Rate-Compatible LDPC Codes Based on Primitive Polynomials and Golomb Rulers

Massimo Battaglioni, *Member, IEEE*, Marco Baldi, *Senior Member, IEEE*, Franco Chiaraluce, *Senior Member, IEEE*, and Giovanni Cancellieri

***Abstract*— We introduce and study a family of rate-compatible Low-Density Parity-Check (LDPC) codes. The design of these codes starts from simplex codes, defined by parity-check matrices having a simple form stemming from the coefficients of a primitive polynomial. For this reason, we call the new codes Primitive Rate-Compatible LDPC (PRC-LDPC) codes. By applying puncturing to these codes, we obtain a bit-level granularity of the code rate. We show that, in order to achieve good LDPC codes, the underlying polynomials, besides being primitive, must meet some more stringent conditions with respect to those of classical punctured simplex codes. We leverage non-modular Golomb rulers to take these new requirements into account. We characterize the minimum distance properties of PRC-LDPC codes, and study and discuss their encoding and decoding complexity. Finally, we assess the error rate performance of high rate PRC-LDPC codes under iterative decoding.**

***Index Terms*— Golomb rulers, LDPC codes, minimum distance, rate-compatible codes, simplex codes.**

I. Introduction

LOW-DENSITY Parity-Check (LDPC) codes are a family of error correcting codes widely employed in modern communication systems, due to their ability to provide excellent error rate performance with low decoding complexity. Recently, Rate-Compatible LDPC (RC-LDPC) codes, introduced in [3], have gained increasing attention, since they offer the flexibility required by many modern applications. In fact, these codes can support different code rates by using a single code design, as for the wider class of general rate compatible codes [4]. Such a flexibility can represent a significant advantage, for example, in modern radio and wireless communications like the fifth generation (5G) or the sixth generation (6G) of mobile communications, since the underlying transmission systems need to support a wide range of data rates and channel conditions to meet the requirements of the various application scenarios [5]. Another advantage of RC-LDPC codes is their ability to achieve low encoding and decoding complexity. In fact, since all the obtainable codes can be constructed starting from a common structure (e.g., a base matrix), encoding and decoding can be simplified with respect to the case of conventional Low-Density Parity-Check (LDPC) codes. This makes the new family of codes suitable for real-time communication systems, where low latency is a critical requirement. The small encoding/decoding complexity of RC-LDPC codes makes them suitable also for energy-efficient communications systems, which are characterized by low power and computational capacity. For all these reasons, RC-LDPC codes are actually employed in 5G [6] and will likely remain a critical component of future communication systems and standards.

A. Our Contribution

The goal of this paper is threefold. First, we introduce a new family of RC-LDPC codes, called Primitive RC-LDPC (PRC-LDPC) codes. Secondly, we provide insights on how Golomb rulers can be employed to design LDPC codes. Finally, we demonstrate the possibility of using simple encoders for codes in this family and assess their error-correction performance, showing that it is similar to that of LDPC codes available in the literature and included in the 5G standard.

Our design starts from simplex codes, but we show that the primitive polynomial representing the parity-check matrix of the code needs to comply with some additional constraints in order to be an LDPC code. Namely, first of all, we show that to satisfy the Row-Column Constraint (RCC) [7], the support of the coefficients vector of the primitive polynomial must be a Golomb ruler. We also show that this is a necessary but not sufficient condition to obtain a good LDPC code. In fact, inaccurate choices of the polynomial can yield poor Hamming weight distributions and very poor minimum distance properties for the code. Therefore, we provide several rules to avoid these inaccurate choices. In order to obtain a

Manuscript received 25 December 2023; revised 8 April 2024; accepted 5 June 2024. Date of publication 17 June 2024; date of current version 19 December 2024. The work of Marco Baldi and Franco Chiaraluce was supported in part by the Italian Ministry of University's PRIN 2022 program under the "Mathematical Primitives for Post Quantum Digital Signatures" (Grant P2022J4HRR), by the "Post Quantum Identification and Encryption Primitives: Design and Realization (POINTER)" projects funded by the European Union-Next Generation EU (Grant 2022M2JLF2), and by the European Union under the Italian National Recovery and Resilience Plan of NextGenerationEU, partnership on "Telecommunications of the Future" (PE00000001 - "RESTART"). The work of Massimo Battaglioni was supported by the European Union under the Italian National Recovery and Resilience Plan of NextGenerationEU, partnership on "Telecommunications of the Future" (PE00000001 - "RESTART"). An earlier version of this paper was presented in part at the 2022 61st FITCE [DOI: 10.23919/FITCE56290.2022.9934548] and in part at the 2023 IEEE ICC [DOI: 10.1109/ICC45041.2023.10279448]. The associate editor coordinating the review of this article and approving it for publication was M. Shirvanimoghaddam. *(Corresponding author: Massimo Battaglioni.)*

Massimo Battaglioni, Marco Baldi, and Franco Chiaraluce are with the Department of Information Engineering, Università Politecnica delle Marche, 60131 Ancona, Italy, and also with CNIT, 43124 Parma, Italy (e-mail: m.battaglioni@univpm.it).

Giovanni Cancellieri, retired, was with the Department of Information Engineering, Università Politecnica delle Marche, 60131 Ancona, Italy.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2024.3415597.

Digital Object Identifier 10.1109/TCOMM.2024.3415597

very fine rate-compatibility, we make use of puncturing [8], in such a way that both high code rates and low code rates can be achieved. If needed, also shortening operations can be performed. The minimum distance profile of the new codes is then analyzed. Many theoretical results are provided, and a method to estimate their minimum distance is discussed. Numerical examples show that the proposed method yields a good predictability of minimum distance properties. We finally show that, as mentioned, the PRC-LDPC codes can be encoded by using a very simple encoder circuit, and have a good error rate performance even when they are decoded with the low-complexity versions of Belief Propagation (BP) algorithms typically used for decoding LDPC codes.

The advantages of PRC-LDPC codes over existing LDPC code designs are underscored by the simultaneous presence of: low representation cost; low encoding/decoding complexity; structured parity-check matrices facilitating theoretical analysis; inherent rate compatibility.

B. Related Works

The literature contains many works on rate compatible codes and codes whose design is based on Golomb rulers. Next we briefly describe those that are most closely related to our study.

Some methods utilize modular Golomb rulers to construct circulant matrices that form the parity-check matrix of binary Quasi-Cyclic LDPC (QC-LDPC) codes, as described in [9], [10], and [11]. Similarly, non-binary LDPC codes that employ modular Golomb rulers are developed in [12] and [13]. Compared with the other methods, PRC-LDPC codes are: (i) binary, (ii) non-QC, and (iii) based on standard (non-modular) Golomb rulers. The latter issue is particularly important as it is indeed the removal of the modular constraint that leads to the bit-level rate adaptivity of the new family of codes.

In [14], primitive rateless codes are designed. It is recognized that these codes can be realized as punctured simplex codes. However, differently from our codes, primitive rateless codes are not necessarily LDPC codes and are not treated as such. The inclusion of the requirement on the sparsity of the parity-check matrix deserves a self-standing analysis, taking into account the peculiarities of LDPC codes and their decoding algorithms. In the same paper, many results on the average Hamming weight distribution of these codes are provided. We instead do not restrict ourselves to studying the average case, but analyze the minimum distance properties for any individual code, based on its parity-check polynomial. Furthermore, we show that the satisfaction of the RCC, necessary for LDPC codes, leads to new results on the code minimum distance, which do not necessarily hold if length-4 cycles are present in the code Tanner graph [15], the latter actually being the setting of most previous works, where punctured simplex codes are not considered as LDPC codes. Practically speaking, the satisfaction of the RCC for PRC-LDPC codes translates into the fact that the support vector of the coefficients vector of the parity-check polynomial representing any code family must be a Golomb ruler. Finally, it is worth mentioning that we deal with relatively large values of the code dimension k and, if needed, we combine puncturing and shortening operations to obtain codes with the desired code rate.

In [16] punctured simplex codes are employed for error detection rather than for error correction. Also in this case, the designed codes are not LDPC codes. The provided results on the weight distribution hold for values of the block length $n > 2^{k-1}$. We instead employ values of n that, in most cases, do not satisfy the above condition, and therefore do not (and cannot) make use of the above results.

This work extends and improves on our previous preliminary works [1], [2]. In those papers, we introduced the problem and provided a preliminary theoretical analysis, but many results are partial, or valid only for specific values of the code rate. Moreover, only loose empirical estimates of the minimum distance are given, which may not hold in general. In this paper we generalize, enrich and make both the theoretical and the numerical treatment more robust, by providing a comprehensive overview of PRC-LDPC codes. Specifically, the differences between this paper and [1], [2] involve:

- **Theoretical Perspective:** Neither [1] nor [2] delve into the evolution of the minimum distance profile for different values of n, which is dealt with in Section IV-B.
- **Code Design:** Absent in [1] and [2] are guidelines regarding code design, which is instead addressed in Section IV-C.
- **Error Rate Performance:** Differently from [1] and [2], we apply shortening techniques to PRC-LDPC codes to match the desired block length and rate. We also provide many new numerical results, proving the advantages of our rate-compatible design.
- **Complexity:** Complexity aspects, treated in Section V, are not explored in [1] and [2].

C. Outline of the Paper

In Section II we introduce the notation and recall the necessary background. In Section III we describe our design and discuss its requirements. In Section IV we give some rules for the choice of the primitive polynomial and provide theoretical results and examples. In Section V we discuss encoding and decoding complexity. In Section VI we assess the error rate performance of the newly designed codes. Section VII concludes the paper.

II. Preliminaries

We use the notation $[a, b]$ to represent the set of integers between a and b, including the endpoints. The symbol $\bigoplus$ denotes modulo-2 sum. A sequence of non-negative integers where every difference between two integers is distinct is called a Golomb ruler. The Hamming weight of a vector is the number of non-zero symbols it contains and in the following we simply refer to it as its weight. Similarly, the weight of a polynomial is the number of its non-zero coefficients. To every polynomial $h(x) = h_0 + h_1 x + \ldots + h_k x^k$, we associate a coefficients vector $\mathbf{h} = (h_0, \ldots, h_k)$. The reciprocal of a polynomial $h(x)$ is denoted as $h^*(x) = x^k h(x^{-1})$. Since it is widely employed, the Hamming weight of $\mathbf{h}$ is denoted as w_h. A polynomial of degree k in $\mathbb{F}_2[x]$ is said to be primitive if it is the minimal polynomial of a primitive element of $\mathbb{F}_{2^k}$. It is well-known that primitive polynomials have an odd weight.

Given the finite field $\mathbb{F}_q$ with order q, a code C is a $k-$dimensional subspace of $\mathbb{F}_q^n$, where $k < n$. The codewords in C can be obtained as $C = \{\mathbf{c} \in \mathbb{F}_q^n | \mathbf{c}\mathbf{H}^\top = \mathbf{0}\}$, where $^\top$ denotes transposition, and $\mathbf{H} \in \mathbb{F}_q^{r \times n}$ is a full-rank matrix of size $r \times n$, where $r = n - k$, and is known as the parity-check matrix. The code rate R is defined as $R = \frac{k}{n}$. For the rest of this paper, we assume that $q = 2$. The number of codewords of weight w is denoted as $A(w)$. The minimum distance of the code, d, is the smallest positive value of w such that $A(w) > 0$.

LDPC codes are a special type of code characterized by parity-check matrices with a relatively small number of non-zero entries compared to the number of zeros. The RCC in the parity-check matrix specifies that no closed length-4 cycle is formed in the corresponding Tanner graph [15], i.e., the parity-check matrix has no groups of four non-zero entries at the vertices of a rectangle. It is well known that soft-decision decoding algorithms commonly used for LDPC codes, such as the sum-product algorithm [17], encounter convergence issues when the parity-check matrix contains the aforementioned length-4 cycles. The girth of a code is the length of the shortest cycle(s) in the code Tanner graph.

A linear block code is cyclic if the cyclic-shift of each of its codewords is also a codeword. To obtain the generator polynomial $g(x)$ of a cyclic code from a parity-check polynomial

$$h(x) = h_0 + h_1 x + \ldots + h_k x^k,$$

that is a factor of $x^N + 1$ in $\mathbb{F}_2[x]$, it is sufficient to divide $x^N + 1$ by $h^*(x)$, which is also a factor of $x^N + 1$ [7]. If $h(x)$ is a primitive polynomial, the code obtained by taking it as the parity-check polynomial is a cyclic simplex code with a block length $N = 2^k - 1$ and $r_s = N - k$. Therefore, this simplex code is formed by the all-zero codeword and all the cyclic shifts of any non-zero codeword. The following important properties hold.

Property 1: Given any non-zero k-tuple $\mathbf{z}$ and a non-zero codeword $\mathbf{t}$ of the cyclic simplex code, there is exactly one set of k consecutive entries of $\mathbf{t}$ (including those wrapping around) which is equal to $\mathbf{z}$.

Property 2: Any non-zero codeword of the cyclic simplex code is a pseudo-noise sequence of maximum period $N = 2^k - 1$, since $h(x)$ is primitive [18]

We point out that, in the rest of the paper, when looking for specific k-tuples in the pseudo-noise sequence, the latter always "wraps around", i.e., its first entry and its last entry are considered consecutive. We denote the pseudo-noise sequence associated to the codeword(s) of the cyclic simplex code as $\mathbf{p}$.

The parity-check matrix $\mathbf{H}$ of the simplex code has $2^k - 1 - k$ rows, where $\mathbf{h}$, i.e., the coefficients vector of $h(x)$, shifts from left to right by one position for each row. We define the support of $\mathbf{h}$ as a vector $\mathbf{e}$ with w_h elements. The vector $\mathbf{e}$ contains all $\{i \in [0, k] | h_i = 1\}$ in ascending order. We also define a vector $\mathbf{s}$ with $w_h - 1$ elements, where $s_i = e_{i+1} - e_i$, $i \in \{0, \ldots, w_h - 2\}$. It holds that

$$\sum_{i=0}^{w_h - 2} s_i = k.$$

In the following, we will call *separations* the entries of $\mathbf{s}$ and distinguish between *external* separations (s_0 and s_{w_h-2}) and *internal* separations (s_i with $i \in [1, w_h - 3]$). It is easy to observe that the maximum number of distinct separations between (not necessarily consecutive) pairs of non-zero entries in $\mathbf{h}$ is $\binom{w_h}{2}$. Finally, the largest entry of $\mathbf{s}$ is denoted as $s_{\max}$.

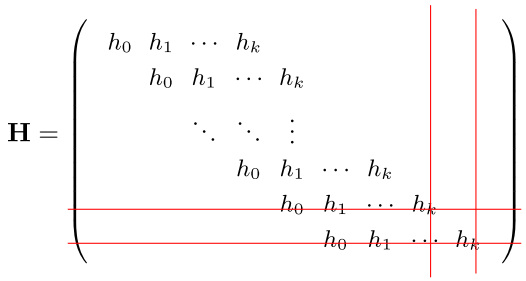

Fig. 1. Effect of puncturing on the parity-check matrix.

III. Design Principles of PRC-LDPC Codes

In this section we describe the main principles and requirements of our code design.

A. Parity-Check Matrix of PRC-LDPC Codes

From the parity-check matrix of the original simplex code, for which $n = N = 2^k - 1$ and therefore $R = \frac{k}{2^k - 1}$, puncturing can be used to obtain the parity-check matrix of a code with larger rate. An example is shown in Fig. 1, where the last two rows (and columns) have been eliminated. After performing a certain number of puncturing operations, say x, the parity-check matrix is reduced to $r = r_s - x$ rows and $n = r + k = N - x$ columns. For a given $h(x)$ of degree k, the PRC-LDPC code family it defines contains all the PRC-LDPC codes with block length ranging from $k+1$ to $2^k - 1$. Referring to Fig. 1, we notice that the non-zero codewords of the punctured code, whose number is $2^k - 1$ as in the original simplex code, can be obtained as the vectors selected by a sliding window of length $n < 2^k - 1$ that circularly spans over the Pseudo-Noise (PN) sequence introduced in Property 2, i.e., circularly spans over a non-zero codeword of the parent simplex code. This unique origin of the codewords enables us to investigate the generation of low-weight codewords.

Let us discuss the requirements that $h(x)$ must fulfill in order to define a PRC-LDPC code.

B. Requirements for the Parity-Check Polynomial

First of all, since the design starts from simplex codes, $h(x)$ is a primitive polynomial. Differently from previous works, in this paper we need to tighten the requirements on $h(x)$. The following result holds.

Theorem 1: A necessary and sufficient condition for the satisfaction of the RCC in a PRC-LDPC code is that the support $\mathbf{e}$ is a Golomb ruler [1]

Therefore, in the following we choose $h(x)$ in such a way that Theorem 1 holds. This way, a fundamental condition required for effective convergence of the iterative decoders commonly used to decode LDPC codes is satisfied.

Theorem 2: The girth of any PRC-LDPC code with $n \geq k + s_0 + 1$ is upper bounded by 6.

Proof: Let us consider $n = N$. By construction, we have that $h_{0,0} = h_{0,s_0} = h_{0,k} = 1$. Since the parity-check matrix is formed by shifts of $\mathbf{h}$ towards right, we also have that $h_{0+s_0,0+s_0} = h_{0+s_0,k+s_0} = 1$, as well as $h_{0+k,0+k} = h_{0+k,s_0+k} = 1$. Therefore, $h_{0,s_0} \rightarrow h_{0,k} \rightarrow h_{k,k} \rightarrow h_{k,s_0+k} \rightarrow h_{s_0,s_0+k} \rightarrow h_{s_0,s_0} \rightarrow h_{0,s_0}$ is an unavoidable cycle of length 6. This unavoidable cycle involves the first $k+s_0+1$ columns of $\mathbf{H}$, and thus also exists in all the codes of the family for which $n \geq k+s_0+1$. ■

At this point, we make some considerations on the sparsity of the code and of $h(x)$. Parity-check matrices of PRC-LDPC codes are row-regular, since all the rows have Hamming weight equal to w_h, but not column-regular. The average column weight is

$$\langle w_c \rangle = \frac{r w_h}{n} = (1-R) w_h. \tag{1}$$

This parameter plays a crucial role in terms of iterative decoding complexity, which increases for increasing values of $\langle w_c \rangle$. Therefore, in order to keep a low decoding complexity, for a given value of R, we will choose a relatively small value of w_h. Complexity issues will be discussed more in depth in Section V.

We remark that having a small w_h may not be sufficient to guarantee that $\mathbf{e}$ is a Golomb ruler. For example, suppose that $k = w_h = 5$. Then, even though $\langle w_c \rangle$ is small, $h(x) = 1 + x + x^3 + x^4 + x^5$ and $\mathbf{e} = [0, 1, 3, 4, 5]$ is not a Golomb ruler, leading to a lot of length-4 cycles and thus a potentially bad performance under iterative decoding. We thus need to include the condition that w_h is relatively small compared to k. In particular, a necessary, but not sufficient, condition for the fulfilment of the RCC derives from the following statement.

Corollary 1: Given a parity-check polynomial $h(x)$, if $\binom{w_h}{2} > k$ then the RCC does not hold for any PRC-LDPC code described by $h(x)$.

Proof: As noted in Section II, a polynomial $h(x)$ of weight w_h yields $\binom{w_h}{2}$ positive separations between its exponents. In order to satisfy the RCC, based on Theorem 1, we need all of them to be different. If the degree of $h(x)$ is k, then the separations must take values in $[1, k]$ (which obviously contains k elements). Therefore, the pigeonhole principle tells us that if $\binom{w_h}{2} > k$, i.e., if there are more separations than available different values, then at least two separations are equal and, therefore, the RCC is not satisfied. ■

We remark that, in order to satisfy all the needed constraints, i.e., to find a polynomial that is both primitive, sparse and such that $\mathbf{e}$ is a Golomb ruler, the values of w_h are not necessarily tight to the bound given by Corollary 1. A visual summary of the concepts discussed in this section is depicted in Fig. 2, where the searched polynomials live in the gray region.

C. Shortening Operations: A Further Degree of Freedom

Aiming to gain an even finer rate adaptability, it is possible to apply code shortening. This operation consists in eliminating z information symbols. Therefore, the shortened code has block length $n' = n - z$ and $k' = k - z$ information symbols.

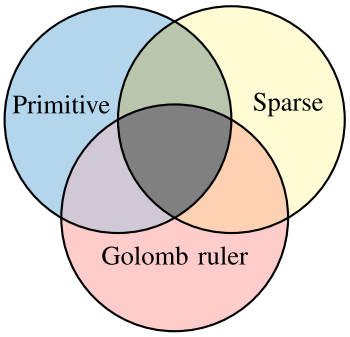

Fig. 2. Requirements of the parity-check polynomial.

$$\mathbf{H} = \begin{pmatrix} h_0 & h_1 & \cdots & h_k & & & & \\ & h_0 & h_1 & \cdots & h_k & & & \\ & & \ddots & \ddots & \vdots & & & \\ & & & h_0 & h_1 & \cdots & h_k & \\ & & & & h_0 & h_1 & \cdots & h_k \\ & & & & & h_0 & h_1 & \cdots & h_k \end{pmatrix}$$

Fig. 3. Effect of shortening on the parity-check matrix.

A simple example is shown in Fig. 3, where the first two columns have been eliminated.

IV. Code Design Approach

In this section we provide sufficient conditions for the existence of low-weight codewords in PRC-LDPC codes and convert them into a practical approach to code design.

A. Low-Weight Codewords

We have mentioned in Section II that any non-zero codeword of the parent cyclic simplex code corresponds to a shifted version of the same PN sequence $\mathbf{p}$ of length $N = 2^k - 1$.

Remark 1: When puncturing is applied, and a code of length $n < N$ is obtained, the code is no longer cyclic and each codeword is a different n-tuple of consecutive symbols of $\mathbf{p}$. In other words, it is possible to consider a sliding window of size n spanning over $\mathbf{p}$. Each of the possible N initial positions of the sliding window covers a different non-zero codeword.

Remark 1 is fundamental for our theoretical analysis and its implications will be used many times in the following. Therefore, we provide a toy example to make it clearer.

Example 1: Consider the $(7, 3)$ cyclic symplex code characterized by the parity-check polynomial $h(x) = 1 + x^2 + x^3$. The corresponding PN sequence is given by $g(x) = 1 + x + x^2 + x^4$ followed by $N - r - 1 = 2$ zeros, i.e.,

$$\mathbf{p} = [1, 1, 1, 0, 1, 0, 0].$$

Let us suppose that we puncture two code symbols, obtaining a $(5, 3)$ code. The $2^k - 1 = 7$ non-zero codewords of this code

can be obtained by letting a window of size 5 cyclically slide over $\mathbf{p}$*, for all possible initial positions, that is,*

$$\begin{aligned}
&[\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{0},\mathbf{1},0,0],\\
&[1,\mathbf{1},\mathbf{1},\mathbf{0},\mathbf{1},\mathbf{0},0],\\
&[1,1,\mathbf{1},\mathbf{0},\mathbf{1},\mathbf{0},\mathbf{0}],\\
&[\mathbf{1},1,1,\mathbf{0},\mathbf{1},\mathbf{0},\mathbf{0}],\\
&[\mathbf{1},\mathbf{1},1,0,\mathbf{1},\mathbf{0},\mathbf{0}],\\
&[\mathbf{1},\mathbf{1},\mathbf{1},0,1,\mathbf{0},\mathbf{0}],\\
&[\mathbf{1},\mathbf{1},\mathbf{1},\mathbf{0},1,0,\mathbf{0}],
\end{aligned}$$

where the codewords of the punctured code are marked in bold.

Keeping these concepts in mind, we are interested in finding sub-sequences of $\mathbf{p}$ that may cause the existence of low-weight codewords. We first focus on the following two tuples, called $\mathbf{T}_1$ and $\mathbf{T}_2$ in the following, respectively:

- a k-tuple containing $k-1$ zeros and a single one;
- a $(k+1)$-tuple coinciding with $\mathbf{h}^*$.

While the existence of $\mathbf{T}_1$ in $\mathbf{p}$ is implied by Property 1, next we prove a sufficient condition for the existence of $\mathbf{T}_2$ in $\mathbf{p}$.

Theorem 3: Given a primitive parity-check polynomial $h(x)$ *defining a cyclic simplex code of length* N*, if* $\mathbf{e}$ *is a Golomb ruler, then* $\mathbf{p}$ *contains a* $(k+1)$*-tuple coinciding with* $\mathbf{h}^*$.

Proof: Let us evaluate the quantity $\mathbf{h}^*\mathbf{h}^\top \mod 2$. If k is odd we have that

$$\mathbf{h}^*\mathbf{h}^\top \mod 2 = \bigoplus_{i=0}^{k} h_i h_{k-i} = 0, \tag{2}$$

since the sum contains an even number of pairs of equal terms.

If k is even, instead, we have that $\mathbf{h}^*\mathbf{h}^\top$ is

$$\bigoplus_{i=0}^{k} h_i h_{k-i} = h_0 h_k + \cdots + h_{k/2}h_{k/2} + \cdots + h_k h_0, \tag{3}$$

which is $0 \mod 2$ only if $h_{k/2} = 0$. If $\mathbf{e}$ is a Golomb ruler, $h_{k/2}$ is necessarily 0. In fact, $h_{k/2} = 1$ implies that h_0 and $h_{k/2}$ are at the same distance of h_k and $h_{k/2}$, which contravenes the definition of Golomb ruler for $\mathbf{e}$. In summary, if $\mathbf{e}$ is a Golomb ruler, we have that $\mathbf{h}^*\mathbf{h}^\top = 0 \mod 2$.

Given any codeword $\mathbf{t}$ of the cyclic simplex code, since the rows of the parity-check matrix are formed by cyclic shifts of the vector $[\mathbf{h}, \mathbf{0}_{N-k-1}]$, from $\mathbf{t}\mathbf{H}^\top = \mathbf{0}$ it follows that

$$\bigoplus_{i=0}^{k} t_i h_i = 0. \tag{4}$$

From the comparison with (2) and (3) we notice that, if $\mathbf{e}$ is a Golomb ruler, we can substitute $t_i = h_i^*$, for $i \in [0,k]$. Therefore, any codeword of the cyclic simplex code contains $\mathbf{h}^*$ and, as codewords are cyclically shifted versions of $\mathbf{p}$, also $\mathbf{p}$ contains $\mathbf{h}^*$, proving the thesis. ■

We now show that, under certain circumstances, the relative position of $\mathbf{T}_1$ and $\mathbf{T}_2$ can be predicted.

Lemma 1: If, for a family of PRC-LDPC codes,

$$s_0 > \sum_{j\in[1,w_h-2]} s_j, \quad \text{or} \quad s_{w_h-2} > \sum_{j\in[0,w_h-3]} s_j,$$

and $\mathbf{e}$ *is a Golomb ruler, then* $\mathbf{T}_1$ *and* $\mathbf{T}_2$ *are consecutive in* $\mathbf{p}$.

Proof: By definition of simplex codes, $\mathbf{p}$ is given by

$$\mathbf{p} = [\mathbf{g}, \mathbf{0}_{k-1}],$$

where $\mathbf{g}$ is the coefficients vector of the generator polynomial, and $\mathbf{0}_{k-1}$ is a $(k-1)$-tuple of zeros. Thus, since $g_0 = g_{n-k-1} = 1$, in order to prove the thesis we just need to show that, under the above hypotheses, either $\mathbf{g}$ begins or ends with $\mathbf{h}^*$ (i.e., $\mathbf{T}_2$), which is for sure contained in $\mathbf{p}$ due to Theorem 3. Let us suppose that

$$s_{w_h-2} > \sum_{j\in[0,w_h-3]} s_j, \tag{5}$$

that is, the first separation between the exponents of $h(x)$ is larger than the sum of all the others. This implies that the last separation between the exponents of $h^*(x)$ is larger than the sum of all the others. To study the structure of $\mathbf{g}$ we can perform classical polynomial division between $x^N + 1$ and $h^*(x)$. The first partial remainder of the division is

$$r_0(x) = x^{N-k}h^*(x) + x^N = \sum_{i=0}^{w_h-2} x^{N-\sum_{j=0}^{i} s_j}$$

(in $\mathbb{F}_2[x]$), and the first quotient is obviously $q_0(x) = x^{N-k}$. The largest and the smallest exponents of $r_0(x)$ are therefore x^{N-s_0} and x^{N-k}, respectively. Then, we notice that the next term of the quotient must be x^{N-k-s_0} and that the corresponding $h^*(x)x^{N-k-s_0}$ is in the form x^{N-s_0} plus some terms with degree lower than x^{N-k}, due to (5). The same reasoning holds for the remaining w_h-2 terms of the quotient. Summarizing, the first w_h terms of the quotient (that is $g(x)$) are

$$x^{N-k}, x^{N-k-s_0}, x^{N-k-(s_0+s_1)}, \ldots, x^{N-2k},$$

that is a shifted version of $h^*(x)$ (in particular, it is $x^{N-2k}h^*(x)$). Therefore, the first $k+1$ bits of $\mathbf{g}$ are $\mathbf{h}^*$, implying that $\mathbf{T}_2$ follows $\mathbf{T}_1$ in $\mathbf{p}$. The specular case, in which

$$s_0 > \sum_{j\in[1,w_h-2]} s_j,$$

is perfectly equivalent, but it is convenient to perform the polynomial division from right to left. In this case, the last $k+1$ symbols of $\mathbf{g}$ are $\mathbf{h}^*$ and $\mathbf{T}_2$ precedes $\mathbf{T}_1$ in $\mathbf{p}$. ■

Lemma 1 has important implications on the minimum distance of some codes.

Corollary 2: Given any PRC-LDPC code of rate $R \geq \frac{1}{2}$*, if*

$$s_0 > \sum_{j\in[1,w_h-2]} s_j, \quad \text{or} \quad s_{w_h-2} > \sum_{j\in[0,w_h-3]} s_j,$$

and satisfying the RCC, then $d \leq w_h$.

Proof: It follows from Lemma 1 that, under its hypotheses, $\mathbf{p}$ contains a $2k$-tuple of weight w_h, formed by the

concatenation of the $k-1$ zeros in $\mathbf{T}_1$ and $\mathbf{T}_2$ (of size $k+1$ and weight w_h). Therefore, since the codewords of PRC-LDPC codes can be covered by a window of size n sliding over $\mathbf{p}$, all the PRC-LDPC codes of block length $n \leq 2k$ satisfying the hypotheses of Lemma 1 contain at least one codeword of weight $\leq w_h$. ■

In order to better understand the role of large external separations, we deepen the analysis for the case of $R = \frac{1}{2}$, and define a sufficient condition for the existence of codewords of weight w_h when an external separation is larger than the sum of the internal ones.

Lemma 2: In a PRC-LDPC code with $R = \frac{1}{2}$ satisfying the RCC, if for either $i = 0$ or $i = w_h - 2$,

$$s_i > \sum_{j \in [1, w_h - 3]} s_j,$$

then there are at least $s_i - \sum_{j \in [1, w_h-3]} s_j$ codewords of weight w_h.

Proof: The proof is the same as in [2, Lemma 2], and is thus omitted. ■

Following the above reasoning, we can also analyze the case in which $s_0 + s_{w_h-2}$ exceeds the sum of the internal separations, and also the case in which an internal separation exceeds the sum of all the others. These two conditions cannot be simultaneously true. The following two lemmas can be proved using similar arguments as those used in the proof of Lemma 2, and their proofs are hence omitted for brevity.

Lemma 3: In a PRC-LDPC code with $R = \frac{1}{2}$ satisfying the RCC, if

$$s_0 + s_{w_h-2} > \sum_{j \in [1, w_h - 3]} s_j, \tag{6}$$

then there are at least $s_0 + s_{w_h-2} - \sum_{j \in [1, w_h-3]} s_j$ codewords of weight w_h.

Lemma 4: In a PRC-LDPC code with $R = \frac{1}{2}$ satisfying the RCC, if there exists $i \in [1, w_h - 3]$ such that

$$s_i > \sum_{j \neq i} s_j,$$

then there are at least $s_i - \sum_{i \neq j} s_j$ codewords of weight w_h.

It should be noted that Lemmas 2, 3 and 4 remain applicable even for $R > \frac{1}{2}$. Specifically, if a particular code satisfies $d > w_h$ at $R = \frac{1}{2}$ and is subsequently punctured to attain $d = w_h$, then the methodologies outlined in Lemmas 3 and 4 can be utilized to determine the multiplicity of low-weight codewords.

We remark that the hypotheses of Lemmas 1 to 4 may imply each other. For example, the hypotheses of Lemma 1 imply those of Lemma 2. In Fig. 4 we provide an illustrative scheme for the cases in which $d \leq w_h = 5$, and thus $|\mathbf{s}| = 4$. The gray region contains all the sufficient conditions causing the minimum distance to be upper bounded by w_h.

Based on the above considerations, some further good practices, besides the strict rules provided in Section III, are given in the following remark.

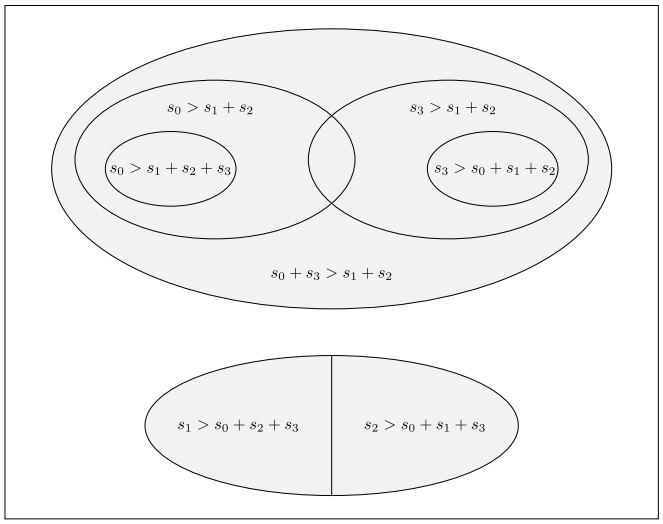

Fig. 4. Summary of conditions causing $d \leq w_h$, when $w_h = 5$.

Remark 2: When designing PRC-LDPC codes, $h(x)$ should be chosen such that:

- *for either $i = 0$ or $i = w_h - 2$,*
$$s_i \leq \sum_{j \in [1, w_h - 3]} s_j;$$
- *for $i \in [1, w_h - 3]$,*
$$s_i \leq \sum_{j \in [0, w_h - 2], j \neq i} s_j;$$
- *it holds that*
$$s_0 + s_{w_h-2} \leq \sum_{j \in [1, w_h - 3]} s_j.$$

At this point, it is interesting to study the structure of $\mathbf{p}$ when none of the sufficient conditions causing $d \leq w_h$ are satisfied. By computing $g(x)$ as the polynomial division of $x^N + 1$ by $h^*(x)$, with the same arguments used in the proof of Lemma 1, it is straightforward to prove that the region around $\mathbf{T}_2$ appears as shown in Fig. 5, if the RCC is satisfied, along with some further requirements discussed in the following. Specifically, on either side of $\mathbf{h}^*$, there exist two regions containing only zeros, of size Z_0 and Z_1, respectively. The size of these two regions is computed in the following theorem.

Theorem 4: Given a PRC-LDPC code characterized by a certain $h(x)$ of degree k, for which the conditions in Remark 2 are satisfied, by defining i^* as the index for which $s_{i^*} = s_{\max}$, we have that

$$Z_0 = 2\Big(\sum_{i=i^*}^{w_h-2} s_i\Big) - k - 1, \quad Z_1 = 2\Big(\sum_{i=0}^{i^*} s_i\Big) - k - 1.$$

Proof: The proof follows the same idea of the proof of Lemma 1. Therefore, for saving space, we omit it. ■

It easily follows from Theorem 4 that

$$Z_0 + Z_1 = 2(s_{\max} - 1),$$

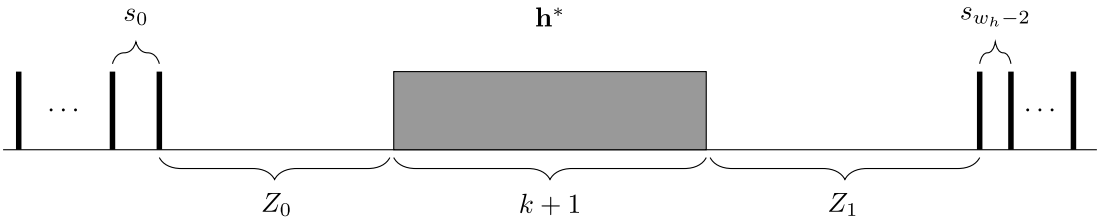

Fig. 5. Structure of $\mathbf{p}$ around $\mathbf{h}^*$.

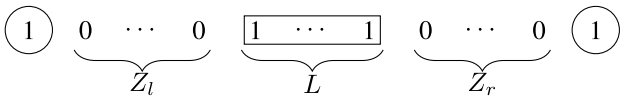

Fig. 6. Portion of $\mathbf{p}$ describing a family of codewords.

which has the very interesting implication that the total width of the all-zero region around $\mathbf{h}^*$ depends only on the largest separation between the exponents of $h(x)$.

Then, under the hypotheses of Theorem 4, next to the all-zero region of size Z_0, there are some symbols 1, each at distance s_{w_h-2} from the other. Similarly, consecutive to the all-zero region of size Z_1, there are some symbols 1, each at distance s_0 from the other.

Let us now convert the above reasoning into conditions on the minimum distance.

Corollary 3: Given any PRC-LDPC code of block length $n \leq 2s_{\max} + k - 1$ satisfying the hypotheses of Theorem 4, then $d \leq w_h$.

Proof: The proof follows from the equality

$$Z_0 + Z_1 + k + 1 = 2\Big(\sum_{i=0}^{i^*} s_i + \sum_{i=i^*}^{w_h-2} s_i\Big) - k - 1 = 2k + 2s_{\max} - k - 1 = 2s_{\max} + k - 1.$$

All the windows of size $n \leq Z_0 + Z_1 + k + 1$ cover at least one codeword of weight $\leq w_h$. ∎

Fig. 5 also provides a further confirmation that the external separations should not be too large. If this condition is met, sliding windows of size n slightly larger than $Z_0 + Z_1 + k + 1$ would cover codewords with relatively large weight.

B. Families of Codewords

In this section we make some further considerations on the minimum distance of PRC-LDPC codes. For a given n, we say that if two codewords can be obtained from each other by a non-cyclic shift, then they belong to the same *family*. A family of codewords derives from a portion of $\mathbf{p}$, as shown in Fig. 6. Any window of size n starting in the all-zero region of size Z_l and ending in that of size Z_r covers codewords belonging to the same family.

The family of codewords (say the i-th and composed by codewords of weight w) defined by the portion within the rectangle in Fig. 6 exists for $n \in [L, L + Z_l + Z_r]$. It is interesting to count $A_i(w)$ for each n. Supposing, without loss of generality, that $Z_l \geq Z_r$, and calling $Z_{rl} = Z_r + Z_l$, we have that

$$A_i(w) = \begin{cases} n - L + 1 & \text{if } L \leq n \leq L + Z_r, \\ 1 + Z_r & \text{if } L + Z_r < n \leq L + Z_l, \\ 1 + Z_{rl} + L - n & \text{if } L + Z_l < n \leq L + Z_{rl}, \\ 0 & \text{elsewhere.} \end{cases} \tag{7}$$

We also introduce the quantities $r(d)$ and $n(d)$ which, given a parity-check polynomial $h(x)$, are the minimum number of rows and columns of $\mathbf{H}$ such that the corresponding PRC-LDPC code has distance d. To establish the starting point of our analysis, let us begin by examining an extreme scenario. While the codes described in Lemmas 5 and 6 may not be practical themselves, they play a valuable role in identifying regions within $\mathbf{p}$ where low-weight codewords of codes with larger block length can potentially originate.

Lemma 5: A PRC-LDPC code satisfying the RCC has $d = 1$ if and only if $r < r_{\min} = s_{\max}$.

Proof: Let us suppose that the two entries of $\mathbf{h}$ separated by $s_{\max}$ are h_i and h_j. Since the RCC is satisfied, there exists only one such pair. Then, there are $s_{\max} - 1$ zeros separating h_i and h_j. In order to avoid the occurrence of all-zero columns in the parity-check matrix, r must necessarily be at least equal to $s_{\max}$. This way, all the columns from the $(i+1)$-th to the $(j-1)$-th contain at least one symbol one, that is h_i shifting to the right for increasing values of r. If $r = \hat{r} < s_{\max}$, instead, due to the diagonal form of the parity-check matrix, $s_{\max} - \hat{r}$ columns are zero, i.e., those ranging from the $(i + \hat{r} - 1)$-th to the $(j - 1)$-th. The same reasoning can be applied to the other smaller separations. Therefore, if $r < s_{\max}$, the parity-check matrix contains columns of weight 0, implying that $d = 1$; if $r \geq s_{\max}$, instead, there are no all-zero columns and $d \geq 2$. ∎

In other words, Lemma 5 states that $r(2) = s_{\max}$, i.e., that $n(2) = k + s_{\max}$.

Lemma 6: Let i^* be the index for which $s_{i^*} = s_{\max}$. In a PRC-LDPC code of block length $n(2)$, there are two families of codewords of weight 2:

- *one such that the separation between the non-zero symbols is $\sum_{i=0}^{i^*} s_i$, called Family 1,*
- *the other such that the separation between the non-zero symbols is $\sum_{i=i^*}^{w_h-2} s_i$, called Family 2.*

Proof: Suppose again that the entries of $\mathbf{h}$ separated by $s_{\max}$ are h_i and h_j, $j > i$. Given that the RCC is fulfilled, there can be only one pair of this kind. By definition, the closest non-zero symbol to the left of h_j is h_i, and it holds that $j - i = s_{\max}$. When $n = n(2)$, implying $r = s_{\max}$, we have

that $h_{0,i} = h_{1,i+1} = \ldots = h_{s_{\max}-1,i+s_{\max}-1} = h_i = 1$. Since $j > i+s_{\max}-1$, we have that the only non-zero element in the j-th column is $h_{1,j}$. Therefore, the 0-th and the j-th column of $\mathbf{H}$ are equal, implying the existence of a codeword of weight 2, where the two symbols 1 are separated by $j = s_0 + \ldots + s_{\max}$ positions, proving the thesis. The second part of the lemma is proved with identical arguments. ∎

By performing polynomial division between $x^N + 1$ and $h^*(x)$, it can be readily noticed that the two families described in Lemma 6 appear consecutively in $\mathbf{p}$.

The following straightforward result holds for general values of d.

Lemma 7: Consider any codeword $\mathbf{c}$ of weight w of a PRC-LDPC code $\mathcal{C}$ with length n. Also consider the PRC-LDPC code with length $n' = n - 1$ obtained by puncturing $\mathcal{C}$, called $\mathcal{C}'$. Then, the vector obtained by removing the last entry of $\mathbf{c}$ is a codeword for $\mathcal{C}'$.

Proof: It is a straightforward consequence of Remark 1. ∎

Lemma 7 implies that, when puncturing a symbol of any PRC-LDPC code, either the minimum distance is preserved, or it decreases by at most one unit. Therefore, the minimum distance follows a step-like behavior with respect to n, and the height of the steps is always unitary, whereas the width depends on the given code family (see Fig. 8, introduced later).

In the following we prove some other useful results on the weight distribution. According to the definition of family of codewords, we can state that

$$A(w) = \sum_i A_i(w),$$

where $A_i(w)$ is the number of codewords of weight w belonging to the i-th family.

Theorem 5: If the PRC-LDPC code of length n is characterized by a certain $A_i(w) = A$, for any i, then the PRC-LDPC code of length $n' = n+1$ has either $A_i(w) = A$, or $A_i(w) = \max\{A-1, 0\}$, or $A_i(w) = A+1$.

Proof: As evident from (7), a unitary increase in n cannot cause the number of codewords in a given family to increase or decrease by more than one unit. ∎

Theorem 6: If, for a certain value of i, the PRC-LDPC code of length n, called $\mathcal{C}$, exhibits $A_i(d) = 1$, then the PRC-LDPC code of length $n' = n + 1$, called $\mathcal{C}'$, has $A(d+1) \geq 2$. If we call $\mathbf{c}$ the codeword of $\mathcal{C}$ belonging to family i and having weight d, then $\mathcal{C}'$ contains $[\mathbf{c}, 1]$ and $[1, \mathbf{c}]$ as codewords.

Proof: Under the assumption that we have only one codeword of weight d left in the i-th family, called $\mathbf{c}$, then it is very easy to show that the first and the last symbols of $\mathbf{c}$ are 0's; furthermore, in $\mathbf{p}$, the two symbols immediately to the right and left of $\mathbf{c}$ must be 1's. Any other configuration breaks either the definition of minimum distance or the assumption that the codeword is unique for the i-th family. Given this, when considering a larger code length $n' = n + 1$, then the larger window will cover the symbol 1 on the right, forming a codeword of weight $d+1$. Similarly, the symbol 1 on the left will be covered, forming a codeword of weight $d+1$, as well. Both these codewords belong to different families than $\mathbf{c}$, since they include a larger number of symbols one. This proves the thesis. ∎

Corollary 4: If, for a certain value of i, the PRC-LDPC code of block length n shows $A_i(d) = 1$ then the PRC-LDPC code of length $n' = n + 2$ has $A(d+2) \geq 1$. If we call $\mathbf{c}$ the codeword of $\mathcal{C}$, then $\mathcal{C}'$ contains $[1, \mathbf{c}, 1]$ as a codeword.

Proof: Same as Theorem 6, but the window covers both the symbols 1 around $\mathbf{c}$. ∎

All the above results can be used to find the low-weight codewords of PRC-LDPC codes. In particular, our method relies on searching low-weight codewords in the portions of $\mathbf{p}$ that we have shown to be sparse. Some examples are provided next, where we choose parity-check polynomials with a relatively small degree k. We compute the weight distribution of the resulting family of PRC-LDPC codes, validating the theoretical analysis. The reason for the choice of small values of k is twofold: on the one hand, they lead to a more comprehensible treatment; on the other hand, we want to compare our results with the actual minimum distance profile, which would be uncomputable for large values of k.

Example 2: Let us start from the family of PRC-LDPC codes obtainable from

$$h(x) = 1 + x + x^5 + x^{11} + x^{13}.$$

We have that $\mathbf{e} = [0, 1, 5, 11, 13]$ is a Golomb ruler, and the separations vector is $\mathbf{s} = [1, 4, 6, 2]$. We thus notice that the conditions in Remark 2 are satisfied.

By performing division between x^N+1 and $h^*(x)$ we obtain the following portion of $\mathbf{p}$

$$\ldots|1111|00000000|\mathbf{10100000100011}|00|1010101|\ldots,$$

where we identify $\mathbf{h}^*$, marked in bold, surrounded by two all-zero portions of size $2(1+4+6) - 13 - 1 = 8$ and $2(6+2) - 13 - 1 = 2$, and two comb-like zones, where symbols 1 are distant by $s_0 = 1$ positions on the left and $s_{w_h-2} = 2$ positions on the right. This corroborates the analysis in Section IV and, in particular, in Fig. 5. The shown sub-sequence already tells us that $n(6) > 24$, but it is possible to study the distance profile even more in detail. The analysis of the low-weight codewords by studying the portion of $\mathbf{p}$ described by Lemma 6 can be found in Appendix. The reasoning in Appendix already shows that the weight distribution of PRC-LDPC codes, at least for relatively small values of n, can be characterized starting from a very thin portion of $\mathbf{p}$. In this particular case, such a portion of $\mathbf{p}$ is that defined by Lemma 6. Let us slightly widen this portion, i.e., let us consider

$$000010101010100001000110000000000100000001010010 11.$$

It is interesting to notice that all the minimum weight codewords of all the PRC-LDPC codes with $19 \leq n \leq 37$, except for one of them when $n = 21$, are contained into this portion of $\mathbf{p}$, whose length is only 0.6% of the total length of the PN sequence, i.e., $2^k - 1$.

In Fig. 7 we compare the actual $A(d)$ of the codes in Example 2, for different values of n, to that obtained using our method (looking around the small portion of $\mathbf{p}$ specified by Lemma 6) and also to the average value, computed as in [14].

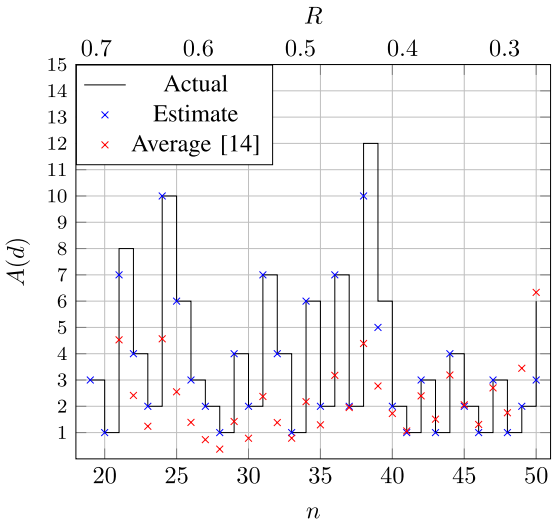

Fig. 7. Comparison of $A(d)$ for the PRC-LDPC codes in Example 2, with $k = 13$.

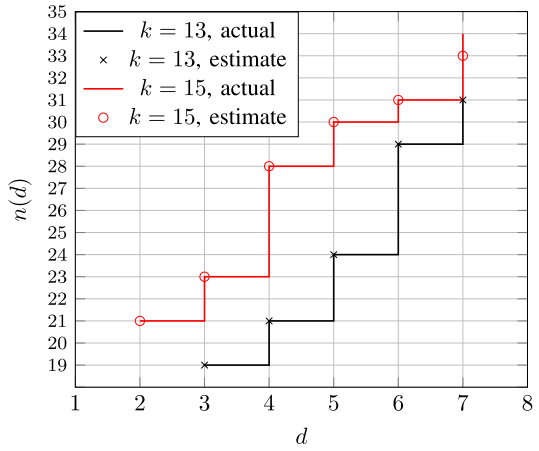

Fig. 8. Minimum distance profile of the PRC-LDPC codes in Examples 2 and 3.

We notice that, for all the considered values, our estimate almost perfectly matches the actual value of $A(d)$. The values of d, which depend on n, are deducible from Fig. 8, introduced in Example 3. Namely, $d = 3$ when $n = 19$, and then each increase of $A(d)$ corresponds to a unitary increase of d. For the codes in Example 2, the average estimator (working for $d \geq 3$) provides estimates that are, in some cases, either significantly smaller or significantly larger than the actual value.

Example 3: We consider the family of PRC-LDPC codes obtainable from

$$h(x) = 1 + x^2 + x^8 + x^{12} + x^{15},$$

which implies $\mathbf{e} = [0, 2, 8, 12, 15]$, *and then* $\mathbf{s} = [2, 6, 4, 3]$.

For the sake of brevity, in this example we do not list all the codewords as in Example 2 (and in Appendix), since the rationale is exactly the same. We only mention that the portion of $\mathbf{p}$ *around the families introduced in Lemma 6, which is*

$$\ldots 01110000101000000010000000000010110 \ldots,$$

contains all the minimum distance codewords of the PRC-LDPC codes with $n = \{22, 26, 27, 28, 29\}$. *More in general, somehow surprisingly, this portion of* $\mathbf{p}$ *consists of* 0.1% *of the total length of* $\mathbf{p}$ *and contains* 59% *of the minimum distance codewords of all the PRC-LDPC codes derived from the above* $h(x)$, *with* $21 \leq n \leq 33$.

We show in Fig. 8 the initial values of $n(d)$ for the codes in Examples 2 and 3. It is also remarkable that, if the only goal of the analysis is to estimate the minimum distance (and not enumerating the number of minimum distance codewords), our method basically provides exact matching in all the considered cases, except for a single discrepancy when $k = 15$ and $n = 33$, where the estimated distance is 7, instead of 6.

C. Code Design

In this section we describe the method we employ to find parity-check polynomials fulfilling the properties described in Section III and pictorially summarized in Fig. 2.

Let us suppose that an (n, k) code must be obtained and the target row-weight of the parity-check matrix is w_h. The design procedure we adopt is as follows:

1) Pick a Golomb ruler, say $\mathcal{G}$, of order $L \gg w_h$, containing k among its marks.[1]
2) Remove all the marks larger than k, so obtaining a new Golomb ruler, say $\mathcal{G}'$, of order L'. If $L' < w_h$, go back to Step 1). Set $j = 0$.
3) Pick the j-th subset of size $w_f < w_h$ of the marks of $\mathcal{G}'$, say $S_f^{(j)}$. If the polynomial associated to these marks breaks the conditions in Remark 2, set $j = j + 1$ and restart this step. Set $i = 0$.
4) Let us call $S_r^{(j)}$ the complementary set of $S_f^{(j)}$. Pick the i-th subset of $S_r^{(j)}$ of size $w_h - w_f$, called $S_r^{(j,i)}$.
5) Test whether the polynomial

$$p^{(j,i)}(x) = \sum_{l \in \{S_f^{(j)} \cup S_r^{(j,i)}\}} x^l$$

is primitive and store $p^{(j,i)}(x)$ if it is. If $i = \binom{L'-w_f}{w_h-w_f} - 1$, set $j = j+1$ and go back to Step 3), else set $i = i+1$ and go back to Step 4).

This procedure generates all the primitive polynomials, if any, associated to a certain Golomb ruler $\mathcal{G}'$. It requires testing at most $\binom{L'}{w_f}\binom{L'-w_f}{w_h-w_f}$ polynomials for primitivity. It is evident that when the objective is to generate a single family of PRC-LDPC codes, the algorithm can be halted as soon as the first parity-check polynomial $p^{(j,i)}(x)$ is found.

$\mathcal{G}$ (and thus L') and w_f should be regarded as degrees of freedom in code design, as their selection impacts the trade-off between search complexity and the probability of finding a primitive polynomial within the search space.

The described procedure frequently yields many parity-check polynomials. The analytical insights from preceding sections are pivotal in distinguishing between better and worse families. In practical terms, in order to find good families of

[1]Golomb rulers with up to 65000 marks have been exhaustively designed and tabulated (see [19]). It is extremely unlikely that there exist practical values of the code dimension k that are not a mark in any of them. In any case, it is always possible to start from $k' > k$ and then adjust k' and the code rate by combining puncturing and shortening operations.

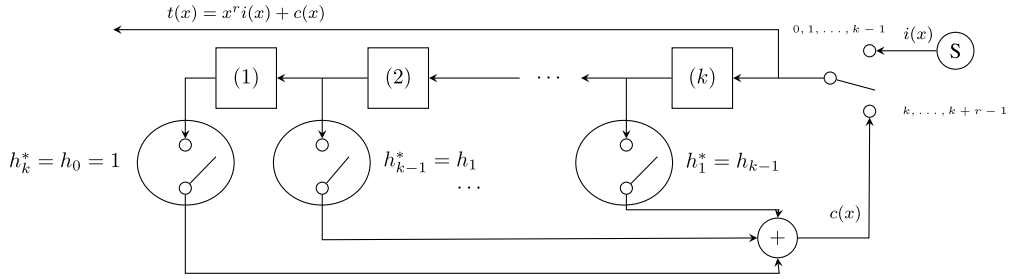

Fig. 9. Encoder circuit for PRC-LDPC codes.

PRC-LDPC codes, we first execute the above algorithm (built upon the principles in Section III), and then select the code families exhibiting the best characteristics in the desired block length range (e.g., largest minimum distance or, in case of candidates with the same $d_{\min}$, smallest $A(d)$), guided by the analytical conclusions from Sections IV-A and IV-B.

V. Encoding and Decoding Complexity

PRC-LDPC codes can be encoded by using an encoder circuit of the type introduced in [20] and later discussed in [14]. We report it, with the notation of this paper, in Fig. 9. The computational complexity of encoding a codeword of a PRC-LDPC code is thus $O(n)$. Such an encoding complexity is clearly smaller than the usual $O(n^2)$ complexity of general systematic encoding [21], based on multiplication of the information vector by the code generator matrix (which, in turn, can be derived from $\mathbf{H}$ through Gaussian elimination, costing $O(n^3)$). In the Richardson-Urbanke (RU) method [22] encoding is performed without using the generator matrix. The complexity of the encoding algorithm is $O(n+g^2)$, where g is the number number of rows of $\mathbf{H}$ that cannot be brought into triangular form by using only row and column permutations and is usually in the order of $\sqrt{n}$. Also QC-LDPC codes can be encoded by using shift registers, as reported, for example, in [23]. Many implementations of encoder design for QC-LDPC codes have been proposed, starting from [24]. However, these encoders usually require more than one shift register, due to the fact that, differently from our codes, QC-LDPC codes are not characterized by a single unidimensional structure, but rather by a bidimensional one (they are represented by a base matrix, rather than a polynomial).

Differently from [20], where time varying convolutional codes are treated, we employ the encoder in Fig. 9 to encode block codes. However, it is interesting to notice that PRC-LDPC codes share some structural properties with LDPC convolutional codes. For example, in both cases the parity-check matrix has a diagonal-like structure. As a spark for future works, we mention that a convolutional version of PRC-LDPC codes can be obtained by applying a different type of puncturing, called constant-length puncturing [25]. If such operation is applied, the resulting parity-check matrix has the same number of columns as the initial simplex code, but a smaller number of rows. Then, an infinite frame has to be constructed. However, the analysis of convolutional PRC-LDPC codes requires a self-standing approach and, for this reason, we do not delve into them in this paper.

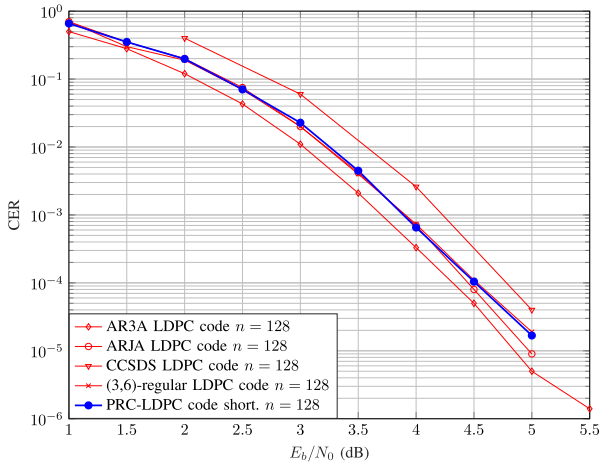

Fig. 10. Codeword Error Rate vs E_b/N_0, for various codes in comparison with PRC-LDPC codes under LLR-SPA decoding running at most 200 iterations.

TABLE I
Parameters of the Considered Codes With $k = 512$

TYPE	R	n	p	Number of transmitted bits	z
PRC	3/4	683	-	683	41
PRC	4/5	640	-	640	41
PRC	5/6	615	-	615	41
PRC	6/7	598	-	598	41
Raptor	3/4	731	48	683	16
Raptor	4/5	688	48	640	16
Raptor	5/6	663	48	615	16
Raptor	6/7	646	48	598	16

Regarding decoding complexity, the sparsity of the parity-check matrix of PRC-LDPC codes enables the use of low-complexity BP-based decoding algorithms, such as the Sum-Product Algorithm (SPA) [17], Min-Sum (MS) [26], Normalized Min-Sum (NMS) [27], and many others. Let us consider the implementation of the Log-Likelihood Ratio SPA (LLR-SPA) decoder proposed in [28] (which is also the one we consider for the Monte Carlo simulations reported in Section VI) and operations between 8-bit values. The number of binary operations required to decode a codeword, by running on average I_{avg} decoding iteration, is

$$\Lambda = nI_{\text{avg}}[8(8\langle w_c\rangle + 12R - 11) + \langle w_c\rangle]. \tag{8}$$

By substituting (1) into (8), we get that Λ is $O(nw_h)$.

TABLE II
SUPPORT VECTOR (PRC) AND BASE GRAPH (RAPTOR) OF THE CONSIDERED CODES

TYPE	R	$\mathrm{supp}(\mathbf{h})$ (PRC) / Base graph, Z (Raptor)
PRC	3/4	$[0, 3, 66, 97, 142, 220, 221, 295, 330, 354, 382, 402, 486, 546, 553]$
PRC	4/5	$[0, 3, 41, 95, 97, 152, 220, 221, 242, 295, 330, 338, 382, 415, 486, 504, 523, 546, 553]$
PRC	5/6	$[0, 3, 15, 41, 97, 106, 142, 152, 220, 242, 295, 338, 382, 388, 402, 415, 486, 504, 523, 546, 553]$
PRC	6/7	$[0, 3, 41, 66, 97, 106, 142, 152, 220, 225, 242, 295, 330, 338, 382, 388, 402, 415, 486, 504, 523, 546, 553]$
Raptor	3/4	1, 24
Raptor	4/5	1, 24
Raptor	6/7	1, 24
Raptor	7/8	1, 24

VI. PERFORMANCE ASSESSMENT

In this section we assess the performance of some codes in terms of Codeword Error Rate (CER) and Bit Error Rate (BER), through Monte Carlo simulations of Binary Phase Shift Keying (BPSK) transmissions over the Additive White Gaussian Noise (AWGN) channel.

A. Short Codes

Let us compare the performance, in terms of CER, of a PRC-LDPC code characterized by $\mathbf{e} = [0, 2, 21, 29, 60, 72, 75]$ with state-of-the-art short LDPC codes with $R = \frac{1}{2}$ and $n = 128$ [29], namely: an AR3A LDPC code [30], an ARJA LDPC code [31], the CCSDS telecommand LDPC code based on protographs [32] and a conventional $(3, 6)$-regular LDPC code. The PRC-LDPC code, for which $\mathbf{s} = [2, 19, 8, 31, 12, 3]$, verifies Remark 2. Moreover, $s_{\max}$ is such that Corollary 3 does not hold. For $R = \frac{1}{2}$, i.e., $n = 150$, the estimated minimum distance of this code (obtained by using the analysis developed in this paper and confirmed by the tool in [33]) is 11. In order to have the exact same length of the other codes, we have performed shortening of the PRC-LDPC code, by considering $z = 11$ and punctured 11 more symbols, in such a way that $k' = k - z = 64$, and $n' = n - z - 11 = 128$. The estimated minimum distance of the shortened code is 10. From the simulation results, reported in Fig. 10, we observe that its performance is comparable to that of the other codes. Clearly, there is a trade-off between the encoding/decoding complexity and the error rate performance, and the slight performance loss observed with respect to the AR3A and ARJA codes counterbalances the advantage of allowing extremely simple encoding, as discussed in Section V.

B. Rate-Compatible Codes

Let us compare the error correction performance of some high rate RC raptor-like codes, designed following the 5G standard [34], to that of PRC-LDPC codes. The parameters of all the considered codes are summarized in Tables I and II, where Z denotes the circulant size for 5G codes. The Golomb ruler employed to generate the parity-check polynomials of PRC-LDPC codes is

[0, 3, 15, 41, 66, 95, 97, 106, 142, 152, 220, 221, 225, 242, 295, 330, 338, 354, 382, 388, 402, 415, 486, 504, 523, 546, 553].

Remark 3: The reported number of punctured bits, denoted as p in the table, refers to the actual number of bits which are not transmitted (i.e., n is the number of transmitted bits plus the number of punctured bits p), but the corresponding variable nodes are present in the code Tanner graph. This is also the reason of the larger value of n for 5G codes, with respect to PRC-LDPC codes with the same code rate and $k = 512$. However, as shown in Table I, the number of bits transmitted over the channel is the same, for codes with the same rate.

In Table I we also report z, i.e., the number of symbols that must be shortened in order to achieve the desired values of k and R, for both PRC-LDPC codes and 5G codes. To be noticed that, for PRC-LDPC codes, the shortened symbols have been chosen randomly, thus some further performance gain could be achieved by optimizing the shortening pattern. However, this goes beyond the scope of the present paper.

Simulation results, reported in Fig.s 11 and 12, obtained by running the LLR-SPA decoder with at most 100 decoding iterations, show that, for the considered code rates, the performance of PRC-LDPC codes is very close to that of 5G codes, and even better, in terms of BER, for some relatively low values of $\frac{E_b}{N_0}$. In general, we notice that the loss of PRC-LDPC codes in terms of $\frac{E_b}{N_0}$ is always within 0.2 dB, with respect to 5G LDPC codes. If we consider that PRC-LDPC codes enable an extremely fast encoding process, and do not exploit punctured nodes for decoding, the result seems remarkable.

We have also computed the throughput of PRC-LDPC codes, defined as the number of bits per channel use needed to achieve reliable transmission and denoted as R^*_{PRC}, and compared it to that of the 5G LDPC codes, denoted as R^*_{5G}. As a reliability target, we consider $\mathrm{BER} \leq \epsilon$, with $\epsilon \in \{10^{-4}, 10^{-5}\}$ and, coherently with Fig. 11, we set $k = 512$. The throughput values, reported in Fig. 13, denote that the performance of PRC-LDPC codes and 5G codes is

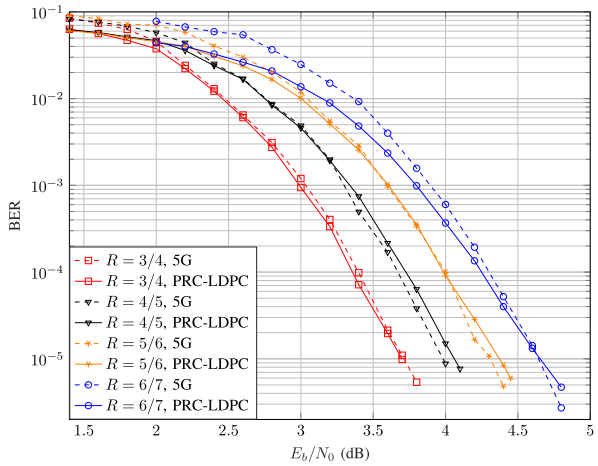

Fig. 11. Bit Error Rate vs E_b/N_0, for 5G codes in comparison with PRC-LDPC codes, with $k = 512$, under LLR-SPA decoding running at most 100 iterations.

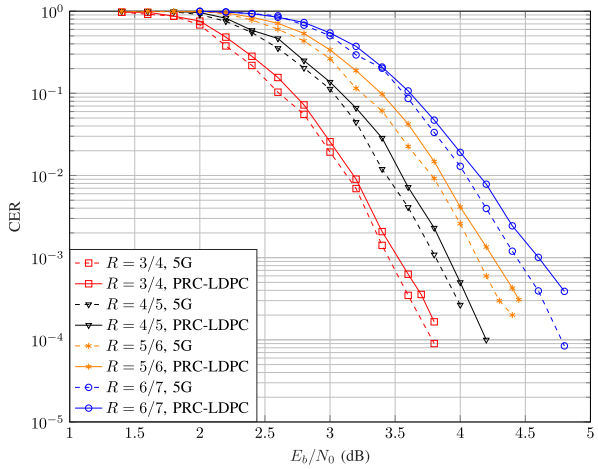

Fig. 12. Codeword Error Rate vs E_b/N_0, for 5G codes in comparison with PRC-LDPC codes, with $k = 512$, under LLR-SPA decoding running at most 100 iterations.

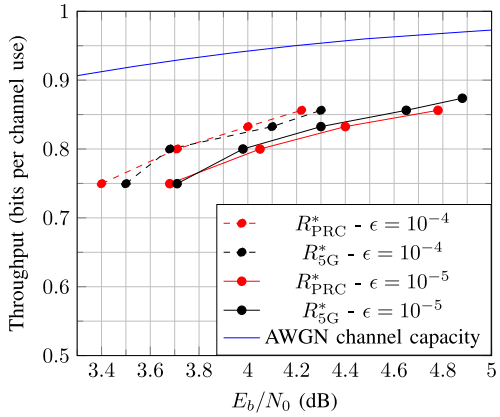

Fig. 13. Throughput of PRC-LDPC and 5G LDPC codes.

closely matched across the evaluated ϵ values, with neither code consistently outperforming the other. The AWGN channel capacity is also reported in Fig. 13, as an ultimate bound.

VII. Conclusion

A new family of LDPC codes, obtained by puncturing cyclic simplex codes and called PRC-LDPC codes, has been proposed. These codes exhibit several advantages, including simple encoders, the possibility to be decoded with low-complexity algorithms and inherent rate compatibility. As another advantage, the minimum distance properties of PRC-LDPC codes can be analyzed and estimated using theoretical methods. The proposed codes seem particularly suitable in applications where low latency and complexity are paramount. The inherent rate-compatibility of PRC-LDPC codes, along with their highly structured parity-check matrices, provide them with high adaptability. This adaptability should be particularly beneficial in dynamic scenarios where latency and complexity requirements might fluctuate.

Appendix
Low-Weight Codewords of the PRC-LDPC Codes From Example 2

Given the parity-check polynomial $h(x) = 1 + x + x^5 + x^{11} + x^{13}$, we have that $\mathbf{e} = [0, 1, 5, 11, 13]$ is a Golomb ruler. We study next the portion of $\mathbf{p}$ described by Lemma 6, showing how low-weight codewords arise for increasing values of n.

For this family of PRC-LDPC codes, we have $r(2) = 6$ and therefore $n(2) = 19$. By exhaustive search we find $A(2) = 3$, and these three codewords indeed belong to Families 1 (2 out of 3) and 2 (the remaining 1), as described in Lemma 6. They are marked in bold in the following portion of $\mathbf{p}$

$$\ldots 10001|\mathbf{1000000000010000000}|101001\ldots,$$
$$\ldots 100011|\mathbf{0000000001000000001}|01001\ldots,$$
$$\ldots 1000110|\mathbf{0000000001000000010}|1001\ldots$$

By increasing n to 20, we get $A(2) = 1$, still belonging to Family 2

$$\ldots 100011|\mathbf{0000000001000000010}|1001\ldots$$

According to Corollary 1, we have $n = 21 = n(3)$ and indeed $d = 3$ and $A(d) = 8$, where the following codewords are derived from Families 1 and 2 as described in Lemma 7

$$\ldots 10001|\mathbf{1000000000010000000010}|1001\ldots,$$
$$\ldots 100011|\mathbf{0000000001000000000101}|001\ldots$$

and the others belong to new families. All the 8 codewords of weight 3, except for one of them, can be found in the above portion of $\mathbf{p}$. Finally, when $n = 22$, the two aforementioned codewords merge in a single codeword of weight 4

$$\ldots 10001|\mathbf{1000000000010000000101}|001\ldots$$

which, however it is not a minimum distance codeword, since $A(3) = 4$ and, again, all the minimum codewords come from the shown portion of $\mathbf{p}$. The PRC-LDPC code instead assumes $d = 4$ when $n = 24$, for which $A(4) = 10$. For the sake of brevity, we only show the four codewords of weight 4 deriving from the straightforward application of the results in Section IV to the codewords listed above but, once more,

all the codewords of weight 4 can be found in the portion of $\mathbf{p}$ shown below

$$\begin{aligned}
&\ldots 1|\mathbf{000110000000000100000001}|01001\ldots,\\
&\ldots|\mathbf{100011000000000010000000}|101001\ldots,\\
&\ldots 100011|\mathbf{000000000010000000101001}|\ldots,\\
&\ldots 10001|\mathbf{100000000001000000010100}|1\ldots
\end{aligned}$$

REFERENCES

[1] M. Battaglioni and G. Cancellieri, "Punctured binary simplex codes as LDPC codes," in *Proc. 61st FITCE Int. Congr. Future Telecommun., Infrastruct. Sustainability (FITCE)*, Sep. 2022, pp. 1–6.

[2] M. Battaglioni, M. Baldi, F. Chiaraluce, and G. Cancellieri, "Rate-adaptive LDPC codes obtained from simplex codes," in *Proc. IEEE Int. Conf. Commun.*, May 2023, pp. 1–6.

[3] J. Ha, J. Kim, and S. W. McLaughlin, "Rate-compatible puncturing of low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 50, no. 11, pp. 2824–2836, Nov. 2004.

[4] G. I. Davida and S. M. Reddy, "Forward-error correction with decision feedback," *Inf. Control*, vol. 21, no. 2, pp. 117–133, Sep. 1972.

[5] *Flexibility Evaluation of Channel Coding Schemes for NR-Discussion and Decision*, Standard TSG RAN WG13, GPP, 2016.

[6] D. Hui, S. Sandberg, Y. Blankenship, M. Andersson, and L. Grosjean, "Channel coding in 5G new radio: A tutorial overview and performance comparison with 4G LTE," *IEEE Veh. Technol. Mag.*, vol. 13, no. 4, pp. 60–69, Dec. 2018.

[7] W. E. Ryan and S. Lin, *Channel Codes—Classical and Modern*. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[8] M. El-Khamy, J. Hou, and N. Bhushan, "Design of rate-compatible structured LDPC codes for hybrid ARQ applications," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 6, pp. 965–973, Aug. 2009.

[9] F. I. Ivanov and P. S. Rybin, "On the nested family of LDPC codes based on Golomb rulers," in *Proc. 4th Int. Conf. Eng. Telecommun. (EnT)*, Nov. 2017, pp. 67–71.

[10] X. Xiao, B. Vasic, S. Lin, J. Li, and K. Abdel-Ghaffar, "Quasi-cyclic LDPC codes with parity-check matrices of column weight two or more for correcting phased bursts of erasures," *IEEE Trans. Commun.*, vol. 69, no. 5, pp. 2812–2823, May 2021.

[11] I. Kim and H. Song, "A construction for girth-8 QC-LDPC codes using Golomb rulers," *Electron. Lett.*, vol. 58, no. 15, pp. 582–584, Jul. 2022.

[12] C. Chen, B. Bai, Z. Li, X. Yang, and L. Li, "Nonbinary cyclic LDPC codes derived from idempotents and modular Golomb rulers," *IEEE Trans. Commun.*, vol. 60, no. 3, pp. 661–668, Mar. 2012.

[13] S. Zhao and X. Ma, "Construction of high-performance array-based nonbinary LDPC codes with moderate rates," *IEEE Commun. Lett.*, vol. 20, no. 1, pp. 13–16, Jan. 2016.

[14] M. Shirvanimoghaddam, "Primitive rateless codes," *IEEE Trans. Commun.*, vol. 69, no. 10, pp. 6395–6408, Oct. 2021.

[15] R. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vols. IT-27, no. 5, pp. 533–547, Sep. 1981.

[16] M. Baldi, M. Bianchi, F. Chiaraluce, and T. Klove, "A class of punctured simplex codes which are proper for error detection," *IEEE Trans. Inf. Theory*, vol. 58, no. 6, pp. 3861–3880, Jun. 2012.

[17] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[18] S. Fredricsson, "Pseudo-randomness properties of binary shift register sequences (Corresp.)," *IEEE Trans. Inf. Theory*, vols. IT-21, no. 1, pp. 115–120, Jan. 1975.

[19] A. Dimitromanolakis. (2023). *Golomb Rulers and Sidon Sets*. Accessed: May 19, 2023. [Online]. Available: http://www.cs.toronto.edu/

[20] A. J. Felstrom and K. S. Zigangirov, "Time-varying periodic convolutional codes with low-density parity-check matrix," *IEEE Trans. Inf. Theory*, vol. 45, no. 6, pp. 2181–2191, Sep. 1999.

[21] T. T. B. Nguyen, T. Nguyen Tan, and H. Lee, "Efficient QC-LDPC encoder for 5G new radio," *Electronics*, vol. 8, no. 6, p. 668, Jun. 2019.

[22] T. J. Richardson and R. L. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 638–656, Feb. 2001.

[23] S. Lin and D. J. Costello, *Error Control Coding*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2004.

[24] Z. Li, L. Chen, L. Zeng, S. Lin, and W. H. Fong, "Efficient encoding of quasi-cyclic low-density parity-check codes," *IEEE Trans. Commun.*, vol. 54, no. 1, pp. 71–81, Jan. 2006.

[25] G. Cancellieri, *Polynomial Theory of Error Correcting Codes*. Cham, Switzerland: Springer, 2015.

[26] M. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *IEEE Trans. Commun.*, vol. 47, no. 5, pp. 673–680, May 1999.

[27] J. Chen and P. Fossorier, "Density evolution for BP-based decoding algorithms of LDPC codes and their quantized versions," in *Proc. IEEE Global Telecommun. Conf.*, vol. 2, 2002, pp. 1378–1382.

[28] X.-Y. Hu, E. Eleftheriou, D.-M. Arnold, and A. Dholakia, "Efficient implementations of the sum-product algorithm for decoding LDPC codes," in *Proc. IEEE Global Telecommun. Conf.*, Nov. 2001, p. 1036.

[29] G. Liva, L. Gaudio, T. Ninacs, and T. Jerkovits, "Code design for short blocks: A survey," 2016, *arxiv:1610.00873*.

[30] D. Divsalar, S. Dolinar, and J. Thorpe, "Accumulate-repeat-accumulate-accumulate-codes," in *Proc. IEEE 60th Veh. Technol. Conf., VTC-Fall.*, Aug. 2004, pp. 2292–2296.

[31] D. Divsalar, S. Dolinar, C. Jones, and K. Andrews, "Capacity-approaching protograph codes," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 6, pp. 876–888, Aug. 2009.

[32] D. Divsalar, S. Dolinar, and C. Jones, "Short protograph-based LDPC codes," in *Proc. IEEE Mil. Commun. Conf.*, Oct. 2007, pp. 1–6.

[33] D. J. C. MacKay. (2008). *Source Code for Approximating the MinDist Problem of LDPC Codes*. [Online]. Available: http://www.inference.eng.cam.ac.uk/mackay/MINDIST_ECC.html

[34] *Group Radio Access Network; NR; Multiplexing Channel Coding*, Standard TS 38.212, 3GPP, 2020.

Open Access funding provided by 'Università Politecnica delle Marche' within the CRUI CARE Agreement