IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 30, 2020, accepted August 31, 2020, date of publication September 11, 2020, date of current version September 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3023639

Versatile Network Codes: Energy Consumption in Heterogeneous IoT Devices

VU NGUYEN[1,3], **(Graduate Student Member, IEEE), JUAN A. CABRERA**[1], **GIANG T. NGUYEN**[1], **(Member, IEEE), DONGHO YOU**[1], **(Member, IEEE), AND FRANK H. P. FITZEK**[1,2], **(Senior Member, IEEE)**
[1]Deutsche Telekom Chair of Communication Networks, Technical University of Dresden, 01062 Dresden, Germany
[2]Centre for Tactile Internet with Human-in-the-Loop (CeTI), Technical University of Dresden, 01062 Dresden, Germany
[3]Department of Computer Engineering, Vietnam - Korea University of Information and Communication Technology, Da Nang 550000, Vietnam

Corresponding author: Dongho You (dongho.you@tu-dresden.de)

This work was supported in part by the German Research Foundation, Deutsche Forschungsgemeinschaft (DFG) as part of Germany's Excellence Strategy–EXC 2050/1–Project ID 390696704–Cluster of Excellence, Centre for Tactile Internet with Human-in-the-Loop (CeTI), Technical University of Dresden, and in part by the Ministry of Education and Training, Vietnam, Project 911, under Grant 911/QD-TTg.

ABSTRACT In an Internet of Things (IoT), the number of interconnected devices is huge and has been increasing drastically. Their generated data requires powerful aggregated computing resources and consumes enormous energy for processing and transmission. Having said that, most IoT devices are very limited and heterogeneous in computing capabilities, causing a big challenge for designing a commonly used interconnect that is both reliable and energy-efficient. Random Linear Network Coding (RLNC) schemes have proven its capability both theoretically and in practical deployment not only to increase throughput and reliability but also to reduce latency and energy consumption. However, it is unclear how different variations of RLNC, in particular, Fulcrum codes aimed for heterogeneous devices perform in heterogeneous IoT settings. In this paper, we conduct a measurement campaign, allowing for a fair comparison among the state of the art RLNC families, with regard to energy consumption, decoding probability, and goodput. The study provides insights and guidelines for applying RLNC schemes to data transmission in heterogeneous IoT networks.

INDEX TERMS Network code, fulcrum code, heterogeneity, versatility, energy efficiency, the IoT devices.

I. INTRODUCTION

With the fast-dissemination of wireless smart devices, the interconnection between a large number of devices becomes more important. Since these devices are getting smaller, cheaper, and diverse in purposes, we can expect an Internet of Things (IoT) environment built with a massive amount of heterogeneous devices. This allows communicating quickly and efficiently between interoperable IoT devices. Here, the interoperability refers to the basic ability of heterogeneous IoT devices to interconnect and interact with each other [1]. However, many of the IoT devices will be difficult to recharge or even change their batteries, especially those deployed in hard-to-access places such as forests, mountains, and underwater. Therefore, reducing energy consumption is one of the hot research topics. Furthermore, since the wireless sensor network (WSN), as a subset of IoT, concentrates on computational resource-constrained IoT devices, it cannot achieve massive interoperability and interconnection with traditional communication strategies. Therefore, additional techniques are also required for such operations in WSNs, which can lead to increase cost and reduce functionality [2].

Many studies have attempted to solve the interconnection issues of WSN, and network coding (NC) has proven to have advantages enhancing the robustness and reducing delays of WSN communications [3]. Since then, NC has been in the spotlight and researched continuously in both wired and wireless communication networks. In conventional NC, a challenge is how to determine the local coding vector so that exists an invertible decoding matrix at each receiving node [4]. Ho *et al.* [5] proposed random linear network

The associate editor coordinating the review of this manuscript and approving it for publication was Irfan Ahmed.

coding (RLNC), which has been considered as the most practical NC scheme. RLNC enables all nodes to perform a random linear combination of input data to generate output data over a finite field, such as the Galois field (GF). Since this combination is independent at each node, the coding coefficients used to generate each coded packet that is transmitted along with the payloads. This allows the decoders to obtain the global coding vector and use it to recover the original data at the destination node (details are described in Section II).

The success of RLNC has derived many variations of the coding strategies to optimize for such low latency, higher throughput, security, and computational complexity. These include sliding-window RLNC [6]–[8], sparse RLNC (S-RLNC) [9], [10], tunable sparse network coding (TSNC) [11]–[14], perpetual network coding [15], fulcrum network coding (FNC) [16], dynamic sparsity and expansion packets FNC (DSEP-FNC) [17], and so on. A drawback of network coding when compared to traditional end-to-end coding schemes is its complexity. In general, the larger the field sizes used, the better the performance of the code, but also the complexity involved. For that reason, FNC was specially designed for heterogeneous devices, in which receiving nodes can trade-off the number of received packets with decoding complexity. The decoders may choose the size of the GF to decode. Depending on the computing capabilities of the device, the decoder can decode with fewer packets and a larger GF, or listen for more packets and decode with a smaller GF, which is less complex. Previous studies [16], [18]–[20] have shown that FNC outperforms conventional RLNC in terms of not only a reduction in computational complexity but also an increase in decoding probability and throughput. Besides, in the case of DSEP-FNC [17], a variation of FNC, it significantly reduces the computational complexity while maintaining high decoding probability and throughput. However, all of the performance improvements of FNC and DSEP-FNC were verified by simulation only, not by practical implementations.

Therefore, this paper focuses on investigating whether those advantages of FNC and DSEP-FNC can actually be translated into reduced computational complexity and energy consumption in high-end, off-the-shelf IoT devices, namely Odroid-XU4 and Odroid-C2 [21]. Although some previous studies measured the consumed energy of RLNC and compared it with its variations such as S-RLNC and TSNC [22], [23], this paper is, to our knowledge, the first study to explore the energy consumption and performance aspects of the latest FNC and DSEP-FNC through IoT devices. We first choose four notable representatives of network codes that are energy-efficient. Then, we measure and compare the energy consumed for the encoding and decoding processes in the Odroid devices. In addition, we evaluate the decoding probability and goodput (i.e., processing speed) of the chosen NC schemes. In particular, since it is well-known there is a trade-off between decoding probability and goodput, we focus on providing a perspective on which type of NC scheme can best compensate for this along with low energy consumption in the IoT devices.

The remainder of this paper is organized as follows: Section II provides fundamentals of RLNC, FNC, and their variations, S-RLNC and DSEP-FNC. Section III and IV describe the evaluation setup and metrics, and evaluate the performances, such as energy consumption, goodput, and decoding probability, in the two types of Odroid devices with different computing power, respectively. Finally, Section V provides its conclusion and new directions for future work.

II. BACKGROUND AND FUNDAMENTALS

It is well known that end-to-end erasure correcting codes, such as Raptor codes [24]–[26] and LT codes [27], reduce the computational complexity of the decoding process although it imposes non-negligible delay. These codes are even better than RLNC in terms of computational complexity, but they do not provide the advantages that can be gained from intermediate nodes such as recoding, unlike RLNC. Therefore, the sparsity property was applied to RLNC and led to such S-RLNC and TSNC, which was proven to be effective in the reduction of the computational complexity while maintaining the unique advantages of RLNC. In particular, [22] and [23] showed that S-RLNC consumes less energy than TSNC in IoT devices, while maintaining similar throughput performance. In other words, S-RLNC has been considered as one of the most energy-efficient network code. However, these RLNC-based network codes including S-RLNC are not actually suitable for heterogeneous IoT devices because of the adaptively unchangeable GF size in accordance with the network condition and the capability of IoT devices.

For that reason, FNC [16] and its adaptive decoders [20], [28] are proposed in which the GF size can be adaptively changed at receivers. That is, the receivers can choose GF size that they can afford. FNC has advantages in terms of resilience and heterogeneity but does not show much reduction of computational complexity compared to RLNC. In fact, reducing the computational complexity is very important to reduce the energy consumed in IoT devices. Therefore, DSEP-FNC [17] applied the dynamic sparsity property to FNC and showed outstanding performance in terms of the decoding probability and computational complexity. However, both FNC and DESP-FNC have never been practically evaluated in practical high-end IoT devices as well as never been measured the energy consumed in the devices to compare with other codes.

Therefore, this paper focuses on comparing the heterogeneity-proper FNC and DSEP-FNC with the energy-efficient S-RLNC. Toward this end, this section first describes the fundamentals of four NC schemes: i) *RLNC*, ii) *S-RLNC*, iii) *FNC*, and iv) *DSEP-FNC*.

A. ORIGINAL RLNC

For encoding of RLNC (see 2nd quadrant of Fig. 1), original input data is first divided into n packets P (or symbols), known as a generation. These packets are multiplied with

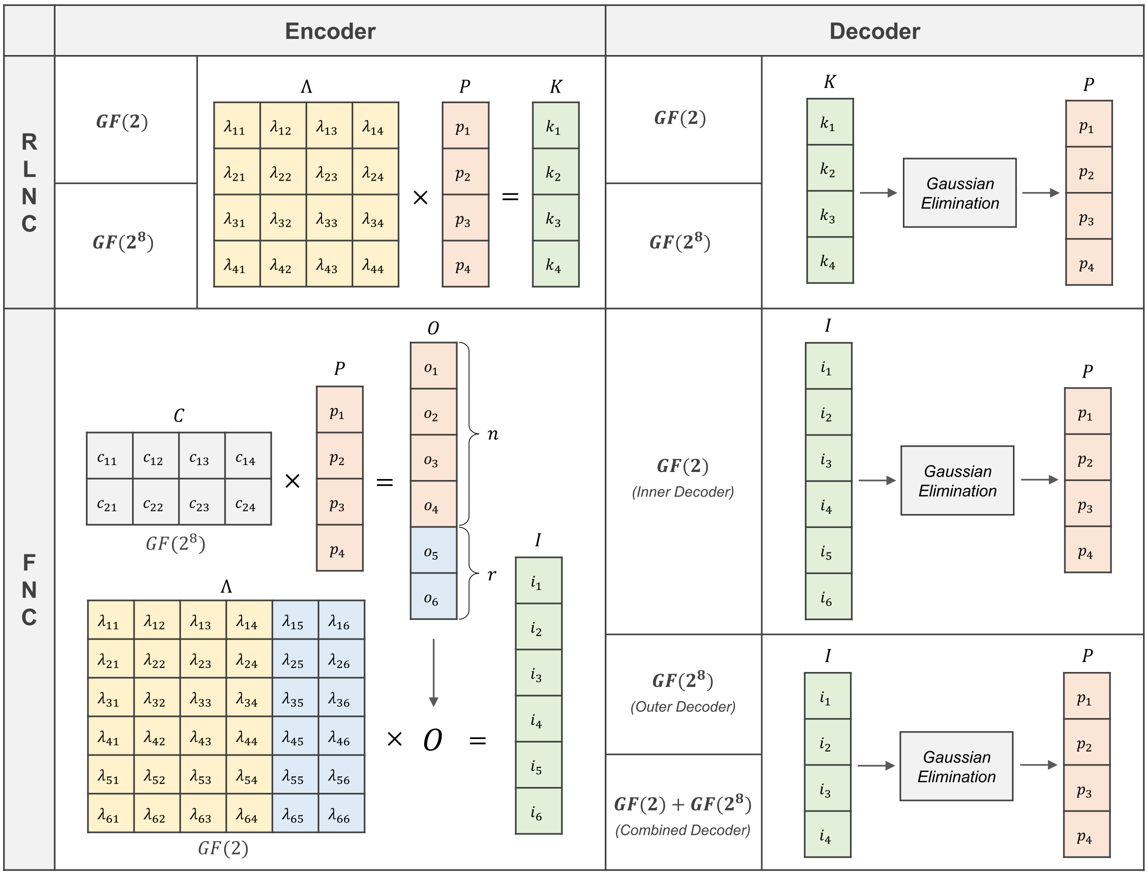


FIGURE 1. Illustration of the encoding and decoding for original RLNC and FNC schemes. Both sender and receiver attempt to encode and decode $n = 4$ original packets with different finite field sizes, and FNC generates $r = 2$ expansion packets ($n \gg r$ in the real implementation) with high finite field.

a coding coefficient vectors Λ randomly generated from a GF. Through this series of the linear combination, the size of coded packets K remains equal to that of original packets. In other words, the overall encoding process generating multiple coded packets can be considered as a system of linear equations with randomly generated coefficients over GF.

In order for successful decoding of the original packets, the coding vectors used for encoding must be transmitted to the receiver. In practice, they are appended as headers to the encoded packets and sent to the receiver. To recover the original packets P from the coded packets K, the receiver needs enough linearly independent coded packets to solve the linear system of equations. Gaussian elimination is commonly used to solve a linear system of equations $AX = B$, where A, X, and B correspond to Λ, P, and K, respectively. This has an advantage over the conventional uncoded approach because the receiver no longer needs to collect each individual data packet. In theory, it is enough to receive coded packets equal to the number of original data packets. However, due to the randomness of the generation of coding coefficients, some packets might be linearly dependent. Therefore, the system might need some extra packets (overhead) to overcome these linear dependencies [29]. If the finite field used is large enough e.g. $GF(2^8)$, this overhead can be made arbitrarily small. However, the increase of finite field causes the computational cost which is not suitable for heterogeneous IoT devices.

B. S-RLNC

This scheme was proposed to reduce the computational complexity for both encoding and decoding [9]. The idea is to utilize sparse coding coefficient rows, with only a relatively small prescribed number of non-zero coding coefficients w in the coding coefficient matrix Λ. The studies [9], [11] have shown that S-RLNC is a good for computation-limited applications. However, since it increases the overhead due to redundant transmissions, the probability of receiving linearly dependent packets increases with the sparsity of the coefficients vectors. That is, for a fixed number of received packets the decoding probability of S-RLNC is much lower than that of original RLNC. Thus, the sparsity lever w needs to be dynamically adjusted to reduce the number of linear dependencies and to increase the decoding probability at a decoder.

C. ORIGINAL FNC

In RLNC and its variations including S-RLNC, only one GF size must be fixed for encoding, e.g. $GF(2^8)$, and it cannot be changed in the middle of transmission even if the receiver has not enough capability to recover it. Solving this problem would be more critical to heterogeneous IoT devices having different capabilities. For that reason, FNC exploits a combination of two GF sizes, e.g. $GF(2)$ and $GF(2^8)$, for the encoding and decoding. As illustrated in 3rd quadrant of Fig. 1, the original packets P are first multiplied with the outer coding coefficient vectors C that randomly selected by a higher-field size such as $GF(2^8)$ to create r expansion packets o_ℓ, which is as follows:

$$o_\ell = \sum_{j=1}^{n} c_{\ell,j} p_j, \quad \ell = 1, 2, \ldots, r. \tag{1}$$

These packets are concatenated into the original ones P to create outer-coded packets ($O = \{p_1, p_2, \ldots, p_n\} \cup \{o_1, o_2, \ldots, o_r\}$). This stage is also called *pre-coding* and generally performed by a *systematic code*. More specifically, the r expansion packets ensure a high probability of receiving linearly independent coded packets and increase the robustness against packet losses over error-prone networks. A small r is suitable when linearly dependent coded packets and packet losses are unlikely to occur. However, when there are frequent linearly dependent coded packets or packet losses, a larger r will substantially increase the probability of linearly independent coded packets and the resilience against packets losses. Subsequently, each outer-coded packet o_j ($j = 1, 2, \ldots, n, n+1, n+2, \ldots, n+r$), is also multiplied with the inner coding coefficient vector $\lambda_{\ell,j}$ to generate inner-coded packets (I) as

$$i_\ell = \sum_{j=1}^{n+r} \lambda_{\ell,j} o_j, \quad \ell = 1, 2, \ldots, \tag{2}$$

where $\lambda_{\ell,j}$ are randomly selected by the encoder from a lower-field size such as $GF(2)$. The inner case is processed by either a *systematic code* or a *non-systematic code*.

Similar to RLNC, FNC also needs to receive coded packets equal to the number of original data packets to recover the original data packets. For instance, if the number of original packets P is four and encoded like 3rd quadrant of Fig. 1, the number of transmitted packets I are six. Although of course, this is higher than the number of original ones, the overhead is assumed to be negligible for larger generation sizes [8]. Assuming that the receiver with high capability only collects four among six packets due to packet erasures, it is possible to recover the original packets since the *outer* and *combined* decoders employed at the receiver. On the other hand, if the receiver has a low computational capability, it can wait to receive six linearly independent packets to decode the original information using the *inner* decoder. In essence, Fulcrum codes are the versatile network codes allowing to trade between the total number of packets needed to decode, and the computational complexity that the device must invest.

D. DSEP-FNC

DSEP-FNC [17], which inherits all the benefits of original FNC, is the combination of dynamic sparse coding and dynamic tuning the expansion packets in FNC. The main purpose is to reduce the computational complexity while maintaining a high decoding probability at both sending and receiving devices. This allows to deal with the challenges of static sparsity w in S-RLNC, static sparse FNC [18], and fixed expansion packets r by dynamically adapting these parameters in FNC encoding process.

For the dynamic sparse inner coding coefficients, we define $w(j)$ [17, Eq. (13)] as the number of packets from the set O of outer-coded packets that are to be combined for forming the next inner-coded packet i_j, which is given as

$$w(j) = (n + \mu) \min \left\{ \frac{1}{2},\ 1 - \sqrt[n+\mu-j]{1 - \frac{n+r}{n+r+\delta}} \right\}, \tag{3}$$

where $j = 0, 1, \ldots, n + r - 1$, δ is a nominal prescribed number of extra coded packets, and μ is the number of expansion packets that are considered for the inner encoding ($0 \leq \mu \leq r$). The function automatically adjusts the sparsity level toward each new inner-coded packet i_l generated by Eq. (2). These sparsity values are set to 1 and not exceed $(n+r)/2$. The low sparsity level of inner-coded packets transmitted benefits the computational complexity at the beginning of a generation, while the increase of the sparsity levels at the end of a generation helps to reduce the number of linearly dependent coded packets [11], [17]. This leads to an increase in the decoding probability compared to the S-RLNC and a reduction of the energy consumption compared to the original FNC.

In addition, the dynamic expansion packets r minimize the computational complexity toward the outer and combined decoders by adjusting the number of expansion packets μ from 0 to r within a given generation. First, the encoder generates the inner-coded packets without expansion packets at the beginning of a generation and then to increase the number of expansion packets included in the inner-coded packets towards the end of a generation. With the combination of variable μ, the inner-coded packets generated by Eq. (2) can be rewritten as

$$i_\ell = \sum_{j=1}^{n+\mu} \lambda_{\ell,j} o_j, \tag{4}$$

where $\lambda_{\ell,j}$ are randomly generated depending on the sparsity level $w(j)$ as defined in Eq. (3). When nearing the end of a generation, the decoder has collected a large number of linearly independent coded packets, which decreases the probability of newly received coded packets being linearly independent. With this adjustment, DSEP-FNC encoder can create the inner coded packets with strong robustness and a high probability of receiving linearly independent coded packets against packet losses over error-prone networks.

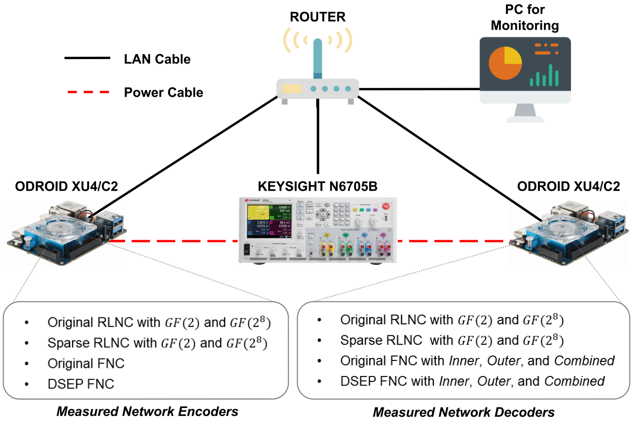

ROUTER
PC for Monitoring
LAN Cable
Power Cable
ODROID XU4/C2
KEYSIGHT N6705B
ODROID XU4/C2
Original RLNC with $GF(2)$ and $GF(2^8)$
Sparse RLNC with $GF(2)$ and $GF(2^8)$
Original FNC
DSEP FNC
Original RLNC with $GF(2)$ and $GF(2^8)$
Sparse RLNC with $GF(2)$ and $GF(2^8)$
Original FNC with *Inner*, *Outer*, and *Combined*
DSEP FNC with *Inner*, *Outer*, and *Combined*
Measured Network Encoders
Measured Network Decoders

FIGURE 2. Energy measurement setup: The power of IoT devices is supplied by an energy measurement device and these devices are connected with one another over the Ethernet to exchange the controlling commands (i.e. SCPI) and measured data.

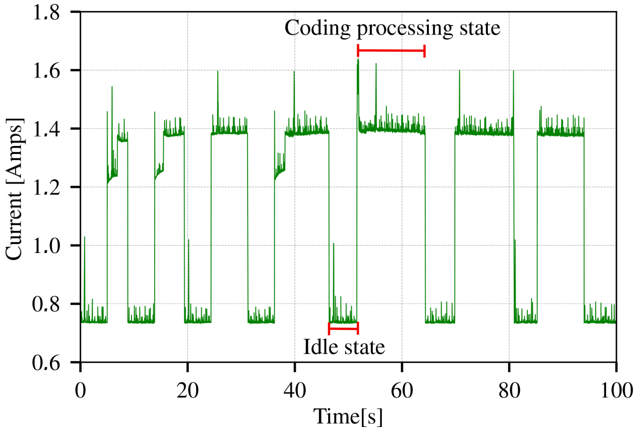


FIGURE 3. The automatically captured samples in Odroid-XU4 with two different states, which are transitions between idle state and coding processing state.

III. MEASUREMENT CAMPAIGN

The section describes the evaluation methodology of energy consumption and goodput, and furthermore, the evaluation setup and the parameters.

A. IOT DEVICES AND EVALUATION SETUP

To measure the energy consumption, we use a DC Power Analyzer Agilent N6705B from Keysight, which provides the accuracy of up to 0.025% for both the Voltmeter and Ammeter. Odroid-XU4 and Odroid-C2, representing high-end IoT devices [21], are used for this measurement. Their powers are supplied by the power analyzer and connected with one another over an Ethernet to exchange the data, as shown in Fig. 2. Odroid-XU4 has heterogeneous cores with 32 bits processors providing high performance and energy saving, while Odroid-C2 has homogeneous cores only with 64 bits processors. More specifically, Odroid C2 and XU4 feature the Amlogic system on a chip (SoC) and Samsung Exynos SoC, respectively. Both include an ARM central processing unit (CPU) and an on-chip graphics processing unit (GPU) which are widely used in IoT nodes. The C2 model has an Amlogic S905 Cortex-A53 quad-core CPU clocked up to 1.5 GHz, whereas the XU4 one has a Samsung Exynos5422 Cortex-A15 and Cortex-A7 quad-core CPUs clocked up to 2.0 GHz.

We implemented the four NC schemes on top of the Kodo library [30] to measure energy consumption and goodput. To automate the measurement process, i.e. the running of multiple tests with multiple parameters, the storing of log files with the measurement results and the synchronization between the measurements, and the loggings, we used the standard commands for programmable instruments (SCPI) which is a programming language for controlling tests and measurement instruments. This helped us to trigger the measurements and data logging of the energy device when the measured devices start carrying out the encoding or decoding process. These triggers allowed us to determine the starting and ending points of the coding process, and easily differentiate the idle state from the coding processing state (for both encoding and decoding) as shown in Fig. 3.

B. ENERGY CONSUMPTION AND GOODPUT METRICS

The energy consumption for each configuration parameter is calculated by multiplying the measured supply voltage by the measured current provided by the energy measurement device. We compute the encoding and decoding current by subtracting the average current while in the idle state as shown in Fig. 3. The energy per bit E is defined as the product of the average power P (Watt) by the coding processing time T (s) and divided by the total number of coded bits which is specifically presented in [23] and rewritten as follows.

$$E = P \times \frac{T}{8 \times n \times B} \; [Joule/bit], \tag{5}$$

where B is the data packet size (byte). Due to using r expansion packets, n is replaced with $n+r$ in Eq. (5) for both FNC and DSEP-FNC schemes.

The encoding goodput is defined as the amount of payload data in a generation, divided by the encoding processing time. The amount of data is determined by $n \times B$ for both RLNC and S-RLNC encoders, and $(n + r) \times B$ for both FNC and DSEP-FNC encoders including outer encoding $o_1, o_2, \ldots, o_r$ and inner encoding $i_1, i_2, \ldots, i_{n+r}$. Similarly, the decoding goodput is also defined as the amount of payload data in a generation $(n + r) \times B$ for the FNC inner decoder and $n \times B$ for the FNC outer, FNC combined, and RLNC decoders, divided by the total decoding processing time for successfully recovering n original packets.

C. PARAMETER SETTINGS

The data packet size was fixed to 1500 bytes which mimic the maximum size of Ethernet packets. The Galois field $GF(2^8)$ was used for FNC and FNC-DSEP outer coding, whereas the FNC and FNC-DSEP inner coding was carried out over the Galois field $GF(2)$. The number of original packets in a generation was varied from $n = 16$ to 256. The number of expansion packets was varied from $r = 2$ to 6 and the density is fixed to $d = 0.05$ or $d = 0.1$. We conducted over ten thousand independent replications for each evaluation scenario resulting in 95% confidence intervals that are too tight to be visible in the plots.

IV. PERFORMANCE EVALUATION

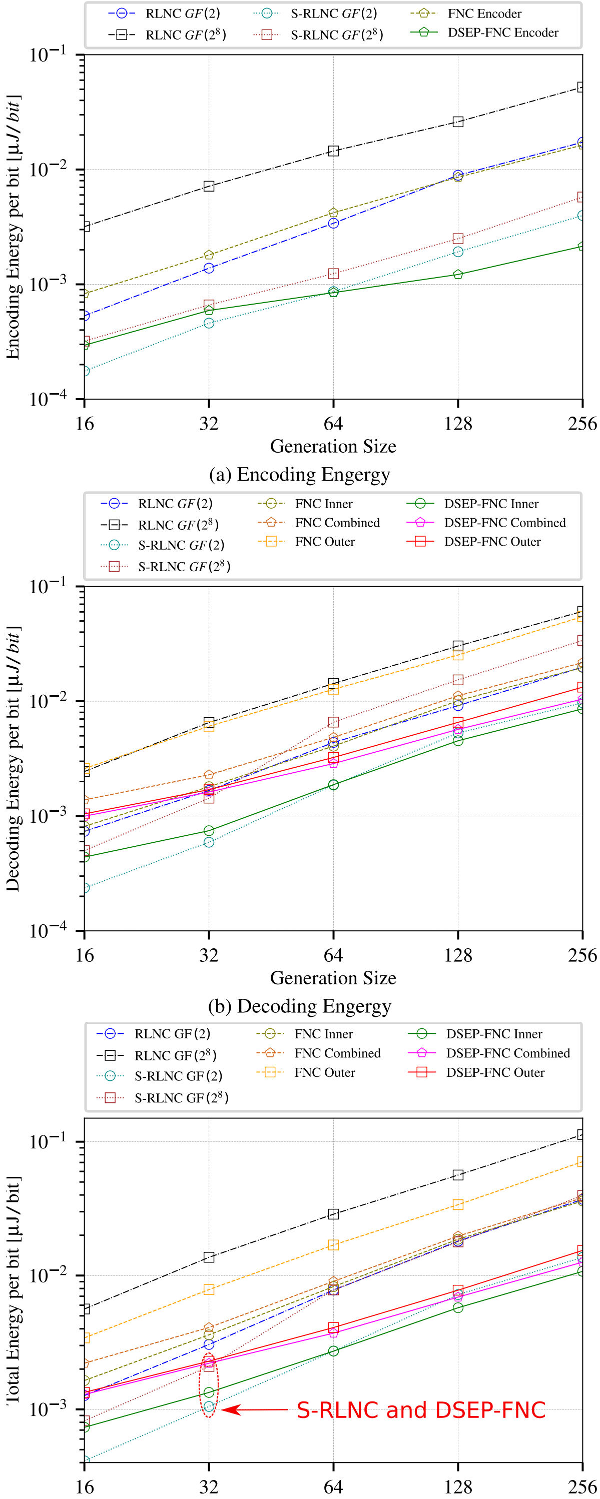
This section examines the energy consumption, decoding probability, goodput for RLNC, S-RLNC, FNC, and DSEP-FNC with the evaluation methodology in Section III.

A. ENERGY CONSUMPTION

Fig. 4 shows the energy consumed for the encoding, decoding, and total processes in the Odroid-XU4. We first observe that S-RLNC (dot line) and DSEP-FNC (dash line) consume less energy compared to the original RLNC and FNC in both encoding and decoding processes due to their sparsity characteristics. Since it is already well-known that the sparsity characteristics allow a reduction of computational complexity, this fact is predictable enough. However, as the generation sizes increase, S-RLNC with high GF size (dot line with square markers) consumes more energy and has a similar tendency to the original FNC. That is, all types of DSEP-FNC and S-RLNC with low GF size only show the improved performance compared to others in terms of energy consumption.

Fig. 4(a) shows the consumed energy of encoding as an increase in the number of generation sizes. In particular, we observe that the encoder of DSEP-FNC consumes very small energy compared to FNC encoder and RLNC $GF(2)$ encoder, and it consumes the lowest energy even for large generation sizes ($n \geq 64$). This is mainly due to the particular adjustment of sparsity level (the number of non-zero coefficients in each coding vector is controlled by $w(j)$ and changed from very low to high during the encoding process) and expansion packets (by adjusting the number of expansion packets μ from 0 to r within a given generation) in DSEP encoder. S-RLNC $GF(2)$ encoder also takes advantage of low sparsity by fixing the static sparsity during the encoding process, but its energy consumption is higher than DSEP-FNC encoder in large generation sizes ($n > 64$). This is because the energy consumption of S-RLNC is the fixed level of sparsity, e.g. $d = 0.05$, that gives nearly constant the number of non-zero coefficients in each coding vector within a given generation size. Therefore, for $n = 256$, the number of non-zero coefficients in each coding vector is 12.8 on average, whereas, the sparsity level $w(j)$ in DSEP-FNC is dynamically adjusted during the encoding process.

Similarly, the decoding process of S-RLNC and DSEP-FNC surpasses the counterpart schemes in terms of energy consumption, as shown in Fig. 4(b). This is because of low sparsity level in both S-RLNC and DSEP-FNC, and there is, thus, a significant reduction of the computational complexity of decoding process which mainly impacts on the decrease of energy consumption. Consequently, the total energy consumption of DSEP and S-RLNC is also small, as shown in Fig. 4(c). More specifically, we observe that DSEP-FNC inner reduces the total energy consumption by up to 73%,

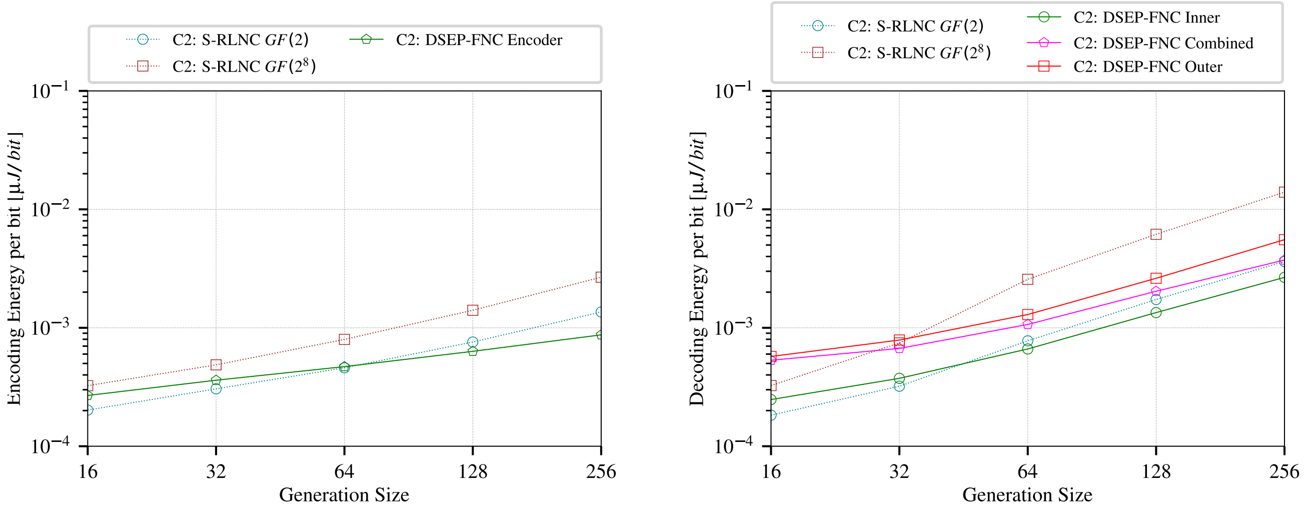


(a) Encoding Engergy

(b) Decoding Engergy

(c) Total Engergy

FIGURE 4. The consumed energy of (a) encoding, (b) decoding, and (c) total in Odroid-XU4. Different generation sizes *n*, static density of S-RLNC $d = 0.05$ and expansion packets $r = 2$ for both FNC and DSEP-FNC.



(a) Encoding

(b) Decoding

FIGURE 5. **The consumed energy of encoding and decoding in Odroid-C2. Different generation sizes n, static density of S-RLNC $d = 0.05$ and expansion packets $r = 2$ used for both FNC and DSEP-FNC.**

25%, and 72% as compared with RLNC, S-RLNC, and FNC, respectively, for the low field using $GF(2)$. DSEP-FNC outer and combined also reduce the total energy consumption by up to 90%, 75%, and 70%, respectively, for the high field using $GF(2^8)$. This significant reduction is because of the sparse inner coding which is sped up the reduced number of non-zero coding coefficients in the DSEP-FNC approaches. This fact is clearer in the case of S-RLNC with $GF(2)$, which saves the most consumed energy in the small generation sizes between $n = 16$ and 64, with only one and three non-zero coding coefficients in average.

Fig. 5 shows the consumed energy in Odroid-C2 for both encoding and decoding processes. For simplicity, we only choose and present only S-RLNC and DSEP-FNC that are shown low energy consumption in Odroid-XU4 (see Fig. 4(c)). We observe that Odroid-C2 has similar tendency to Odroid-XU4 for both encoding and decoding. However, Odroid-C2 equipped with 64-bit quad-core single boards (cortex-A53 CPU) shows that it is low power hardware with high-performance [31], compared to Odroid-XU4 equipped with four 32-bit cores (cortex-A15 and cortex-A7 CPU). This leads to the energy consumption of both encoding and decoding for Odroid-C2 is lower than Odroid-XU4 as presented in Fig. 5.

B. DECODING PROBABILITY

From the results of energy consumption, we can observe that S-RLNC and DSEP-FNC consume similar energy in the IoT devices. Of course, S-RLNC consumes slightly lower energy for small generation sizes n. However, the fixed sparsity in S-RLNC significantly impacts on the decoding probability as we explained in Section II-B.

Fig. 6 shows the decoding probability of S-RLNC and DSEP-FNC as a function of the number of the received extra packet. In the low field size, both S-RLNC $GF(2)$ and DSEP-FNC inner have severe performance degradation. Even if S-RLNC $GF(2)$ has better performance than DSEP-FNC inner, both have very poor performance, which may impose significant delay due to the retransmission. On the other hand, DSEP-FNC outer and combined outperform S-RLNC $GF(2^8)$ in terms of decoding probability, even if S-RLNC $GF(2^8)$ is slightly better when the number of extra packet is greater than or equal to 1. In addition, only DSEP-FNC outer and combined achieve almost 100% decoding probability when receiving more than two extra packets. This means that DSEP-FNC requires a shorter delay to complete the transmission of a given generation because DSEP-FNC is designed as a dynamic sparsity and expansion packets in the coding coefficients. It also allows either to substantially reduce the computational complexity at the beginning of generation, or to increase the decoding probability and robust packets against the loss of channels by adjusting high-density

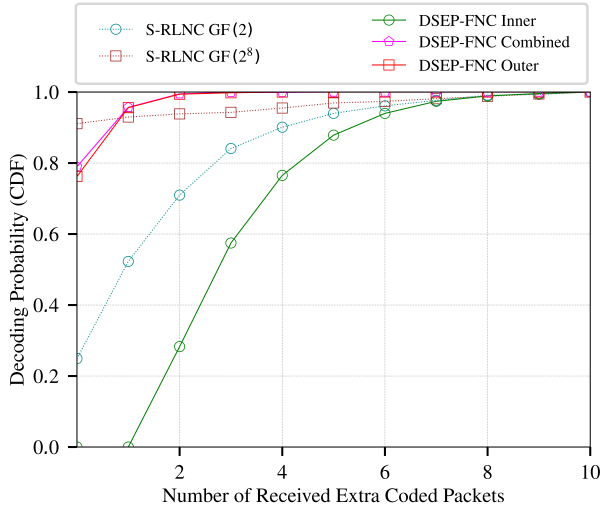
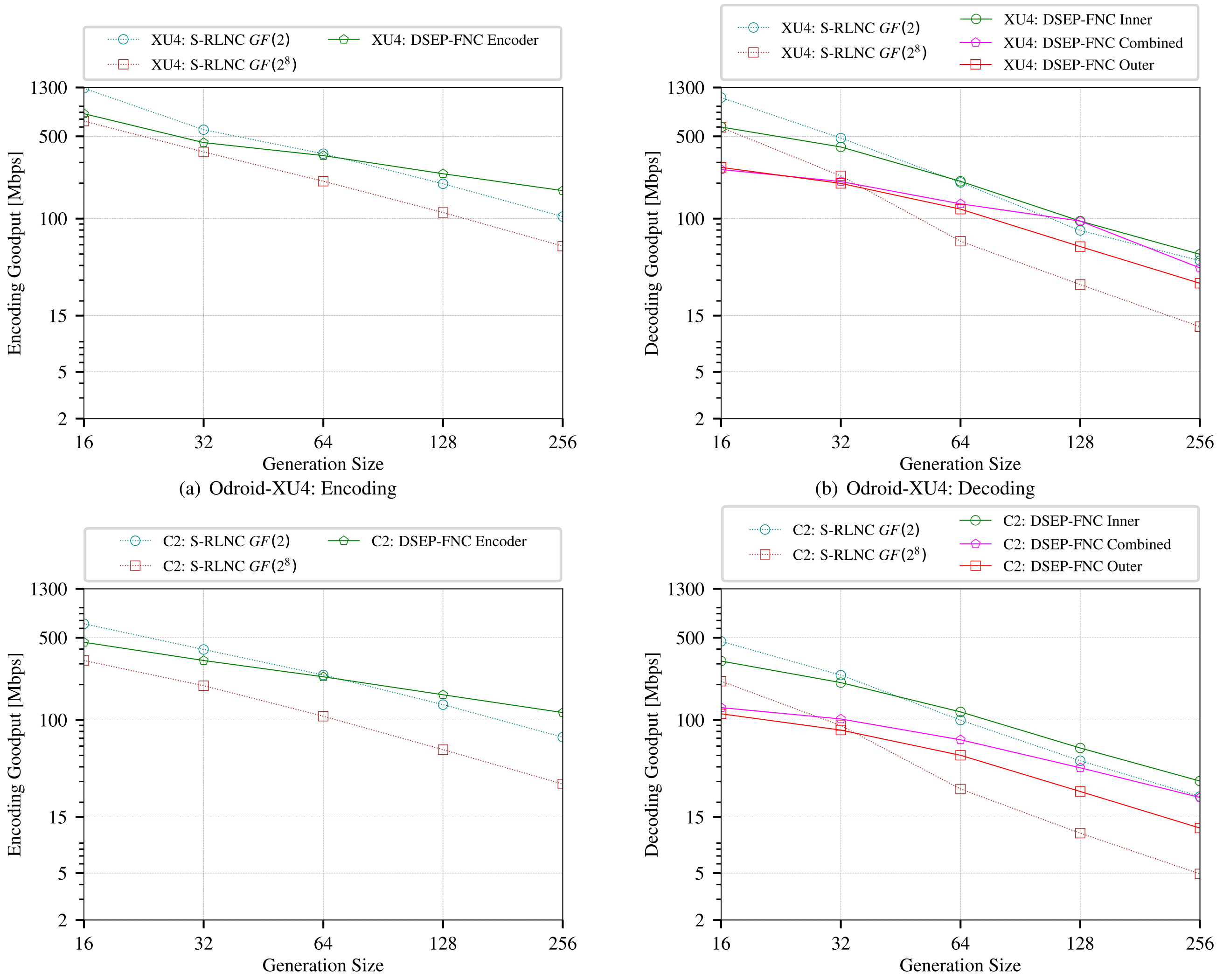


FIGURE 6. **Decoding probability of S-RLNC and DSEP-FNC. Generation size of $n = 64$, static density of S-RLNC $d = 0.1$, and expansion packets $r = 2$.**



(a) Odroid-XU4: Encoding

(b) Odroid-XU4: Decoding

(c) Odroid-C2: Encoding

(d) Odroid-C2: Decoding

FIGURE 7. Encoding and decoding goodput in Odroid-XU4 and Odroid-C2. Different generation sizes *n*, static density of S-RLNC $d = 0.05$ and expansion packets $r = 2$ used for both FNC and DSEP-FNC.

level at the end of generation. On the contrary, S-RLNC sets the static sparsity in the coding coefficients during the coding process. Therefore, although the energy consumption is low, the decoding probability is not as high as DSEP-FNC. In summary, only DSEP-FNC combined and outer achieve almost 100% decoding probability while maintaining low energy consumption.

C. GOODPUT

From the previous results, we confirm that only DSEP-FNC outer and combined not only significantly improve the decoding probability but also keep the low energy consumption in the IoT devices. In addition, they also show the improved performance in terms of goodput, especially for encoding, as shown in Fig. 7(a) and (c). More specifically, as compared to S-RLNC $GF(2^8)$, the processing speed of DSEP-FNC encoding is three times faster despite saving energy up to 42% in the high generation size.

This trend is also maintained in the decoding process. We observe that DSEP-FNC combined decoder saves about 32% of the consumed energy in the decoding process, and furthermore, its processing speed is up to four times faster, as compared with S-RLNC GF(2^8) (see Fig. 7(b) and (d)). Of course, S-RLNC GF(2) shows less energy consumption and faster processing than DSEP-FNC combined, but these performances become similar to DSEP-FNC combined when the generation size is high. In particular, since S-RLNC GF(2) aggravates the decoding probability, it would impose significant delay rather than DSEP-FNC combined.

And one more interesting thing is that Odroid-XU4 consumes more energy than Odroid-C2 but it shows a faster processing, as illustrated in Fig. 7. This is because Odroid-XU4 is equipped with multiple cores, while Odroid-C2 is equipped with a single core. Therefore, Odroid-XU4 can achieve a fast encoding or decoding process.

D. EVALUATION SUMMARY

Our evaluations can be summarized in following points:

1) Because of the sparsity characteristics, S-RLNC and DSEP-FNC consume significantly less energy in the two types of Odroid devices than their original versions, RLNC and FNC, respectively.
2) For encoding, DSEP-FNC overall consumes less energy than S-RLNC regardless of GF sizes, but more energy than S-RLNC GF(2) in small generation sizes.
3) For decoding using low field size, i.e. S-RLNC GF(2) and DSEP-FNC inner, they consume similar energy on average in the IoT devices. However, they aggravate the decoding probability because of the bad influence of the fixed sparsity and would impose significant delay due to additional transmissions.
4) For decoding using higher field size, DSEP-FNC outer and combined outperform S-RLNC GF(2^8) in terms of energy consumption, except for the small generation sizes.
5) All, S-RLNC GF(2^8), and DSEP-FNC outer and combined, have outstanding performance in terms of decoding probability. However, DSEP-FNC combined shows faster processing time and is up to four times faster than S-RLNC GF(2^8).
6) Consequently, in terms of versatility including energy efficiency, decoding probability, and goodput in IoT devices, using DSEP-FNC combined is more proper than using other network codes.

V. CONCLUSION

It is well known among the network coding community that S-RLNC is more energy-efficient than the original RLNC at the trade-off of a reduced decoding probability. Furthermore, recently proposed codes in the literature, such as Fulcrum Network Codes (FNC) also demonstrate advanced characteristics. To find out how different variations RLNC codes perform, in terms of reliable and energy-efficient in the IoT domain, we conduct a measurement campaign. Particularly, we measure energy consumption, decoding probability, and goodput of four representatives, RLNC, S-RLNC, FNC, and DSEP-FNC. We observe that DSEP-FNC outperforms S-RLNC and achieves both high energy efficiency and high decoding probability. Specifically, DSEP-FNC encoder reduces energy consumption by about 42% in most of the cases without compromising the decoding probability, as compared to S-RLNC encoder. Additionally, the combined decoder of DSEP-FNC increases the goodput by up to four times while reducing energy consumption by at least 30%. Demonstrating an outstanding performance across a wide range of parameters, such as generation size and the number of extra packets, DESP-FNC and its combined decoder can be an appropriate candidate for interconnecting heterogeneous IoT devices.

Possible directions for future work include analyzing the impact of using these versatile network codes in practical MAC protocols such as ZigBee. This direction is important because ZigBee-based WSN may consist of thousands of nodes that perform the processing and communicate with each other. In particular, NC recoding mechanism should be investigated more in this MAC protocol. It would improve the performances of intermediate IoT devices that do not support for higher layers.

REFERENCES

[1] R. Gravina, C. E. Palau, M. Manso, A. Liotta, and G. Fortino, *Integration, Interconnection, and Interoperability of IoT Systems*. Cham, Switzerland: Springer, 2018.

[2] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, "Middleware for Internet of Things: A survey," *IEEE Internet Things J.*, vol. 3, no. 1, pp. 70–95, Feb. 2016.

[3] C. Fragouli, J.-Y. Le Boudec, and J. Widmer, "Network coding: An instant primer," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, pp. 63–68, Jan. 2006.

[4] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. M. G. M. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Inf. Theory*, vol. 51, no. 6, pp. 1973–1982, Jun. 2005.

[5] T. Ho, M. Medard, 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.

[6] 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, pp. 44787–44802, 2018.

[7] D. Malak, M. Medard, and E. M. Yeh, "Tiny codes for guaranteeable delay," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 4, pp. 809–825, Apr. 2019.

[8] S. Wunderlich, F. Gabriel, S. Pandi, F. H. P. Fitzek, and M. Reisslein, "Caterpillar RLNC (CRLNC): A practical finite sliding window RLNC approach," *IEEE Access*, vol. 5, pp. 20183–20197, 2017.

[9] A. Tassi, I. Chatzigeorgiou, and D. E. Lucani, "Analysis and optimization of sparse random linear network coding for reliable multicast services," *IEEE Trans. Commun.*, vol. 64, no. 1, pp. 285–299, Jan. 2016.

[10] Y. Li, W.-Y. Chan, and S. D. Blostein, "On design and efficient decoding of sparse random linear network codes," *IEEE Access*, vol. 5, pp. 17031–17044, 2017.

[11] S. Feizi, D. E. Lucani, C. W. Sorensen, A. Makhdoumi, and M. Medard, "Tunable sparse network coding for multicast networks," in *Proc. Int. Symp. Netw. Coding (NetCod)*, Jun. 2014, pp. 1–6.

[12] P. Garrido, C. W. Sørensen, D. E. Lucani, and R. Aguero, "Performance and complexity of tunable sparse network coding with gradual growing tuning functions over wireless networks," in *Proc. IEEE 27th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Sep. 2016, pp. 1–6.

[13] P. Garrido, D. E. Lucani, and R. Aguero, "How to tune sparse network coding over wireless links," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Mar. 2017, pp. 1–6.

[14] C. W. Sorensen, D. E. Lucani, F. H. P. Fitzek, and M. Medard, "On-the-fly overlapping of sparse generations: A tunable sparse network coding perspective," in *Proc. IEEE 80th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2014, pp. 1–5.

[15] J. Heide, M. V. Pedersen, F. H. P. Fitzek, and M. Medard, "A perpetual code for network coding," in *Proc. IEEE 79th Veh. Technol. Conf. (VTC Spring)*, May 2014, pp. 1–6.

[16] D. E. Lucani, M. V. Pedersen, D. Ruano, C. W. Sorensen, F. H. P. Fitzek, J. Heide, O. Geil, V. Nguyen, and M. Reisslein, "Fulcrum: Flexible network coding for heterogeneous devices," *IEEE Access*, vol. 6, pp. 77890–77910, 2018.

[17] V. Nguyen, E. Tasdemir, G. T. Nguyen, D. E. Lucani, F. H. P. Fitzek, and M. Reisslein, "DSEP fulcrum: Dynamic sparsity and expansion packets for fulcrum network coding," *IEEE Access*, vol. 8, pp. 78293–78314, 2020.

[18] V. Nguyen, G. T. Nguyen, F. Gabriel, D. E. Lucani, and F. H. P. Fitzek, "Integrating sparsity into fulcrum codes: Investigating throughput, complexity and overhead," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, May 2018, pp. 1–6.

[19] V. Nguyen, E. Tasdemir, G. Nguyen, D. Lucani, and F. Fitzek, "Tunable expansion packets for Fulcrum codes," in *Proc. Eur. Wireless*, 2019, pp. 1–7.

[20] V. Nguyen, J. A. Cabrera, D. You, H. Salah, G. T. Nguyen, and F. H. P. Fitzek, "Advanced adaptive decoder using fulcrum network codes," *IEEE Access*, vol. 7, pp. 141648–141661, 2019.

[21] M. O. Ojo, S. Giordano, G. Procissi, and I. N. Seitanidis, "A review of low-end, middle-end, and high-end IoT devices," *IEEE Access*, vol. 6, pp. 70528–70554, 2018.

[22] C. W. Sorensen, 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.

[23] N. H. Marcano, C. Sørensen, J. Cabrera G., S. Wunderlich, D. Lucani, and F. Fitzek, "On goodput and energy measurements of network coding schemes in the raspberry pi," *Electronics*, vol. 5, no. 4, p. 66, Oct. 2016.

[24] M. Luby, A. Shokrollahi, M. Watson, and T. Stockhammer, *Raptor Forward Error Correction Scheme for Object Delivery*, Internet Requests for Comments, RFC Editor, RFC 5053, Oct. 2007. [Online]. Available: https://www.rfc-editor.org/rfc/rfc5053.txt

[25] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2551–2567, Jun. 2006.

[26] A. Shokrollahi and M. Luby, "Raptor codes," *Found. Trends Commun. Inf. Theory*, vol. 6, nos. 3–4, pp. 213–322, May 2011.

[27] M. Luby, "LT codes," in *Proc. Annu. IEEE Symp. Found. Comput. Sci.*, Nov. 2002, pp. 271–280.

[28] V. Nguyen, J. A. Cabrera, G. T. Nguyen, F. Gabriel, C. Lehmann, S. Mudriievskyi, and F. H. P. Fitzek, "Adaptive decoding for fulcrum codes," in *Proc. IEEE 9th Annu. Inf. Technol., Electron. Mobile Commun. Conf. (IEMCON)*, Nov. 2018, pp. 133–139.

[29] J. Heide, M. V. Pedersen, F. H. P. Fitzek, and M. Medard, "On code parameters and coding vector representation for practical RLNC," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2011, pp. 1–5.

[30] M. V. Pedersen, J. Heide, and F. Fitzek, "Kodo: An open and research oriented network coding library," *Lect. Notes Comput. Sci.*, vol. 6827, pp. 145–152, May 2011.

[31] M. D. Görtz, R. Kühn, O. Zietek, R. Bernhard, M. Bulinski, D. Duman, B. Freisen, U. Jentsch, T. Klöppner, D. Popovic, and L. Xu, "Energy efficiency of a low power hardware cluster for high performance computing," in *Proc. INFORMATIK*, 2017, pp. 2537–2548.

VU NGUYEN (Graduate Student Member, IEEE) received the B.Tech. (I.T.) 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 is currently pursuing the Ph.D. degree with the Deutsch Telekom Chair of Communication Networks, Technical University of Dresden, Germany. He was a Lecturer with the Vietnam - Korea University of Communication and Technology, The University of Da Nang, Da Nang, Vietnam. His research interests include network coding, high-performance, and low-complexity communications.

JUAN A. CABRERA received the M.Sc. degree in wireless communication systems from Aalborg University, Aalborg, Denmark. He is currently a Ph.D. Researcher with the Deutsche Telekom Chair of Communication Networks, Technical University of Dresden, Dresden, Germany. His research interests include network coding, distributed computing, and distributed storage.

GIANG T. NGUYEN (Member, IEEE) received the master's degree (M.Eng.) in telecommunications from the Asian Institute of Technology (AIT), Thailand, in 2007, and the Ph.D. (Dr.-Ing.) degree in computer science from the Technical University of Dresden, Germany, in 2016. He is currently a Senior Researcher with the Deutsche Telekom Chair of Communication Networks, Technical University of Dresden. His research interests include 5G and highly-adaptive and energy-efficient computing (HAEC), network coding and low-latency aspects of network function virtualization, and software-defined networking.

DONGHO YOU (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in media IT engineering from the Seoul National University of Science and Technology, South Korea, in 2012, 2014, and 2018, respectively. He was a Visiting Researcher with the IoT Convergence Research Center, Korea Electronics Technology Institute, in 2014. He is currently a Senior Researcher with the Deutsche Telekom Chair of Communication Networks, Technical University of Dresden, Germany. His current research interests include 5G communication and networks for robust and reliable multimedia communications. He received the Research Grant for the Fostering Next-Generation Researchers Program from the National Research Foundation, South Korea, in 2018. He was also selected to receive the Research Grant from the Alcatel-Lucent Foundation.

FRANK H. P. FITZEK (Senior Member, IEEE) received the Dipl.-Ing. degree in electrical engineering from the University of Technology–Rheinisch-Westfälische Technische Hochschule (RWTH), Aachen, Germany, in 1997, the Ph.D. (Dr.-Ing.) degree in electrical engineering from the Technical University of Berlin, Germany, in 2002, and the Doctor Honoris Causa degree from the Budapest University of Technology and Economy. He was an Adjunct Professor with the University of Ferrara, Italy, in 2002. In 2003, he joined Aalborg University as an Associate Professor and a Professor. He has co-founded several start-up companies with acticom GmbH, Berlin, in 1999. He is currently a Professor and the Head of the Deutsche Telekom Chair of Communication Networks, Technical University of Dresden, Germany. He is also coordinating the 5G Laboratory, Germany. His current research interests include wireless and mobile 5G communication networks, mobile phone programming, network coding, cross layer and 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. He also received the Nokia Achievement Award for the work on cooperative networks in 2008, the SAPERE AUDE Research Grant from the Danish Government in 2011, and the Vodafone Innovation prize in 2012.