{k,j}\{\omega_{j,i}\}_{i=1,2,\ldots,n}. \quad (11)$$

These outer-decoder receivers use computationally more complex $GF(2^h)$ operations for decoding packets, but require only n received packets to recover the necessary linear combinations to decode, while an inner decoder using only $GF(2)$ operations requires $n + r$ received packets.

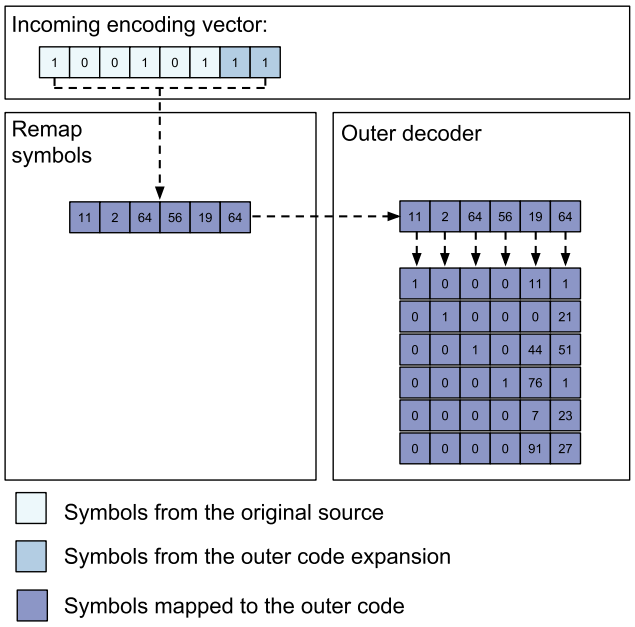


FIGURE 6. Illustration of coding coefficient vector processing in Fulcrum outer decoder: An inner encoding vector in $GF(2)$ is mapped directly back to the outer field $GF(2^h)$. Then, a standard $GF(2^h)$ decoder can be used to decode the data.

b: IMPLEMENTATION

The outer decoder immediately maps from the inner to the outer code, essentially decoding in $GF(2^h)$, as illustrated in Fig. 6. In order to perform this mapping, a small lookup table stores the outer expansion packet coding coefficients. The size of the lookup table depends on whether the outer encoder is systematic or not. In the case of a systematic outer encoder, a lookup table holding the expansion packet $GF(2^h)$ encoding coefficients $\omega_{j,i}, j = n+1, \ldots, n+r; i = 1, \ldots, n$ is sufficient, since the initial n symbols are uncoded (i.e., using the unit vector). However, in case of a non-systematic outer encoder all $n+r$ (for $j = 1, \ldots, n+r$) outer encoding vectors $\{\omega_{j,i}\}_{i=1,\ldots,n}$ need to be stored. An alternative approach would be to use a pseudo-random number generator to generate the encoding vectors on the fly as needed. One advantage of the lookup table is that it may be precomputed and therefore would not consume any additional computational resources during encoding/decoding.

3) COMBINED DECODER

a: SPECIFICATION

Receivers using a combined decoder implement a hybrid between inner and outer decoders with the aim of approaching the decoding (processing) speed of inner decoders while retaining the high decoding probability of outer decoders, see Fig. 4-right. This is achieved by decoding the first n coded packets using $GF(2)$ only. If decoding is unsuccessful in $GF(2)$, all coded packets are mapped to $GF(2^h)$ over which the remaining decoding is performed. Hence, if $r \ll n$ the decoding computation cost of the last r packets is negligible compared to the decoding computation cost of the initial n packets and the decoding processing speed will approach that of an inner decoder.

b: IMPLEMENTATION

The combined decoder attempts to decode as much as possible using the inner decoder before switching to the typically more computationally costly outer decoder. Note that combined decoding in only beneficial if the outer encoder is systematic (or, potentially, very sparse).

We explain the combined decoder implementation with the example shown in Fig. 7. When an encoding vector arrives at a combined decoder, it is first passed to the inner decoder. Internally, the inner decoder is split into two stages. Stage one attempts to eliminate the extension added in the outer encoder (these are the symbols that when mapped to the outer decoder will have coding coefficients from the outer $GF(2^h)$ field). If stage one successfully eliminates the expansion, then the symbol is passed to stage two. The stage two decoder has only linear combinations of original source symbols. These symbols have a trivial encoding vector when mapped to the outer decoder. Once stage one and stage two combined have achieved full rank, then the stored symbols are mapped to the outer decoder. Notice in Fig. 7 how symbols coming from stage two have coding coefficients 0 or 1, and thus require only a few operations to be decoded. On the other hand, the symbols coming from stage one have a dense structure with coding coefficients $\omega_{j,i}$ coming from the outer field $GF(2^h)$. After mapping to the outer decoder, the final step is to solve the linear system shown in the lower right of Figure 7.

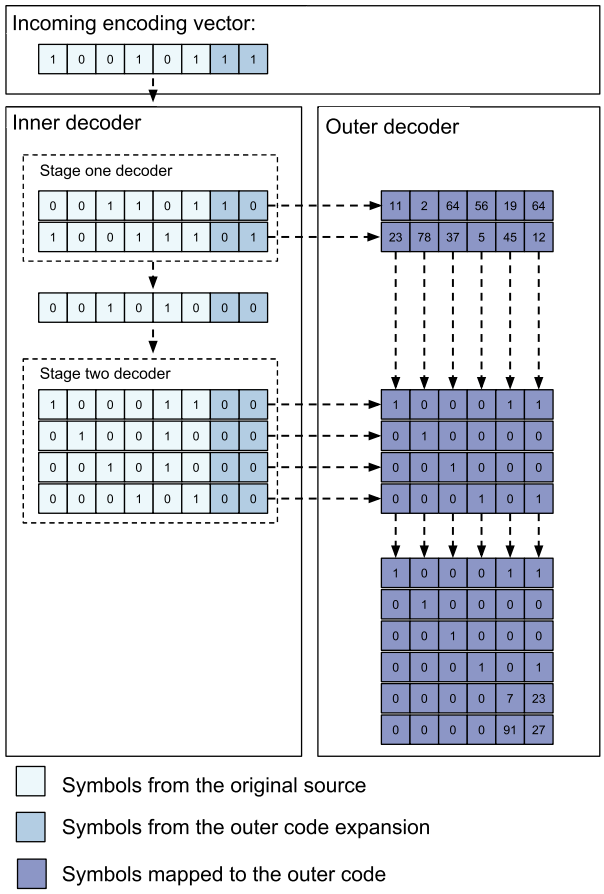


FIGURE 7. Illustration of coding coefficient vector processing in Fulcrum combined decoder: A two stage inner decoder eliminates as much of the contribution of the outer code as possible before mapping the symbols to the outer decoder. Combined decoding requires a systematic outer code.

III. ANALYSIS OF FULCRUM PERFORMANCE WITH RLNC AS INNER CODE

This section examines the delay, decoding probability, and overhead, as well as encoding and decoding throughput of Fulcrum with the outer and combined decoders. We present both theoretical analyses as well as measurements conducted with our Fulcrum implementation on several computing/communication devices, including smartphones. For the theoretical analysis we note that receivers using an inner decoder correspond to a conventional $GF(2)$ receiver that needs to gather $n + r$ independent linear combinations before decoding. The theoretical analysis of such conventional $GF(2)$ receivers is readily tractable [15], [44], [45].

A. PRELIMINARIES: MDS OUTER CODE PROPERTY FOR THEORETICAL ANALYSIS

A key question for the tractability of the theoretical analysis of receivers using the outer or combined decoders is whether receiving n independent coded packets in $GF(2)$ means that the re-mapped version in $GF(2^h)$ is full rank, i.e., whether the original data can be decoded in $GF(2^h)$. In coding theory terms, the tractability of the outer and combined decoder analysis requires (i) that the outer code satisfies the properties of maximum distance separable (MDS) codes [67] and (ii) that the outer code still satisfies the MDS properties after being combined with a restricted set of values due to the use of $GF(2)$ in the inner coding. For brevity, we refer to the combination of these two conditions as the "MDS outer code property", or refer to the outer code as an "MDS outer code". The Appendix explicitly demonstrates that this MDS outer code property is met for an RS outer code under some minor conditions. In particular, Theorem 4 in the Appendix shows that the MDS outer code property is met if the RS code dimension n is greater than or equal to 2^{h-1}, whereby h characterizes the field size $GF(2^h)$. For RS code dimensions smaller than 2^{h-1}, Theorem 5 in the Appendix constructs a basis that ensures that the MDS outer code property is met.

RLNC satisfies general MDS properties with increasing probabilities for increasing field sizes [4], [68]–[72]. However, the analysis of the MDS properties of RLNC after combination with a restricted set of values in the $GF(2)$ inner coding is beyond the scope of this study. The theoretical results in the following subsections apply therefore in an exact sense only to RS outer codes that satisfy Theorems 4 or 5. Nevertheless, the analyses provide insights into the dynamics of the Fulcrum system and we expect them to be a good approximation for a Fulcrum system with RLNC outer coding with moderately large to large field sizes. Indeed, our comparisons of the numerical results obtained from the theoretical analysis with measurement results obtained from our Fulcrum implementation on real devices in Section III-D demonstrate that the theoretical analysis is a good approximation for Fulcrum with RLNC outer encoding.

We emphasize that the actual operation of the Fulcrum network coding framework does *not* require any MDS properties of the employed codes. While large RLNC field sizes (and the resulting closeness to the MDS properties) generally improve the performance of the outer coding, large RLNC field sizes are not required for the correct Fulcrum operation. For good performance, the outer code should be decodable from n received outer coded packet, i.e., have rank equal to n with high probability.

TABLE 1. Computing devices for real system measurements.

Alias	Device	CPU
N6	Nexus 6	Quad-core 2.7 GHz Krait 450
N9	Nexus 9	Dual-core 2.3 GHz Denver
i5	Intel NUC D54250WYK	Dual-core 2.6 GHz Intel core i5-4250U
i7	Dell Latitude E6530	Quad-core 2.7 GHz Intel core i7-3740QM
Rasp v2	Raspberry PI 2 model B V1.1	Quad-core 900MHz ARM Cortex-A7 CPU
S5	Samsung S5	Quad-core 2.5 GHz Krait 400

B. FULCRUM MEASUREMENT SETUP

We conducted measurements on the devices in Table 1. We used a packet size of 1600 bytes and performed systematic outer Fulcrum encoding over $GF(2^8)$. We implemented the Fulcrum encoder and the three decoder types in Kodo [73].

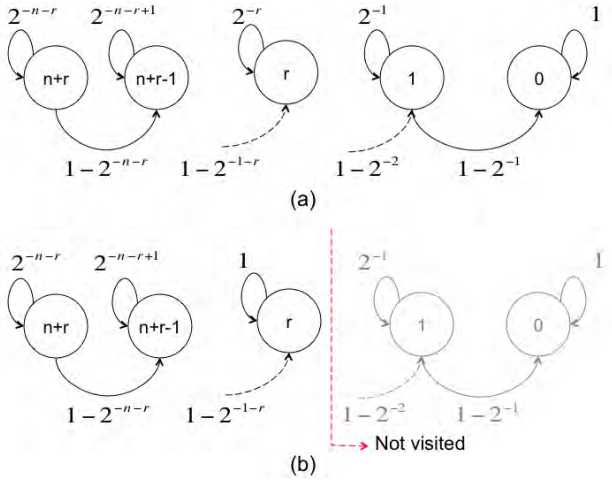


FIGURE 8. Markov chain describing the reception process at a receiver in (a) a classical $GF(2)$ network, where source, intermediate network nodes, and receivers use $GF(2)$ operations, and (b) our Fulcrum network coding framework which adds r expansion packets to a generation of n source packets, i.e., the intermediate network nodes treat expanded packets as $GF(2)$ packets, while the source uses $GF(2^h)$ operations for the expansion.

C. DELAY MODELLING: NUMBER OF REQUIRED PACKET RECEPTIONS FOR DECODING

We analyze a single receiver to understand the fundamental delay characteristics. For the analysis, let us assume that an RS code over $GF(2^s)$ with $n \geq 2^{s-1}$ is used for the outer encoding, so that receiving n independently coded packets in $GF(2)$ guarantees that the re-mapped version in $GF(2^h)$ can be decoded, by Theorem 4. That is, we assume that the MDS outer code property for a tractable analysis is met. We reuse the models for RLNC coding from [15], [44]. Fig. 8 (a) shows the Markov chain representing the process of reception of independent linear combinations over $GF(2)$ using an RLNC inner code. Each stage represents the number of missing independent linear combinations in $GF(2)$ in order to decode using only $GF(2)$ operations [15], [44]. Following the inner decoder principles, see Section II-D.1, the receiver attempts to decode in $GF(2)$ even when the source has made an expansion to $n + r$ dimensions.

Fig. 8(b) shows the process for a successful outer (and combined) decoder. In this case, the underlying $GF(2)$ process needs to only run until n independent linear combinations in $GF(2)$ are received, which are mapped back to the $GF(2^h)$ and decoded with the outer decoder. The combined decoder performs partial decoding in $GF(2)$, before attempting to use the high $GF(2^h)$ field; however, this does not affect the following analysis, only the decoding complexity. Thus, the r rightmost states, i.e., states $r-1$ through 0 are not visited, as state r became an absorbing state. If a different precoding (without the MDS outer code property) is used, there will be some probability of visiting the states below r. However, if we use a sufficiently large field size, this effect will be negligible and the process described in Fig. 8 (b) will be a very good approximation of the expected performance.

Intuitively, in Fig. 8(b), the reduction of the number of missing linearly independent combinations in $GF(2)$ from $n + r$ to $n + r - 1$ (the two leftmost states in Fig. 8(b)) corresponds to the success of a Bernoulli random experiment with success probability $1 - 2^{-n-r}$; i.e., requires on average $1/(1 - 2^{-n-r})$ trials (received packets). For r additional dimensions (expansion packets), the number of missing linear independent combinations needs to be reduced from $n + r$ down to r, i.e., the last reduction step is from $r+1$ to r. Thus, the total mean number of packets that needs to be received from the network to decode using an outer (or combined) decoder is

$$E\left[N_{GF(2)}(r)\right] = \sum_{i=1+r}^{n+r} \frac{1}{1-2^{-i}} \tag{12}$$

$$= n + \sum_{i=1+r}^{n+r} \frac{1}{2^i - 1}. \tag{13}$$

Lemma 1 shows that the overhead due to additional $GF(2)$ coded packet receptions when using an outer or combined decoder decreases exponentially with r.

Lemma 1: For an outer or a combined decoder with an MDS outer code,

$$E\left[N_{GF(2)}(r)\right] = n + 2^{-r} \times \theta(n), \tag{14}$$

for some $\theta(n) \in [1 - 2^{-n},\ 2 - 2^{-n+1}]$.

Proof: We derive an upper and a lower bound on $E\left[N_{GF(2)}(r)\right]$ described in Eq. (13). To derive the upper bound, we use the fact that $2^{i-1} \leq 2^i - 1$ for $i \geq 1$ to convert the sum in Eq. (13) to the sum of a set of elements of a geometric series. Thus,

$$E\left[N_{GF(2)}(r)\right] \leq n + \sum_{i=1+r}^{n+r} 2^{-i+1} \tag{15}$$

$$= n + 2^{-r+1} - 2^{-n-r+1}. \tag{16}$$

The lower bound follows a similar argument, but using the fact that $2^i \geq 2^i - 1$ for $i \geq 1$. Thus,

$$E\left[N_{GF(2)}(r)\right] \geq n + \sum_{i=1+r}^{n+r} 2^{-i} \tag{17}$$

$$= n + 2^{-r} - 2^{-n-r}. \tag{18}$$

□

Another interesting result for receivers with outer and combined decoders is Lemma 2 which states that the variance of $N_{GF(2)}(r)$ decreases exponentially with r.

Lemma 2: For a receiver using an outer or combined decoder with an MDS outer code,

$$V\left(N_{GF(2)}(r)\right) = O(2^{-r}). \tag{19}$$

Proof: The proof follows by bounding the variance of $N_{GF(2)}(r)$. Defining $P_i = 1 - 2^{-n-r+i-1}$ and exploiting the

independence properties of Markov chains, it is straightforward to derive

$$V\left(N_{GF(2)}(r)\right) = \sum_{i=1}^{n} \frac{1-P_i}{P_i^2}. \tag{20}$$

Elementary summation index manipulation gives $\sum_{i=1}^{n} 1/P_i = E\left[N_{GF(2)}(r)\right]$ given in Eq. (12). Thus,

$$V\left(N_{GF(2)}(r)\right) = \sum_{i=1}^{n} \frac{1}{\left(1-2^{-n-r+i-1}\right)^2} - E\left[N_{GF(2)}(r)\right] \tag{21}$$

$$\leq \sum_{i=1}^{n} \frac{1}{\left(1-2^{-r-1}\right)^2} - E\left[N_{GF(2)}(r)\right] \tag{22}$$

$$\leq \frac{n}{\left(1-2^{-r-1}\right)^2} - E\left[N_{GF(2)}(r)\right] \tag{23}$$

$$\leq \frac{n}{\left(1-2^{-r-1}\right)^2} - n \tag{24}$$

$$= \frac{n\left(2\cdot 2^{r+1}-1\right)}{\left(2^{r+1}-1\right)^2} \tag{25}$$

$$= \frac{2n\left(2^{r+1}-1\right)+n}{\left(2^{r+1}-1\right)^2} \tag{26}$$

$$= \frac{2n}{2^{r+1}-1} + \frac{n}{\left(2^{r+1}-1\right)^2} \tag{27}$$

$$\leq \frac{2n}{2^{r+1}-1} + n, \tag{28}$$

whereby (22) follows since $2^{-r-1} \geq 2^{-n-r+i-1}$ for $1 \leq i \leq n$ and (24) follows from (13). Moreover, (25) through (27) follow through elementary algebra and (28) follows by noting that $1/(2^{r+1}-1)^2 \leq 1$ for $r \geq 0$. □

D. DECODING PROBABILITY

We denote K for the number of received inner coded packets. We analyze the decoding probability $P_{\text{dec}}(K, n, r)$ at a single receiver with an outer decoder or a combined decoder.

For $K \leq n$, we can follow the analysis strategies for conventional RLNC decoding that have led to [15, Eq. (13)], [74, Eq. (2)], and [75, Eq. (7)]. Specifically, we briefly outline the analysis steps that retrace the analysis leading to [75, Eq. (7)]. For the inner encoding of $n+r$ packets (produced by the outer encoder), there are 2^{n+r} possible inner coding vectors $\{\iota_{k,j}\}_{j=1,\ldots,n+r}$ for a given inner coded packet k (whereas there are q^n possible coding vectors for the conventional $GF(q)$ RLNC considered in [75]). Similar to a conventional RLNC over n source packets and following the Markov chain model in Fig. 8(b), Fulcrum outer or combined decoding requires the receipt of n linearly independent coded packets.

Following the reasoning leading to [75, eq. (2)], we thus obtain for the Fulcrum outer or combined decoder for the case $K = n$:

$$P_{\text{dec}}(K, n, r) = \prod_{i=0}^{n-1}\left(1-\frac{2^i}{2^{n+r}}\right) = \prod_{i=1}^{n}\left(1-\frac{1}{2^{i+r}}\right). \tag{29}$$

Following the subsequent analysis steps leading to [75, eq. (7)], we obtain $P_{\text{dec}}(K, n, r) = 0$ for $K < n$ and

$$P_{\text{dec}}(K, n, r) = \prod_{i=1}^{n}\left(1-\frac{1}{2^{i+r}}\right) \quad \text{for } K = n. \tag{30}$$

We note that for $K = n$ received inner coded packets and $r = 0$ expansion packets, the Fulcrum decoding probability P_{dec} (30) is equivalent to the conventional RLNC decoding probability $P_{\text{dec}}^{\text{RLNC}}$ [75, Eq. (7)] for $K = n$. Furthermore, for $K = n$ with $r = 1$ expansion packet, the Fulcrum decoding probability

$$P_{\text{dec}} = \frac{\left(1-\frac{1}{2^{n-1}}\right)P_{\text{dec}}^{\text{RLNC}}}{1-\frac{1}{2^1}} \approx 2P_{\text{dec}}^{\text{RLNC}} \tag{31}$$

for moderate to large n. Intuitively, the one ($r = 1$) expansion packet doubles the number of possible inner coding vectors from 2^n to $2\cdot 2^n$; consequently halving the probability of linearly dependent coding vectors (which would prevent successful decoding).

For $K > n$, i.e., when the number K of received coded packets exceeds the number n of source packets in a generation, the analytical strategies from the preceding studies, e.g., [75], do not directly apply to the Fulcrum decoding. Therefore, it becomes necessary to directly analyze the numbers m_j, $m_j \geq 1$, of visits to the states $j = n+r$, n+r-1, $\ldots$, $1+r$ of the Markov chain in Fig. 8(b). The event of successful decoding with K received coded packets corresponds to the union of the mutually exclusive events where the numbers of visits to the Markov chain states sum to K, i.e., the events for which $m_{n+r}+m_{n-1+r}+\cdots+m_{1+r} = K$. We note from Fig. 8(b) that the transition from a state i, $i = n+r, n-1+r, \ldots, 1+r$, "down" to the adjacent state $i-1$ occurs with probability $1-2^{-i}$. When $m_j = 1$, then state j is visited only once, i.e., the next received coded packet transitions the Markov chain to state $j-1$. However, when $m_j > 1$, e.g., $m_j = 2$, then the next received coded packet is linearly dependent (which occurs with probability 2^{-j}) and the packet received thereafter is linearly independent (causing the transition to state $j-1$); effectively, for $m_j = 2$ there is one "extra" visit to state j. Generally, for a given m_j, there are $m_j - 1$ "extra" visits to state j (due to $m_j - 1$ received linearly dependent packets), which occur with probability $2^{-j(m_j-1)}$. Thus,

$$P_{\text{dec}}(K, n, r) = \left[\prod_{i=1+r}^{n+r}(1-2^{-i})\right] \sum_{m_{n+r}\geq 1} \cdots \sum_{m_{1+r}\geq 1} \left[\prod_{j=1+r}^{n+r} 2^{-j(m_j-1)}\right] \quad \text{for } \sum_{j=1+r}^{n+r} m_j = K, \tag{32}$$

which can be readily evaluated with standard numerical techniques.

An alternative numerical approach is to let P denote the transition probability matrix of the Markov chain in Fig. 8(b), including the absorbing state r, and let $u(k)$ denote the probability distribution vector for residing in the states $n + r$, $n + r - 1, \ldots, r$ after k received coded packets. Clearly, the starting distribution is $u(0) = [1\ 0\ 0 \ldots 0]$ since initially there is no knowledge about the data at the receiver. The probability of residing in each specific state after k transitions, i.e., k coded packet receptions, is $u(k) = u(0)P^n$. The last element of the vector $u(k)$ corresponds to the absorbing state, i.e., provides the probability of having reached the absorbing state after k transitions, which in turn corresponds to the cumulative probability of successful decoding after k transitions. The probability of reaching the absorbing state after exactly k transitions is obtained by computing $u(k) - u(k-1)$ and reading the value in the vector corresponding to the absorbing state.

We initially compare the decoding probabilities obtained from the theoretical analysis of the Fulcrum approach for an MDS outer code with measurements of our real implementation of Fulcrum for a $GF(2^8)$ systematic RLNC outer code, while the inner encoder at the source and the intermediate network nodes operates with $GF(2)$. The reported measurement results are the average of 1000 independent replications of the encoded packet transmission in the real system. The outer encoding adds $r = 1, 2$, or 4 expansion packets. Figure 9 shows the cumulative distribution function (CDF) and the probability mass function (PMF) of successful decoding of $n = 64$ original source packets after receiving $K = 64, 65, \ldots, 74$ inner $GF(2)$ coded packets. We observe from Figure 9 that the theoretical decoding probability values match the measurements from the real system quite closely. This verifies that the $GF(2^8)$ field is sufficiently large so that the real coding performance of $GF(2^8)$ RLNC closely approximates the MDS outer code property defined in Section III-A.

We proceed to more comprehensively examine the Fulcrum decoding probability based on the theoretical analysis and compare with conventional $GF(2)$ RLNC in Figure 10. We observe from Figure 10 that standard $GF(2)$ RLNC achieves only a decoding probability of approximately 0.3 when no additional coded packets have been received; five or more additional coded packets are required to decode with a probability exceeding 0.95. For standard $GF(2)$, there is no outer encoder; instead we have effectively only an inner encoder that operates in $GF(2)$ and encodes n source packets to generate $n + 10$ coded packets in $GF(2)$. Due to the small $GF(2)$ field size, the coding coefficient vectors have relatively high correlations. Specifically, there is only a relatively low probability of 0.3 that n received coded packets are all linearly independent so as to permit decoding. With $n+5$ received packets there is probability higher than 0.95 that at least n among the $n + 5$ packets have linearly independent coding coefficient vectors.

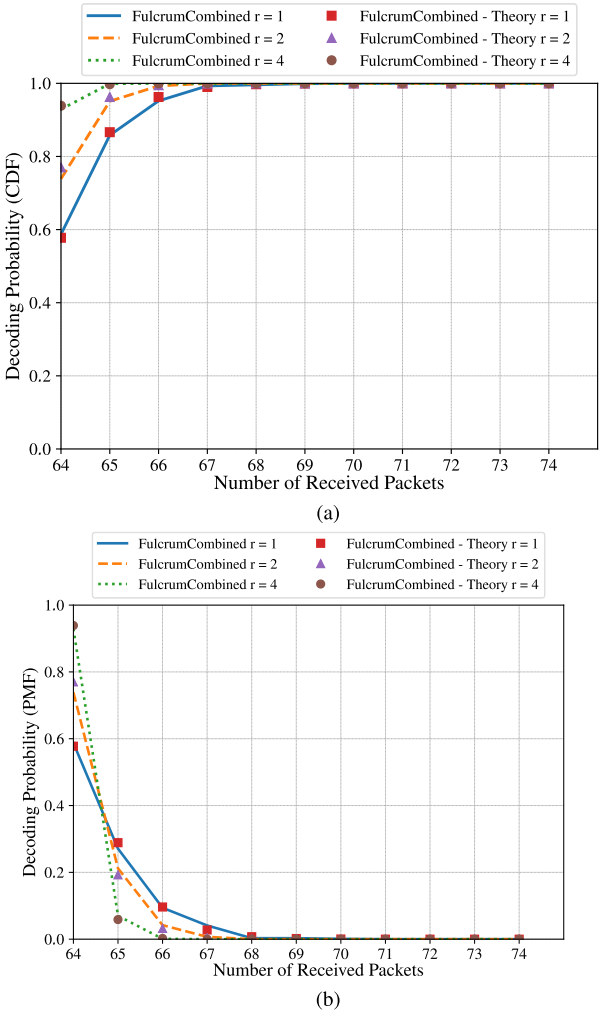


FIGURE 9. Probability of successful decoding of $n = 64$ original source packets with a Fulcrum outer or combined decoder as a function of the number K of received inner $GF(2)$ coded packets for different numbers r, $r = 1, 2, 4$, of outer encoding $GF(2^8)$ expansion packets. Comparison of theoretical analysis for MDS outer code with measurements from real implementation with $GF(2^8)$ RLNC outer code. (a) CDF. (b) PMF.

Next, we observe from Fig. 10 for $r > 0$, i.e., the outer encoder is "switched on", that increasing numbers r of expansion packets substantially increase the decoding probability. For instance, for zero additional received coded packets, the decoding probability doubles from nearly 0.3 for convectional $GF(2)$ RLNC to nearly 0.6 for Fulcrum with $r = 1$ expansion packet, as analyzed in (31). Intuitively, with r outer ($GF(2^h)$) coded expansion packets, there are $n + r$ dimensions in the inner coding (and correspondingly $n+r+1$ states in the Markov chain representing the decoding in Fig. 8). Only an arbitrary subset of n out of these $n + r$ dimensions needs to be "recovered" through the receipt of n linear independent inner coded packets; the remaining r dimensions are recovered through the mapping back to the outer code. Effectively, with r Fulcrum expansion packets, relatively fewer (namely any n out of $n + r$) packets need to be recovered through received inner $GF(2)$ coded packets versus all (n out of n) packets need to be recovered through

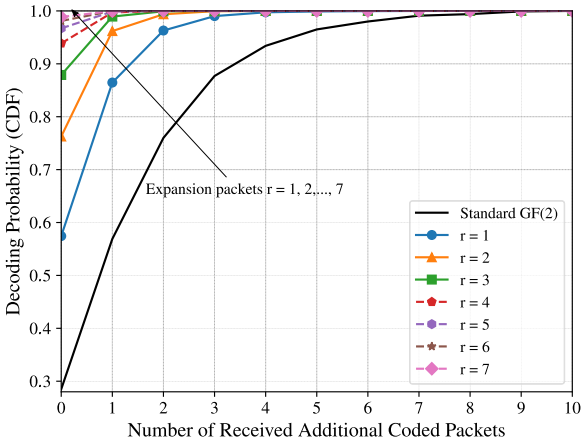


FIGURE 10. Probability of successful decoding for a Fulcrum outer or combined decoder as a function of the number of ($K - n = 0, 1, \ldots, 10$) additional inner $GF(2)$ coded packets received beyond the number n of original source packets in a generation for different numbers r, $r = 1, 2, \ldots, 7$, of outer encoding $GF(2^8)$ expansion packets. Benchmark is standard $GF(2)$ RLNC without any outer coding.

received $GF(2)$ coded packets in conventional $GF(2)$ RLNC. These additional "degrees of freedom" provided by the r expansion packets greatly increase the decoding probability. From the coding vector perspective, the r expansion packets add r bits to the inner coding vector (which becomes $\{\iota_{k,j}\}_{j=1,2,\ldots,n,n+1,\ldots,n+r}$), thus increasing the number of possible inner coding vectors from 2^n to $2^r \cdot 2^n$, which in turn decreases the probability of linear dependent coding vectors by a factor of 2^r. Thus, each additional expansion packet essentially halves the probability of linear dependent inner coding vectors.

TABLE 2. Decoding probability after reception of $n, n+1, n+2$, or $n+3$ coded packets using the outer or combined decoders for various numbers of expansion packets r and assuming RLNC $GF(2)$ inner encoder and recoders.

Code		Decoding after receiving (coded packets)			
	r	n	n + 1	n + 2	n + 3
Fulcrum	4	93.87%	99.75%	99.99%	99.9997%
	7	99.22%	99.996%	99.99998 %	99.99999992%
	10	99.90%	99.9999%	99.99999996%	99.99999999998%
RaptorQ		99%	99.99%	99.9999%	

Table 2 provides key decoding probabilities (in percentages) when receiving $n, n+1, n+2$, and $n+3$ $GF(2)$ RLNC inner coded packets. Table 2 shows that the probability of decoding after receiving exactly n coded packets using an outer or combined decoder is quite high, even for small to moderate r values. It also shows that the performance with $r = 7$ is similar to that provided by RaptorQ codes [76], while $r > 7$ can provide higher decoding guarantees.

E. DECODING PROBABILITY FOR BROADCAST TO HETEROGENEOUS RECEIVERS

This section considers broadcast from one source to two receivers (R_1 and R_2) with independent channels and packet loss probability e_i for receiver R_i. Our goal is to illustrate the effect of using different decoders at receivers with heterogeneous channel qualities as well as to compare the performance of Fulcrum to that of standard RLNC for different finite fields.

We exploit the Markov chain model presented in [15] to provide an accurate representation of the field size effect when broadcasting to two receivers. This model is also easily adapted to incorporate the use of the outer decoding capabilities of Fulcrum. The model in [15] relies on a state definition that incorporates three variables, the number of independent linear combinations at each receiver and the common linear combinations between the two. The key change in the model is similar to the change introduced in the Markov chain in Section III, that is, considering that the dimensions in the Markov chain in [15] have to be increased effectively from $n + 1$ to $n + r + 1$. Then, if one (or both) receivers use the outer (or combined) decoder, gathering n linearly independent combination will effectively correspond to gathering $n + r$ linearly independent combinations. In other words, the required number of linearly independent combinations is effectively reduced from $n + r$ (for the inner decoder) to n for the outer or combined decoder. Modifying the result from [15] and the numerical matrix approach outlined for the single receiver in Section III-D allows the computation of the probabilities of successful decoding for broadcasting, even in the presence of losses in the system.

Figure 11(a) shows the CDF for the number of required packet transmissions to complete the delivery of $n = 10$ packets to two receivers. We observe from Fig. 11(a), that the Fulcrum approach with $r = 7$ outer expansion packets and with both receivers exploiting the outer decoder requires essentially the same number of packet transmissions (i.e., completes as quickly) as $GF(2^{16})$ RLNC. We further observe that Fulcrum with $r = 2$ and two receivers with outer decoder achieves nearly the same performance as $GF(2^{16})$ RLNC; however, the $GF(2^8)$ Fulcrum outer decoder has substantially lower computational complexity than $GF(2^{16})$ RLNC.

Additionally, we observe from Fig. 11(a) that when one of the receivers employs the inner decoder, i.e., uses only $GF(2)$ operations, the Fulcrum approach with $r = 2$ expansion packets gives substantially shorter completion times than conventional $GF(2)$ RLNC. However, Fulcrum with $r = 7$ expansion packets gives lower probabilities of completing the packet delivery than conventional $GF(2)$ RLNC with 20 or less transmitted packets; while for 21 or more transmitted packets, Fulcrum with $r = 7$ achieves higher completion probabilities than $GF(2)$ RLNC. The plotted completion time is the mean of the two receivers. The receiver employing the inner decoder requires at least $n + r$ transmissions for decoding. Thus, with $r = 7$, the inner decoder requires at least 17 inner coded packets for decoding; whereas, for $r = 2$ the inner decoder requires at least 12 packets.

Figure 11(b) provides the corresponding PMF, demonstrating that the Fulcrum framework reduces the variance of the

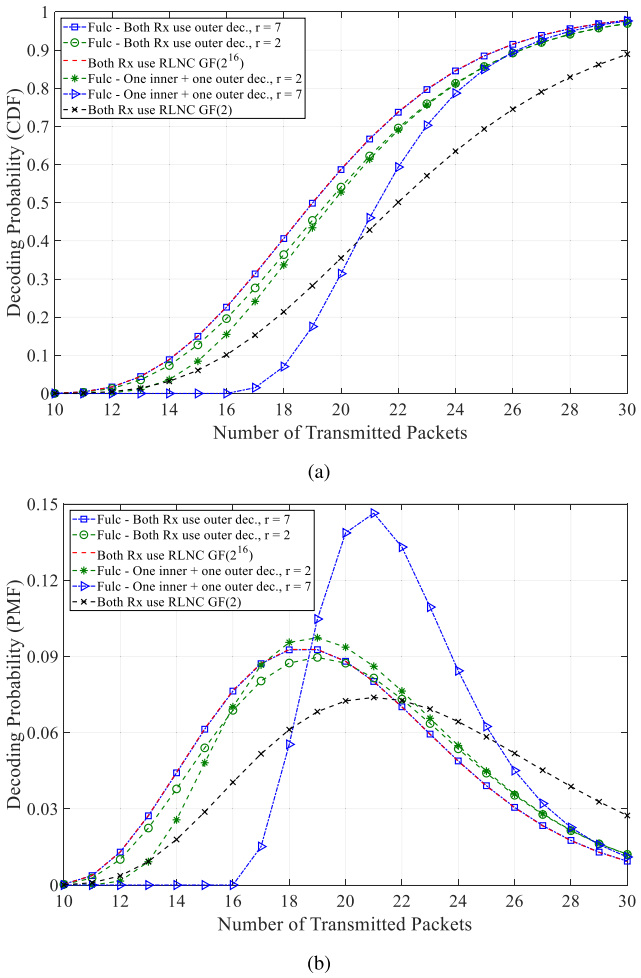


FIGURE 11. Probabilities of successful decoding for broadcast channel with two receivers showing the performance of standard RLNC for $GF(2)$ and $GF(2^{16})$, the performance of Fulcrum when one receiver exploits the outer code and the other only the inner code, and Fulcrum when both receivers exploit the outer code. Parameters $e_1 = 0.1$, $e_2 = 0.5$, $n = 10$ packets, $GF(2^8)$ RLNC Fulcrum outer encoding. (a) CDF. (b) PMF.

number of required packet transmissions, particularly when increasing the number r of expansion packets. The variance reduction is particularly pronounced for Fulcrum with one inner and one outer decoder for $r = 7$ compared to conventional $GF(2)$ RLNC. This numerical result illustrates the theoretical result in Lemma 2.

Overall, the results in Figure 11(b) indicate that Fulcrum coding with a small to moderate number r of outer coded expansion packets appears well suited for networks with heterogenous receivers. For the specific example considered in Figure 11(b), $r = 2$ expansion packets strike a good compromise between reasonably high decoding probabilities and requiring only few (r) additional packet receptions beyond the number n of source packets to enable decoding.

F. OVERHEAD

We define the *overhead* for a generation consisting of n source packets as the number of additional bits transmitted to successfully deliver the n packets to the receiver. The overhead includes the coding coefficients, i.e., the additional header information, and the overhead caused by retransmissions due to linearly dependent packets (while neglecting retransmissions due to channel losses). We consider the standard coding vector representation, i.e., a coefficient per packet is sent attached to the coded packet. We analyze receivers with outer and combined decoders.

For sufficiently large h with negligible probabilities of linearly dependent coding vectors, the overhead of standard $GF(2^h)$ RLNC is proportional to hn^2 bits [44], [62].

The mean Fulcrum overhead due to coding coefficients with a standard coding vector representation is proportional to

$$(n+r)E\left[N_{GF(2)}(r)\right] \leq (n+r)\left(n + 2^{-r+1} - 2^{-n-r+1}\right), \tag{33}$$

whereby $E\left[N_{GF(2)}(r)\right]$ was upper bounded by (16). Typically, $r \ll n$; thus, the coding coefficient overhead will be dominated by n^2. This Fulcrum coding coefficient overhead of n^2 bits for a generation of n source packets is by a factor of h smaller than the $GF(2^h)$ RLNC overhead. Regarding the impact of the number r of expansion packets, note that the coding coefficient vector overhead in Eqn. (33) grows with r.

The mean Fulcrum overhead due to additional $GF(2)$ packet receptions needed due to linearly dependent coding vectors for data packets of length L bits is

$$\left(E\left[N_{GF(2)}(r)\right] - n\right)L \leq \left(2^{-r+1} - 2^{-n-r+1}\right)L. \tag{34}$$

In particular, Lemma 1 showed that this mean overhead due to additional $GF(2)$ packet receptions needed due to linearly dependent coding vectors decreases exponentially with r. Importantly, the numerical evaluations of the decoding probability in Fig. 10 and Table 2 demonstrate that r values in the range 7 to 10, which keep the coding coefficient overhead in Eqn. (33) reasonably small, achieve very high decoding probabilities without or only very few additional packet receptions and should be sufficient for a wide range of practical applications.

G. ENCODING AND DECODING THROUGHPUT MEASUREMENTS

This section presents Fulcrum encoding and decoding throughput results obtained with the measurement setup in Section III-B. The benchmark RLNC encoders and decoders in $GF(2)$ ("Binary" in the Figures) and $GF(2^8)$ ("Binary8" in the Figures) use the standard Kodo RLNC implementations with and without Single Instruction Multiple Data (SIMD) operations for hardware speed up. The $GF(2)$ RLNC benchmark represents the fastest dense code. The $GF(2^8)$ RLNC benchmark represents a commonly used dense code with the same field size as the Fulcrum outer code; whereby, $GF(2^8)$ RLNC decoding probabilities approach one when n packets have been received.

Figure 12(a) shows the Fulcrum decoding throughput for different numbers r of expansion packets and different

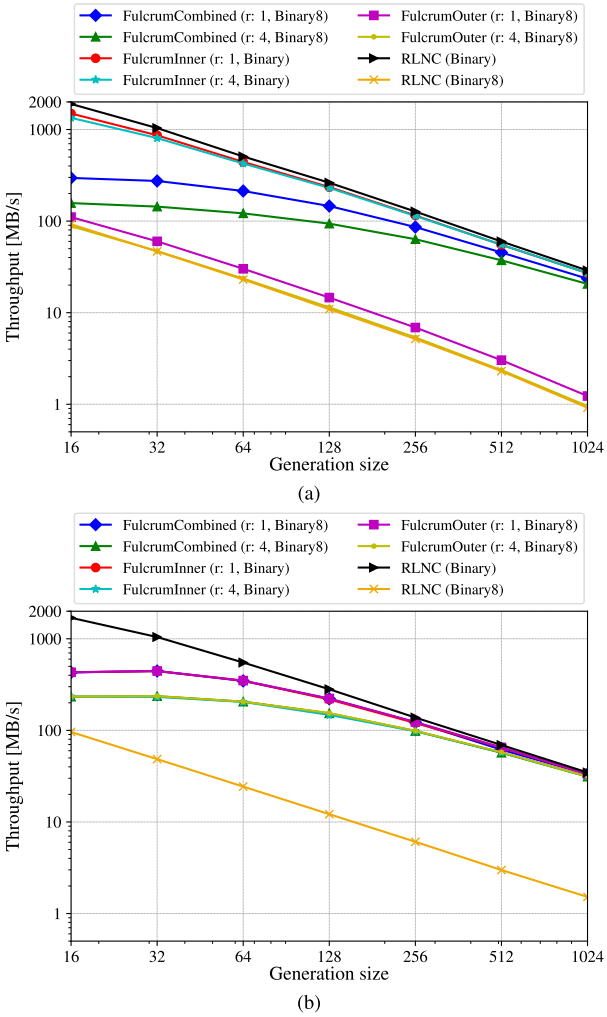


FIGURE 12. Processing speed (throughput) of an i7 without SIMD optimizations for decoding and encoding of Fulcrum compared to RLNC decoding and encoding as a function of generation size n. The encoding speed of Fulcrum does not depend on the decoder type (combined, inner, or outer). (a) Decoding. (b) Encoding.

decoder types. When only the inner code over $GF(2)$ is utilized for decoding in Fulcrum (inner decoder), Fulcrum is similar to RLNC over $GF(2)$. When only the outer code over $GF(2^8)$ is utilized in Fulcrum (outer decoder), Fulcrum becomes similar to $GF(2^8)$ RLNC. Thus, in these two cases the decoding throughput for Fulcrum is expected to be equivalent to $GF(2)$ RLNC and $GF(2^8)$ RLNC, respectively. The measurement results in Fig. 12(a) confirm that the decoding implementation performs as expected in these two known cases. The throughput results for the combined decoder show gains compared to $GF(2^8)$ RLNC. Not only is the Fulcrum combined decoder always faster than $GF(2^8)$ RLNC, but the throughput also approaches that of $GF(2)$ RLNC as the generation size n grows. For $n = 1024$ packets, the combined decoder is 20 times faster than $GF(2^8)$ RLNC while achieving similar decoding probabilities, see Fig. 9a.

We observe from Figure 12(a) that for the outer code with $r = 1$, the throughput of the Fulcrum outer decoder is higher than for the standard $GF(2^8)$ RLNC decoder. The reason is that each inner coded packet has a probability of $1/2$ to have a contribution of the expansion packet, i.e., a packet with high field $GF(2^8)$ coefficients different from zero or one (namely when the inner coding coefficient $\iota_{k,n+1} = 1$). Thus, when mapping back in the outer decoder, roughly half of the rows will have only zeros and ones while the other half will have other elements in $GF(2^8)$. The rows with only zeros and ones require fewer multiplication operations in the decoding process than rows with high field coefficients, speeding up the processing. This speed-up could be more pronounced than depicted in Figure 12(a) if we tried to exploit this particular structure. However, the probability of obtaining rows with only zeros and ones after mapping back decreases dramatically as r increases, as indicated by the throughput results for $r = 4$.

Figure 12(a) shows that the combined decoder has some performance dependence for small generation size n for different numbers r of expansion packets, namely, the higher the r, the lower the throughput, (whereby the combined decoder has still higher throughput than the outer decoder). However, this difference in performance becomes negligible as the number n of data packets per generation increases. For increasing n, most of the processing effort will be spent decoding in the inner code, and the effect of the r expansion packets diminishes.

Decoding throughput is usually given a higher priority than the encoding speed, e.g., if there are more decoders than encoders, or because the decoding process tends to be slower than the encoding process. However, encoding speed can be critical in some cases, e.g., a satellite transmitting to an earth station, or sensor nodes collecting and sending data to a base station, because there is an inherent constraint on the sender's computational capabilities or energy. Figure 12(b) shows the encoding speed compared to the baseline (full vector [62]) $GF(2)$ and $GF(2^8)$ RLNC. For the case of $n = 16$ packets in a generation, the Fulcrum encoder runs 6.6 times and 3.2 times faster for $r = 1$ and $r = 4$, respectively, compared to the $GF(2^8)$ RLNC encoder. As the generation size n increases, so does the gain over $GF(2^8)$ RLNC, while the impact of the number r of expansion packets decreases. For example, for $n = 128$ packets, the Fulcrum encoder is approximately 14 times faster than the $GF(2^8)$ RLNC encoder, and for $n = 256$ the encoding speed is close to $GF(2)$ RLNC.

Figure 13 studies the effect of SIMD instructions on the Fulcrum decoding throughput on the i7 processor. The SIMD optimization only speeds up $GF(2^8)$ operations, which means that the throughput of the $GF(2)$ RLNC decoder and the inner decoder remain unchanged from Figure 12(a). Figure 13 shows similar trends as Figure 12(a), but with reduced gains with respect to $GF(2^8)$ RLNC as SIMD greatly speeds up the $GF(2^8)$ RLNC operations. Nonetheless, for $n = 128$ the Fulcrum decoding throughput is 2.8 times higher than $GF(2^8)$ RLNC and close to the $GF(2)$ RLNC throughput.

Finally, Figures 14(a) and 14(b) show the performance of the combined decoder on various commercial devices without

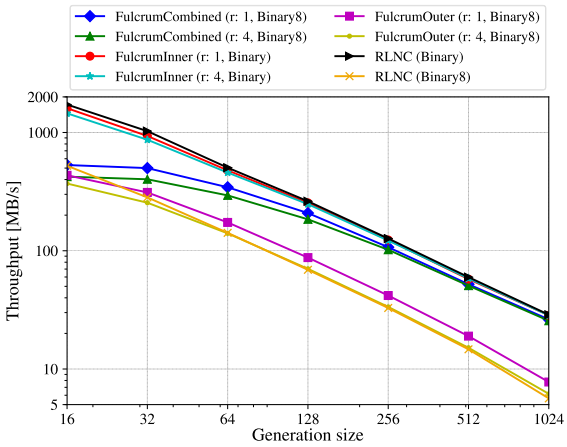
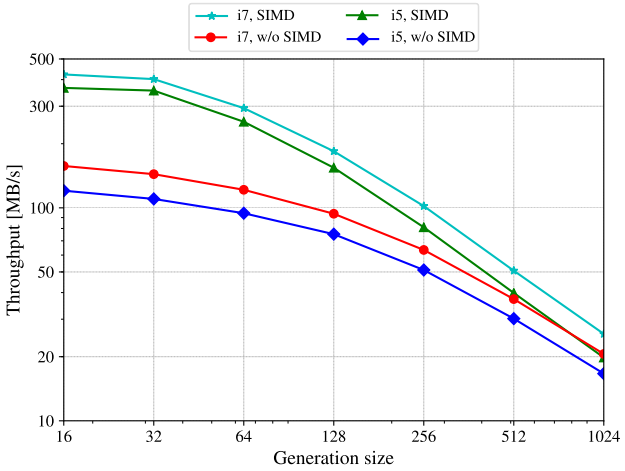
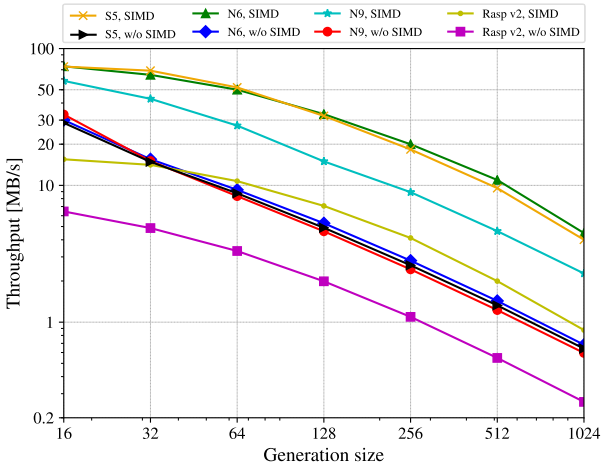


FIGURE 13. Decoding processing speed (throughput) of i7 with SIMD optimization for Fulcrum and conventional RLNC as a function of generation size n.



(a) i5 and i7 PC processors



(b) Mobile device processors

FIGURE 14. Processing speed (throughput) of combined decoder (a) without SIMD and (b) with SIMD for different devices.

and with SIMD optimizations, respectively. SIMD achieves two- to four-fold speed ups on all devices for small generation sizes n. We observe that the benefit of SIMD is slightly more pronounced for mobile devices at high n (up to five times the speed-up), while for desktops the gains over no SIMD have essentially disappeared. This does not mean that SIMD in desktops is not improving computation of $GF(2^8)$, but rather that the main limitation becomes the processing of $GF(2)$ operations at high speeds. Importantly, mobile devices, which are typically energy and computationally limited, can significantly benefit from the combination of Fulcrum and SIMD instructions over a wide range of n.

IV. CONCLUSIONS

This paper has presented the Fulcrum network coding framework, an advanced network coding structure that preserves RLNC's ability to recode seamlessly in the network while providing key mechanisms for practical deployment of network coding. Fulcrum addresses several of the standing practical problems with existing RLNC codes and rateless codes, by employing a concatenated code design. This concatenated code design is highly flexible, tunable, and intuitive. This paper has described the design of Fulcrum network codes and their practical benefits over previous network coding designs. The paper has also provided mathematical analyses of the performance of Fulcrum network codes. The paper has also presented a first implementation of Fulcrum in the Kodo C++ network coding library as well as benchmarking results for the Fulcrum encoding and decoding throughput on a wide range of computational devices.

Our evaluations demonstrate that Fulcrum achieves much higher encoding and decoding processing speeds (throughput) than conventional $GF(2^8)$ RLNC. In fact, the Fulcrum processing speeds approach those of $GF(2)$ RLNC as the generation size grows. Importantly, Fulcrum maintains the high decoding probabilities of $GF(2^8)$ RLNC, while increasing the processing speed by up to a factor of 20 in some scenarios with our initial implementation. Furthermore, in Fulcrum, the trade-off between coding processing speed and decoding probability can be easily adjusted through the number r of outer code expansion packets to meet the requirements of a given application.

Fulcrum solves several long-standing problems for existing RLNC codes. First, Fulcrum enables an adjustable trade-off between coding throughput and decoding probability. Second, it provides a higher coding processing speed when compared to the existing RLNC codes. Third, it reduces the overhead associated with the coding vector representation necessary for recoding, while maintaining a high decoding probability. Fourth, it reduces the type of operations and logic that the network needs to support while allowing end-to-end devices to tailor their desired service and performance, making a key step to widely deploying network coding in practice. This has an added advantage of allowing the network to support future designs seamlessly and naturally providing backwards compatibility.

There are several interesting directions for future research on the Fulcrum coding framework. One future work direction is to study optimal solutions to use the Fulcrum

structure to spread computation complexity over the network. Another interesting future research direction is security; more specifically, Fulcrum provides a simple way to implement some of the ideas in Secure Practical Network Coding (SPOC) [77]. With Fulcrum, the mapping of the outer decoder constitutes the secret key (or part of it) that the source and destinations share and that, in contrast to [77], does not need to be sent over the network along with the coded packets. Using Fulcrum, we would not incur the large SPOC overhead, which consists of two coding coefficients per original packet (one encrypted, one without encryption). In fact, the end points (source and receivers) can choose very large field sizes in the outer code while maintaining $1 + r/n$ bits per packet in the generation as overhead. Fulcrum can also provide security without the need to run Gaussian elimination twice at the time of decoding [78]. As a consequence Fulcrum does not need to trade off field size and generation size (and thus security) for overhead in the network and complexity.

APPENDIX
REED-SOLOMON OUTER CODE: PROOF OF FULL RANK PROPERTY OF REMAPPED PACKETS

We have B, an $m \times (n+r)$-matrix over $\mathbb{F}_2$, and G, an $(n+r) \times n$ matrix over $\mathbb{F}_{2^s} = \mathbb{F}_q$. Suppose that G is the generator matrix of an RS code with length $n + r \leq q - 1 = 2^s - 1$. The value of m is related to the length of the incoming messages, e.g., if it is a single $\mathbb{F}_{2^s}$ symbol, then $m = s$ bits. We remark that vectors are column vectors and that we multiply on the right.

An RS code C, with dimension n, can be defined as the vector space generated by the evaluation of the monomials $1, X, \ldots, X^{n-1}$ at the points $\mathbb{F}_q \setminus \{0\}$. Namely, let α be a primitive element of $\mathbb{F}_q$ and let

$$\text{ev} : \mathbb{F}_q[X] \to \mathbb{F}_q^{n+r}, \tag{35}$$

$$\text{with ev}(f) = (f(\alpha^0), f(\alpha^1), \ldots, f(\alpha^{q-2}). \tag{36}$$

Thus, $C = \langle\{\text{ev}(X^i) : i = 0, \ldots, n-1\}\rangle$. A generator matrix G is given by considering as columns the evaluation of a monomial at $\mathbb{F}_q \setminus \{0\}$. The dual code of an RS code is given by Lemma 3.

Lemma 3 (Lemma 5.3.1 [79]): Let C be a Reed-Solomon code with dimension n, then the dual code of C is given by

$$C^\perp = \langle\{ev(X^1), \ldots, ev(X^{q-1-n})\}\rangle. \tag{37}$$

We consider the $m \times n$-matrix BG and denote the associated linear function by φ. We assume that B and G have full rank and ask whether $\dim(\varphi(V)) < \dim V$ for a vector subspace $V \subseteq \mathbb{F}_q^n$. Since φ is a linear function, the dimension of the image plus the dimension of the kernel is equal to the dimension of the original space. Therefore, we ask whether $\dim(\ker(\varphi)) > 0$.

In order to prove the main result, we shall introduce the cyclotomic coset containing a in $\mathbb{F}_q = \mathbb{F}_{2^s}$, $I_a = \{a, 2a \bmod q - 1, 2^2\ a \bmod q - 1, \ldots, 2^{s-1}a \bmod q - 1\}$. For instance, for $q = 2^4$, the different cyclotomic cosets are $I_0 = \{0\}, I_1 = \{1, 2, 4, 8\}, I_3 = \{3, 6, 12, 9\}, I_5 = \{5, 10\}$, and $I_7 = \{7, 14, 13, 11\}$. One has that $I_1 = I_2 = I_4 = I_8$, but usually one denotes the coset by the smallest number. We can now characterize when $\dim(\ker(\varphi)) = 0$.

Theorem 4: Let $C \subseteq \mathbb{F}_{2^s} = \mathbb{F}_q$ be a Reed-Solomon code with dimension n and the linear map φ defined above. Then, $\dim(\ker(\varphi)) = 0$ *if and only if* $n \geq 2^{s-1}$.

Proof: The linear function φ is the composition of two linear maps, namely the maps associated with G and B. G is a generator matrix; therefore, G is injective. Hence, $BGx = 0$ if and only if $c = Gx \in \ker(B)$. That is, $\dim(\ker(\varphi)) > 0$ if the rows of B are orthogonal to c, which is a word of C. Therefore, the rows of B are words in the dual code of C.

By Lemma 3, the dual code of C is given by $C^\perp = \langle\{\text{ev}(X^1), \ldots, \text{ev}(X^{q-1-n})\}\rangle$. Thus, $C^\perp \subseteq \mathbb{F}_q^{n+r}$, but the rows of B are over $\mathbb{F}_2$. Hence, we should consider the subfield subcode of $C^\perp$, i.e.,

$$\text{SubfSubc}_2(C^\perp) = \{c \in \mathbb{F}_2^{n+r} : c \in C^\perp\}. \tag{38}$$

The columns of the generator matrix of $\text{SubfSubc}_2(C^\perp)$ are the rows of B which reduce the dimension of the image. By [80] and [81, Th. III.8],

$$\dim(\text{SubfSubc}_2(C^\perp)) = \#\{I_j : I_j \subseteq \{1, \ldots, q-1-n\}\}. \tag{39}$$

That is, we shall only consider exponents that are in a cyclotomic coset that is contained in $\{1, \ldots, q-1-n\}$. Clearly, $I_0 \not\subseteq \{1, \ldots, q-1-n\}$. Let $k \geq 2^{s-1}$, then $q-1-n < 2^{s-1}$; therefore, the cyclotomic coset $I_1 = \{1, 2, 2^2, \ldots, 2^{s-1}\}$ is not contained in $\{1, \ldots, q-1-n\}$. Finally, let $j > 2^{s-1}$, then $I_j \neq I_0, I_1$, and $I_j \not\subseteq \{1, \ldots, q-1-n\}$ since $j \in I_j$ and $q-1-n < 2^{s-1}$. Therefore, $\text{SubfSubc}_2(C^\perp) = \{0\}$ and $\dim(\ker(\varphi)) = 0$.

Let us now consider $n < 2^{s-1}$, then $I_1 = \{1, 2, 2^2, \ldots, 2^{s-1}\}$ is contained in $\{1, \ldots, q-1-n\}$ and $\dim(\text{SubfSubc}_2(C^\perp)) \geq s$. Thus, $\dim(\ker(\varphi)) > 0$. □

Explicit generators of $\text{SubfSubc}_2(C^\perp)$ to identify cases of linear dependence for $n < 2^{s-1}$ can be obtained by using results in [80], [81, Theorem III.8] as follows.

Theorem 5: Let C be a Reed-Solomon code with dimension n, then the dimension of $\text{SubfSubc}_2(C^\perp)$ is $\sum_{I_a \subseteq \{1,\ldots,q-1-n\}} \#I_a$, and a basis is given by $\{ev(f_{I_a,\beta^j}) : j \in \{0, \ldots, \#I_a - 1\}\}$, where

$$\beta = \alpha^{(2^s-1)/(2^{\#I_a}-1)}, \tag{40}$$

i.e., a primitive element of $\mathbb{F}_{2^{\#I_a}} \subseteq \mathbb{F}_{2^s}$, and

$$f_{I_a,\beta} = \beta X^a + \beta^2 X^{2a} + \cdots + \beta^{2^{\#I_a-1}} X^{2^{\#I_a-1}a}. \tag{41}$$

As an example, let C_8 be the Reed-Solomon code with dimension $n = 2^3$ in $\mathbb{F}_{2^4}^{15}$. We have that $C_8 = \langle\{\text{ev}(X^0), \ldots, \text{ev}(X^7)\}\rangle$, $C_8^\perp = \langle\{\text{ev}(X^1), \ldots, \text{ev}(X^7)\}\rangle$ and $\text{SubfSubc}_2(C_8^\perp) = \{0\}$ because $I_a \not\subseteq \{1, \ldots, 7\}$ for any a and $\dim(\ker(\varphi)) = 0$. Consider another example with a Reed-Solomon code C_7 with dimension $n = 7$ in $\mathbb{F}_{2^4}^{15}$. We have that

$C_7 = \langle\{\text{ev}(X^0), \ldots, \text{ev}(X^6)\}\rangle$, $C_7^\perp = \langle\{\text{ev}(X^1), \ldots, \text{ev}(X^8)\}\rangle$ and $\dim(\text{SubfSubc}_2(C_7^\perp)) = 4$ because $I_1 = \{1, 2, 4, 8\} \subseteq \{1, \ldots, 8\}$. A basis for $\text{SubfSubc}_2(C_7^\perp)$ is given by $\{\text{ev}(f_{I_1,1}),\ \text{ev}(f_{I_1,\alpha}),\ \text{ev}(f_{I_1,\alpha^2}),\ \text{ev}(f_{I_1,\alpha^3})\}$, that is

$$\begin{aligned}\{&\text{ev}(X + X^2 + X^4 + X^8),\\ &\text{ev}(\alpha X + \alpha^2 X^2 + \alpha^4 X^4 + \alpha^8 X^8),\\ &\text{ev}(\alpha^2 X + \alpha^4 X^2 + \alpha^8 X^4 + \alpha X^8),\\ &\text{ev}(\alpha^3 X + \alpha^6 X^2 + \alpha^{12} X^4 + \alpha^9 X^8)\}.\end{aligned} \tag{42}$$

Therefore, a generator matrix for $\text{SubfSubc}_2(C_7^\perp)$ is

$$\begin{bmatrix} 0&0&0&1&0&0&1&1&0&1&0&1&1&1&1\\ 0&0&1&0&0&1&1&0&1&0&1&1&1&1&0\\ 0&1&0&0&1&1&0&1&0&1&1&1&1&0&0\\ 1&0&1&1&1&0&1&0&1&1&1&1&0&0&0 \end{bmatrix}^T. \tag{43}$$

Future work shall consider exploiting such generators to improve the efficiency of the decoder in these corner cases. In order to consider β as a primitive element of $\mathbb{F}_{2^{\#I_a}}$ in $\mathbb{F}_{2^s}$, one may consider to use Conway polynomials for defining finite fields [82].

ACKNOWLEDGMENT

This paper was presented at the Proceedings of the IEEE International Symposium on Wireless Communications Systems, 2014 [1].

REFERENCES

[1] D. E. Lucani, M. V. Pedersen, J. Heide, and F. H. P. Fitzek, "Coping with the upcoming heterogeneity in 5G communications and storage using fulcrum network codes," in *Proc. 11th IEEE Int. Symp. Wireless Commun. Syst. (ISWCS)*, Aug. 2014, pp. 997–1001.

[2] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.

[3] P. Elias, A. Feinstein, and C. Shannon, "A note on the maximum flow through a network," *IRE Trans. Inf. Theory*, vol. 2, no. 4, pp. 117–119, Dec. 1956.

[4] 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.

[5] 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.

[6] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf. Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003.

[7] T. Ho and 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, no. 10, pp. 4413–4430, Oct. 2006.

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

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] M. Nistor, D. E. Lucani, T. T. V. Vinhoza, R. A. Costa, and J. Barros, "On the delay distribution of random linear network coding," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 5, pp. 1084–1093, May 2011.

[16] 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.

[17] X. Tang, J. Zhou, S. Xiong, J. Wang, and K. Zhou, "Geographic segmented opportunistic routing in cognitive radio ad hoc networks using network coding," *IEEE Access*, vol. 6, pp. 62766–62783, 2018.

[18] 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.

[19] Q. Wang, X. Cheng, Q. Wang, P. Liu, and B. Deng, "A delay-constrained network coding algorithm based on power control in wireless networks," *IEEE Access*, vol. 6, pp. 62224–62233, 2018.

[20] 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.

[21] 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.

[22] F. Zhao and M. Médard, "On analyzing and improving COPE performance," in *Proc. Inf. Theory Appl. Workshop (ITA)*, San Diego, CA, USA, Jan./Feb. 2010, pp. 1–6.

[23] B. W. Chen, W. Ji, F. Jiang, and S. Rho, "QoE-enabled big video streaming for large-scale heterogeneous clients and networks in smart cities," *IEEE Access*, vol. 4, pp. 97–107, 2016.

[24] M. S. Karim, M. Esmaeilzadeh, and P. Sadeghi, "On reducing intercept probability for unsubscribed video layers using network coding," *IEEE Commun. Lett.*, vol. 21, no. 6, pp. 1385–1388, Jun. 2017.

[25] K. Matsuzono, H. Asaeda, and T. Turletti, "Low latency low loss streaming using in-network coding and caching," in *Proc. IEEE INFOCOM*, May 2017, pp. 1–9.

[26] M. Rekab-Eslami, M. Esmaeili, and T. A. Gulliver, "Multicast convolutional network codes via local encoding kernels," *IEEE Access*, vol. 5, pp. 6464–6470, 2017.

[27] H. Seferoglu and A. Markopoulou, "Video-aware opportunistic network coding over wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 5, pp. 713–728, Jun. 2009.

[28] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proc. IEEE*, vol. 99, no. 3, pp. 476–489, Mar. 2011.

[29] Y. Gao, X. Xu, Y. L. Guan, and P. H. J. Chong, "V2X content distribution based on batched network coding with distributed scheduling," *IEEE Access*, vol. 6, pp. 59449–59461, 2018.

[30] G. Joshi, Y. Liu, and E. Soljanin, "On the delay-storage trade-off in content download from coded distributed storage systems," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 989–997, May 2014.

[31] N. Kumar, S. Zeadally, and J. J. P. C. Rodrigues, "QoS-aware hierarchical Web caching scheme for online video streaming applications in Internet-based vehicular ad hoc networks," *IEEE Trans. Ind. Electron.*, vol. 62, no. 12, pp. 7892–7900, Dec. 2015.

[32] M. Sipos, J. Gahm, N. Venkat, and D. Oran, "Erasure coded storage on a changing network: The untold story," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2016, pp. 1–6.

[33] L. Wang, H. Wu, and Z. Han, "Wireless distributed storage in socially enabled D2D communications," *IEEE Access*, vol. 4, pp. 1971–1984, 2017.

[34] C. Gkantsidis, J. Miller, and P. Rodriguez, "Comprehensive view of a live network coding P2P system," in *Proc. 6th ACM SIGCOMM Conf. Internet Meas.*, Rio de Janeriro, Brazil, 2006, pp. 177–188.

[35] U. Lee, J. S. Park, S. H. Lee, W. W. Ro, G. Pau, and M. Gerla, "Efficient peer-to-peer file sharing using network coding in MANET," *J. Commun. Netw.*, vol. 10, no. 4, pp. 422–429, Dec. 2008.

[36] B. Li and D. Niu, "Random network coding in peer-to-peer networks: From theory to practice," *Proc. IEEE*, vol. 99, no. 3, pp. 513–523, Mar. 2011.
[37] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: Practical wireless network coding," *IEEE/ACM Trans. Netw.*, vol. 16, no. 3, pp. 497–510, Jun. 2006.
[38] M. V. Pedersen, J. Heide, F. H. P. Fitzek, and T. Larsen, "PictureViewer—A mobile application using network coding," in *Proc. Eur. Wireless Conf. (EW)*, May 2009, pp. 151–156.
[39] P. Vingelmann, F. H. P. Fitzek, M. V. Pedersen, J. Heide, and H. Charaf, "Synchronized multimedia streaming on the iphone platform with network coding," *IEEE Commun. Mag.*, vol. 49, no. 6, pp. 126–132, Jun. 2011.
[40] J. Acevedo *et al.*, "Hardware acceleration for RLNC: A case study based on the Xtensa processor with the Tensilica instruction-set extension," *Electron.*, vol. 7, no. 9, pp. 180.1–180.22, 2018.
[41] 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.
[42] 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.
[43] J. Heide, M. V. Pedersen, F. H. P. Fitzek, and M. Médard, "On code parameters and coding vector representation for practical RLNC," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2011, pp. 1–5.
[44] D. E. Lucani, M. Médard, and M. Stojanovic, "Random linear network coding for time-division duplexing: Field size considerations," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Nov./Dec. 2009, pp. 1–6.
[45] 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 Int. Conf. Commun. (ICC) Workshops*, Jun. 2009, pp. 1–6.
[46] A. Paramanathan, M. V. Pedersen, D. E. Lucani, F. H. P. Fitzek, and M. Katz, "Lean and mean: Network coding for commercial devices," *IEEE Wireless Commun.*, vol. 20, no. 5, pp. 54–61, Oct. 2013.
[47] M. Nistor, D. E. Lucani, and J. Barros, "A total energy approach to protocol design in coded wireless sensor networks," in *Proc. Int. Symp. Netw. Coding (NetCod)*, Jun. 2012, pp. 31–36.
[48] C. W. Sørensen, A. Paramanathan, J. A. Cabrera, M. V. Pedersen, D. E. Lucani, and F. H. P. Fitzek, "Leaner and meaner: Network coding in SIMD enabled commercial devices," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Apr. 2016, pp. 1–6.
[49] N. J. H. Marcano, C. W. Sørensen, J. A. G. Cabrera, S. Wunderlich, D. E. Lucani, and F. H. P. Fitzek, "On goodput and energy measurements of network coding schemes in the Raspberry Pi," *Electron.*, vol. 5, no. 4, pp. 66.1–66.27, 2016.
[50] D. G. Forney, *Concatenated Codes*. Cambridge, MA, USA: MIT Press, 1966.
[51] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2551–2567, Jun. 2006.
[52] M. Halloush and H. Radha, "Network coding with multi-generation mixing: A generalized framework for practical network coding," *IEEE Trans. Wireless Commun.*, vol. 10, no. 2, pp. 466–473, Feb. 2011.
[53] J. P. Thibault, W. Y. Chan, and S. Yousefi, "A family of concatenated network codes for improved performance with generations," *J. Commun. Netw.*, vol. 10, no. 4, pp. 384–395, Dec. 2008.
[54] S. W. Kim, "Concatenated network coding for large-scale multi-hop wireless networks," in *Proc. IEEE Wireless Commun. Netwo. Conf. (WCNC)*, Mar. 2007, pp. 985–989.
[55] Y. Yin, R. Pyndiah, and K. Amis, "Performance of random linear network codes concatenated with Reed–Solomon codes using turbo decoding," in *Proc. 6th Int. Symp. Turbo Codes Iterative Inf. Process. (ISTC)*, Sep. 2010, pp. 132–136.
[56] V. Skachek, O. Milenkovic, and A. Nedić, "Hybrid noncoherent network coding," *IEEE Trans. Inf. Theory*, vol. 59, no. 6, pp. 3317–3331, Jun. 2013.
[57] N. Thomos and P. Frossard, "Toward one symbol network coding vectors," *IEEE Commun. Lett.*, vol. 16, no. 11, pp. 1860–1863, Nov. 2012.
[58] Steinwurf ApS. (2012). *Kodo Git Repository on GitHub*. Accessed: Oct. 27, 2018. [Online]. Available: http://github.com/steinwurf/kodo-c
[59] F. Gabriel, S. Wunderlich, S. Pandi, F. H. P. Fitzek, and M. Reisslein, "Caterpillar RLNC with feedback (CRLNC-FB): Reducing delay in selective repeat ARQ through coding," *IEEE Access*, vol. 6, p. 44787–44802, 2018.
[60] 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.
[61] 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.
[62] 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.
[63] 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.
[64] J. Heide, M. V. Pedersen, F. H. P. Fitzek, and M. Médard, "A perpetual code for network coding," in *Proc. IEEE 79th Veh. Technol. Conf. (VTC)*, May 2014, pp. 1–6.
[65] S. Feizi, D. E. Lucani, and M. Médard, "Tunable sparse network coding," in *Proc. Int. Zurich Seminar Commun. (IZS)*, Feb. 2012, pp. 107–110.
[66] R. Prior, D. E. Lucani, Y. Phulpin, M. Nistor, and J. Barros, "Network coding protocols for smart grid communications," *IEEE Trans. Smart Grid*, vol. 5, no. 3, pp. 1523–1531, May 2014.
[67] R. C. Singleton, "Maximum distance q-nary codes," *IEEE Trans. Inf. Theory*, vol. IT-10, no. 2, pp. 116–118, Apr. 1964.
[68] H. Balli, X. Yan, and Z. Zhang, "On randomized linear network codes and their error correction capabilities," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3148–3160, Jul. 2009.
[69] X. Guang, F.-W. Fu, and Z. Zhang, "Construction of network error correction codes in packet networks," *IEEE Trans. Inf. Theory*, vol. 59, no. 2, pp. 1030–1047, Feb. 2013.
[70] R. Koetter and F. R. Kschischang, "Coding for errors and erasures in random network coding," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3579–3591, Aug. 2008.
[71] K. Prasad and B. S. Rajan, "Network-error correcting codes using small fields," *IEEE Trans. Commun.*, vol. 62, no. 2, pp. 423–433, Feb. 2014.
[72] D. Silva, F. R. Kschischang, and R. Koetter, "A rank-metric approach to error control in random network coding," *IEEE Trans. Inf. Theory*, vol. 54, no. 9, pp. 3951–3967, Sep. 2008.
[73] M. V. Pedersen, J. Heide, and F. H. P. Fitzek, "Kodo: An open and research oriented network coding library," in *Proc. Int. Conf. Res. Netw.*, in Lecture Notes in Computer Science, vol. 6827. 2011, pp. 145–152.
[74] I. Chatzigeorgiou and A. Tassi, "Decoding delay performance of random linear network coding for broadcast," *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 7050–7060, Aug. 2017.
[75] O. Trullols-Cruces, J. M. Barcelo-Ordinas, and M. Fiore, "Exact decoding probability under random linear network coding," *IEEE Commun. Lett.*, vol. 15, no. 1, pp. 67–69, Jan. 2011.
[76] Qualcomm. (2010). *RaptorQ Technical Overview*. Accessed: Oct. 27, 2018. [Online]. Available: http://www.qualcomm.com/media/documents/raptorq-technical-overview.pdf
[77] J. P. Vilela, L. Lima, and J. Barros, "Lightweight security for network coding," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2008, pp. 1750–1754.
[78] P. Zhang, Y. Jiang, C. Lin, Y. Fan, and X. Shen, "P-coding: Secure network coding against eavesdropping attacks," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.
[79] J. Justesen and T. Høeholdt, *A Course in Error-Correcting Codes* (EMS Textbooks in Mathematics). Zürich, Switzerland: EMS Publishing House, 2004.
[80] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes* (North-Holland Mathematical Library). Amsterdam, The Netherlands: North-Holland Publishing Company, 1977.
[81] F. Hernando, M. E. O'Sullivan, E. Popovici, and S. Srivastava, "Subfield-subcodes of generalized toric codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2010, pp. 1125–1129.
[82] L. S. Heath and N. A. Loehr, "New algorithms for generating Conway polynomials over finite fields," *J. Symbolic Comput.*, vol. 38, no. 2, pp. 1003–1024, Aug. 2004.

DANIEL E. LUCANI (SM'16) received the B.S. degree *(summa cum laude)* and the M.S. degree (Hons.) in electronics engineering from Universidad Simon Bolvar, Caracas, Venezuela, in 2005 and 2006, respectively, and the Ph.D. degree in electrical engineering from the Massachusetts Institute of Technology in 2010. He was an Associate Professor at Aalborg University from 2012 to 2017, and an Assistant Professor at the University of Porto from 2010 to 2012. He is currently an Associate Professor with the Department of Engineering, Aarhus University, as well as the CEO and the Lead Scientist of the start-up company Chocolate Cloud ApS since 2014. He has published over 140 scientific papers in international journals and top-ranked international conferences, as well as eight patents and patent applications. His research interests include communications and networks, network coding, information theory, coding theory, distributed storage and computation, and their applications to cloud computing technologies necessary to enable IoT, BigData, and 5G applications and services. He is a recipient of the IEEE ComSoc Outstanding Young Researcher Award for the EMEA region in 2015, and a recipient of the Danish Free Research Foundation's Sapere Aude Starting Grant. He was the General Co-Chair of the 2014 International Symposium on Network Coding (NetCod2014). He is an Associate Editor of the *EURASIP Journal on Wireless Communications and Networking*.

MORTEN VIDEBÆK PEDERSEN was born in Sall, Denmark, in 1982. He received the M.Sc. degree *(cum laude)* and the Ph.D. degree from the Elite Program in wireless communication from Aalborg University, Denmark, in 2009 and 2012, respectively. Since 2006, he has been a member of the Mobile Devices Research Group, Aalborg University. He Co-Founded the start-up company Steinwurf ApS in 2011, creating industrial grade high performance content distribution systems. Since 2013, he has been a member of the Network Coding Focus Group, Aalborg University. He is currently a CTO at Steinwurf APS, where his main focus is on low complexity network coding algorithms and cooperative networking protocols. His main research interests are low-complexity network coding algorithms, cooperative networking protocols, and low-level programming. He has created and contributed to several scientific software libraries, latest by creating Kodo, a fexible high-performance network coding library, which is currently used in industry and by researchers at several international universities. In 2010, he received the Forum Nokia Developer Champion Award.

DIEGO RUANO was born in Valladolid, Spain, in 1980. He received the M.S. degrees in mathematics from the University of Valladolid, Spain, and the University of Kaiserslautern, Germany, in 2002 and 2003, respectively, and the Ph.D. degree in mathematics from the University of Valladolid in 2007. He was a Post-Doctoral Researcher at the University of Kaiserslautern in 2007, and a H.C. Ørsted Post-Doctoral Fellow at the Technical University of Denmark in 2008. He was an Assistant Professor at Aalborg University, Denmark, from 2009 to 2012, where he was an Associate Professor from 2013 to 2018. He is currently a Ramón-y-Cajal Fellow at IMUVA (Mathematics Research Institute), University of Valladolid, Spain. His research interests include classical and quantum coding theory, secret sharing, network codes, and computer algebra.

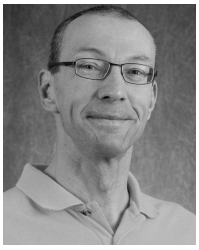
CHRES W. SØRENSEN received the B.Sc. and M.Sc. degrees in computer engineering from Aalborg University, Aalborg, Denmark, in 2011 and 2013, respectively, with the specialization in networks and distributed systems, and the Ph.D. degree from Aalborg University in 2017. Since 2017, he has been working with network coding for distributed cloud storage for a Danish Startup company, Chocolate Cloud ApS. His Ph.D. research focused on the usage of sparse network codes in practical applications and the relationship between coding complexity and coding delay in sparse network codes. His research interests include network coding, distributed systems, distributed storage, and embedded systems.

FRANK H. P. FITZEK received the Diploma (Dipl.Ing.) degree in electrical engineering from the University of Technology, Rheinisch-Westfälische Technische Hochschule Aachen, Aachen, Germany, in 1997, and the Ph.D. (Dr. Ing.) degree in electrical engineering from the Technical University Berlin, Germany, in 2002. He was an Adjunct Professor at the University of Ferrara, Italy, in 2002. In 2003, he joined Aalborg University as Associate Professor, and later became a Professor. He is currently a Professor and the Head of the Deutsche Telekom Chair of Communication Networks, Technical University Dresden, Germany, coordinating the 5G Lab Germany. He co-founded several start-up companies starting with acticom GmbH, Berlin, in 1999. 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 selected to receive the NOKIA Champion Award several times in a row from 2007 to 2011. In 2008, he was a recipient of the Nokia Achievement Award for his work on cooperative networks. In 2011, he received the SAPERE AUDE Research Grant from the Danish Government, and the Vodafone Innovation Prize in 2012. In 2015, he was a recipient of the honorary degree Doctor Honoris Causa from the Budapest University of Technology and Economy.

JANUS HEIDE received the Diploma degree *(cum laude)* and the Ph.D. degree from the Department of Electronic Systems, Aalborg University, Denmark, in 2009 and 2012, respectively. In 2009 and 2010, he was a Visiting Student at the Network Coding and Reliable Communications Group, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA. From 2012 to 2013, he was a Post-Doctoral Researcher at the Section of Antennas, Propagation and Radio Networking, Aalborg University. Since 2011, he has been the CEO and the Co-Founder of Steinwurf ApS, a communication software company that develops Network Coding software and solutions. His research interests are in the areas of network coding and wireless networks, particularly in code design, implementation and application for computational constraint devices and networks under challenging conditions.

OLAV GEIL received the B.Sc. and M.Sc. degrees in mathematics and physics from Aalborg University in 1994 and 1996, respectively, and the Ph.D. degree in mathematics from Aalborg University in 2000. He is currently a Professor with special responsibilities, within the Research Group of Reliable and Secure Communication, Department of Mathematical Sciences, Aalborg University, and serves as the Vice Dean for education of the Faculty of Engineering and Science. His research interests include coding theory, secret sharing, network coding, and commutative algebra.

VU NGUYEN received the B.Tech. (IT) degree from the Hue University of Education, Vietnam, in 2006, and the M.Eng. degree in computer science from the Da Nang University of Technology, Vietnam, in 2011. He was a Lecturer at the Vietnam-Korea Friendship IT College, Danang, Vietnam. He is currently pursuing the Ph.D. degree with the Deutsch Telekom Chair of Communication Networks, Technical University Dresden, Germany. He is interested in the research areas of network coding.



MARTIN REISSLEIN (S'96–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 for the IEEE TRANSACTIONS ON MOBILE COMPUTING, the IEEE TRANSACTIONS ON EDUCATION, the IEEE ACCESS, as well as *Computer Networks*. He is an Associate Editor-in-Chief of the IEEE COMMUNICATIONS SURVEYS & TUTORIALS and a Co-Editor-in-Chief of *Optical Switching and Networking*.