

Decoding of NB-LDPC codes over Subfields

V. B. Wijekoon, Emanuele Viterbo, Yi Hong
Monash University, Australia

Abstract—The non-binary low-density parity-check (NB-LDPC) codes can offer promising performance advantages but suffer from high decoding complexity. To tackle this challenge, in this paper, we consider NB-LDPC codes over finite fields as codes over *subfields* as a means of reducing decoding complexity. In particular, our approach is based on a novel method of expanding a non-binary Tanner graph over a finite field into a graph over a subfield. This approach offers several decoding strategies for a single NB-LDPC code, with varying levels of performance-complexity trade-offs. Simulation results demonstrate that in a majority of cases, performance loss is minimal when compared with the complexity gains.

Index Terms—Non-binary LDPC codes, Graph expansion, Iterative decoding

I. INTRODUCTION

Low-density parity-check (LDPC) codes, introduced by Gallager in 1962 [1], have become the error-correcting codes of choice for many practical applications, such as Ethernet, Wi-Fi, and digital television, due to their capacity approaching performance and low-complexity decoding algorithms [2]. Davey and McKay introduced the non-binary (NB) counterparts of these codes in 1998 [3], and it was soon realized that NB-LDPC codes outperform the binary LDPC codes of comparable length, especially for short-to-moderate code lengths. But these performance gains are yet to be realized in practice due to the high complexity of decoding algorithms.

Best performing decoding algorithm for NB-LDPC codes is the Q -ary sum-product algorithm (QSPA), a generalization of the sum-product algorithm used with binary LDPC codes [3]. Complexity of QSPA is of the order $\mathcal{O}(q^2)$, where q is the cardinality of the algebraic structure over which the code is defined. This complexity is too high for most practical applications, also considering that QSPA requires large memory resources, since the messages used for decoding are probability vectors of length q . Fast Fourier transform based implementation of QSPA (FFT-QSPA) reduces decoding complexity to $\mathcal{O}(q \log q)$, but still requires similar levels of memory resources [4]. Log-domain implementations of QSPA (LLR-QSPA) have also been considered in the literature [5].

In [5], the authors introduced a simplified version of LLR-QSPA, referred to as ‘max-log-SPA’, by extending the simplification used in min-sum decoding to NB-LDPC codes and QSPA. This approach was further developed in [6] by introducing the ‘Extended Min-Sum’ (EMS) algorithm. Instead of considering the complete length q vectors at check node operations, EMS uses the n_m most significant values of each vector, resulting in a complexity order of $\mathcal{O}(n_m q)$. A different approach to simplifying operations of QSPA was proposed in [7], where the ‘Min-max’ algorithm was introduced with the

same complexity order as QSPA, but using only addition and comparison operations. Efficient hardware implementations were proposed for both EMS and min-max algorithms in [8] and [9].

Expanding the parity-check matrix (PCM) of a NB-LDPC code into a *binary* one allows devising low-complexity bit-level decoding strategies for the non-binary code. Such expansion was proposed in [10] and is referred to as the ‘extended binary representation’. Additionally, a decoding algorithm for NB-LDPC codes over the binary erasure channel was introduced in [10]. This strategy is adapted to general channels, as discussed in [11]. In [12], the authors used the *binary image* of the non-binary PCM to decode NB-LDPC codes.

While it is possible to construct NB-LDPC codes over many algebraic structures, they are often defined over finite fields, particularly those of characteristic 2, (i.e., \mathbb{F}_{2^r}) [3]. Given that a finite field \mathbb{F}_{p^r} contains a unique subfield \mathbb{F}_{p^m} , for every $m \mid r$ [13], in this paper, we consider expanding the PCM of a NB-LDPC code over \mathbb{F}_{p^r} to a matrix over any subfield \mathbb{F}_{p^m} . For codes over \mathbb{F}_{2^r} , this includes the expansion into \mathbb{F}_2 , a binary expansion as a special case. We then propose a general decoding algorithm, usable with any one of the possible expansions. Since the operations of the decoder would now be with a smaller size field, significant gains in complexity is achievable, and simulation results demonstrate that the performance loss in comparison to QSPA is minimal. Moreover, using our approach, it is possible to decode the same code over several different fields, each offering a range of performance-complexity trade-offs.

The remaining of the paper is organized as follows. Section II introduces the mathematical concepts used for the expansion, while Section III provides the expansion along with some examples. Section IV presents the decoding strategy, whereas Section V includes simulation results. Section VI analyzes the complexity and memory requirements of the new decoding scheme, and Section VII concludes the paper.

II. α -CONNECTED SUBGROUPS

Consider a finite field of characteristic p , \mathbb{F}_{p^r} , and one of its additive subgroups, G . We denote a primitive element of \mathbb{F}_{p^r} with α . It is easy to verify that multiplying all the elements in G with some $\alpha^i \in \mathbb{F}_{p^r}$ yields another additive subgroup. Then we have the following definition.

Definition 1. Additive subgroups G_1 and G_2 of \mathbb{F}_{p^r} , where one can be obtained from the other by multiplying by some power of α , are called α -connected subgroups.

If subgroups G_1 and G_2 are α -connected, and so are G_2 and G_3 , then clearly G_1 and G_3 are also α -connected. This yields the following definition.

Definition 2. A set of subgroups $S = \{G_1, \dots, G_n\}$ of \mathbb{F}_{p^r} is an α -connected set if

- 1) each $G_i \in S$ is α -connected with all other $G_j \in S$.
- 2) $G_i \in S$ is α -connected with some G , then $G \in S$.

An α -connected set S can be generated using any $G_i \in S$, simply by multiplying with increasing powers of α . Each generated subgroup will be added to the set until $\alpha^{i^*} \cdot G_i = G_i$ for some power i^* . This i^* gives the cardinality $|S|$ of the set S . Lemma 1 considers the minimum possible cardinality of an α -connected set.

Lemma 1. Consider \mathbb{F}_{p^r} and let m divide r (i.e., $m \mid r$). Then the smallest possible α -connected set of additive subgroups of order p^{r-m} has a cardinality of $\frac{p^r-1}{p^m-1}$.

Proof: Let G be an additive subgroup of \mathbb{F}_{p^r} of order $|G| = p^{r-m}$. The minimum possible cardinality of an α -connected set is the minimum value $i \in \{1, \dots, p^r - 2\}$ satisfying $\alpha^i \cdot G = G$, denoted by i_m .

Let $S_{\alpha^{i_m}}$ be the set of elements in \mathbb{F}_{p^r} generated by α^{i_m} . Note that i_m is the minimum non-zero power of α in $S_{\alpha^{i_m}}$. If $\alpha^{i_m} \cdot G = G$, then clearly $\alpha^{ki_m} \cdot G = G$, for any $\alpha^{ki_m} \in S_{\alpha^{i_m}}$. Since we are focused on the minimum, we only consider i_m , for which the following relation holds.

$$i_m |S_{\alpha^{i_m}}| = (p^r - 1) \quad (1)$$

Since we assume $\alpha^{i_m} \cdot G = G$, if $g \in G$, then $g \cdot S_{\alpha^{i_m}} \subset G$. As G is an additive subgroup, it must contain the additive identity 0, and $0 \cdot S_{\alpha^{i_m}} = \{0\}$. Note that for $g_1, g_2 \in G$ that are both $\neq 0$, the sets $g_1 \cdot S_{\alpha^{i_m}}$ and $g_2 \cdot S_{\alpha^{i_m}}$ would be of the same size, and they should either be the same set or disjoint. Then, as the order of G is p^{r-m} , disregarding 0, following should hold for some value n .

$$n |S_{\alpha^{i_m}}| = (p^{r-m} - 1) \quad (2)$$

From (1) and (2), we see that $|S_{\alpha^{i_m}}|$ is a factor of both $(p^r - 1)$ and $(p^{r-m} - 1)$. Since (1) shows that i_m and $|S_{\alpha^{i_m}}|$ are inversely proportional, $|S_{\alpha^{i_m}}|$ should be the *greatest common divisor* of $(p^r - 1)$ and $(p^{r-m} - 1)$. The following relations

$$\begin{aligned} (p^r - 1) &= (p^m - 1) \sum_{i=0}^{\frac{r}{m}-1} (p^m)^i \\ (p^{r-m} - 1) &= (p^m - 1) \sum_{i=0}^{\frac{r}{m}-2} (p^m)^i \\ \sum_{i=0}^{\frac{r}{m}-1} (p^m)^i &= p^m \sum_{i=0}^{\frac{r}{m}-2} (p^m)^i + 1 \end{aligned}$$

allow us to conclude that $\gcd(p^r - 1, p^{r-m} - 1) = (p^m - 1)$, and using (1);

$$i_m = \frac{p^r - 1}{p^m - 1}$$

■

Lemma 1 shows that the smallest α -connected set has cardinality $\frac{p^r-1}{p^m-1}$ but it does not reveal how to construct such a set. It should also be noted that the additive property of the

group G is not used for the proof: only the existence of the additive identity is used.

Lemma 2 outlines a method to explicitly construct an α -connected set of subgroups of order p^{r-m} .

Lemma 2. Let G be a subgroup of order p^m in $H' = \{\mathbb{F}_{p^r}, +\}$, and ψ some surjective homomorphism $\psi : H' \rightarrow G$. The kernels of the set of homomorphisms $\psi_i(h') = \psi(\alpha^{-i}h')$, for $i = \{0, 1, \dots, p^r - 2\}$, form an α -connected set of additive subgroups of order p^{r-m} .

Proof: For $\psi : H' \rightarrow G$, $\ker(\psi)$ is an additive subgroup of order p^{r-m} in \mathbb{F}_{p^r} [13]. Since $\psi_i(h') = \psi(\alpha^{-i}h')$, it is clear that $\ker(\psi_i) = \alpha^i \ker(\psi)$. Thus, $\ker(\psi)$ and $\ker(\psi_i)$ are α -connected subgroups for all possible i . Also, for any i_1, i_2 , $\ker(\psi_{i_1}) = \alpha^{i_1-i_2} \ker(\psi_{i_2})$. Then, the set of kernels $S = \{\ker(\psi_0), \ker(\psi_1), \dots, \ker(\psi_{p^r-2})\}$, where possible duplicates have been removed, satisfy the conditions of Definition 2, and thus form a α -connected set. ■

The cardinality of an α -connected set generated as in Lemma 2 depends on the homomorphism ψ . Therefore, to construct the smallest α -connected set, one must find a suitable homomorphism. The homomorphism we use is based on the representation of \mathbb{F}_{p^r} as an *extension* of the subfield \mathbb{F}_{p^m} . The following Lemma establishes the structure of \mathbb{F}_{p^m} in \mathbb{F}_{p^r} .

Lemma 3. Let S_β be the set of elements in \mathbb{F}_{p^r} generated by $\beta = \alpha^{\frac{p^r-1}{p^m-1}}$. Then, $S_\beta \cup \{0\}$, where 0 is the additive identity of \mathbb{F}_{p^r} , is the subfield \mathbb{F}_{p^m} .

Proof: Since $m \mid r$, \mathbb{F}_{p^r} contains the subfield \mathbb{F}_{p^m} . Let us consider the multiplicative groups $K' = \{\mathbb{F}_{p^r}, \times\}$ and $K = \{\mathbb{F}_{p^m}, \times\}$. Then K is a subgroup of order (p^m-1) of K' . Note that both K and K' are cyclic. From properties of subgroups of cyclic groups [13], there should be only one unique subgroup of a specific order in K' . The set of elements generated by β , S_β , is such a subgroup, of order $(p^m - 1)$, and thus $K = S_\beta$. This allows to conclude that $\mathbb{F}_{p^m} = S_\beta \cup \{0\}$. ■

We are interested in the polynomial representation of \mathbb{F}_{p^r} as an extension of \mathbb{F}_{p^m} . In such a representation, some $\alpha^i \in \mathbb{F}_{p^r}$ is represented with a polynomial $E_{\alpha^i}(x)$ over \mathbb{F}_{p^m} , of degree at most $(\frac{r}{m} - 1)$. In the case of elements belonging to \mathbb{F}_{p^m} (for β^i), the polynomials would be of degree 0. The primitive polynomial $\Pi(x)$ for the representation is of degree $\frac{r}{m}$ and, since $\Pi(x)$ is irreducible, it must have a non-zero constant term. Based on this representation, we define a homomorphism ψ^* between the additive groups of \mathbb{F}_{p^r} and \mathbb{F}_{p^m} as follows.

Definition 3. Let $H' = \{\mathbb{F}_{p^r}, +\}$, and $H = \{\mathbb{F}_{p^m}, +\}$. The homomorphism $\psi^* : H' \rightarrow H$ is mapping $h' \in H'$ to $h \in H$ where the constant term in $E_{h'}(x)$ is h .

Using homomorphism ψ^* in the method proposed in Lemma 2 generates an α -connected set of minimum cardinality.

Lemma 4. The set of kernels of homomorphisms $\psi_i^*(h') = \psi^*(\alpha^{-i}h')$, for $i = \{0, 1, \dots, p^r-2\}$ form an α -connected set of additive subgroups of order p^{r-m} of the minimum cardinality $\frac{p^r-1}{p^m-1}$.

Proof: Let $\psi^* : H' \rightarrow H$, where $H' = \{\mathbb{F}_{p^r}, +\}$ and $H = \{\mathbb{F}_{p^m}, +\}$. Since $|H| = p^m$, $\ker(\psi^*)$ is a subgroup of order p^{r-m} in H' . Using ψ_i^* in Lemma 2, it is clear that kernels of $\psi_i^*(h') = \psi^*(\alpha^{-i}h')$ form an α -connected set. The cardinality of this set is equal to the minimum value of i for which $\ker(\psi_i^*) = \alpha^i \ker(\psi_0^*) = \ker(\psi_0^*)$, which we denote with i_m .

Let $g_j \in \ker(\psi_0^*)$ and $\alpha^{i_m} g_j = \gamma_j$, $j = 1, \dots, p^{r-m}$. Let polynomial representations (in the extended form) of $\alpha^{i_m} g_j$ and γ_j be $E_{\alpha^{i_m} g_j}(x)$, $E_{g_j}(x)$ and $E_{\gamma_j}(x)$, respectively. These are related by the mod $\Pi(x)$ polynomial multiplication

$$E_{\alpha^{i_m} g_j}(x) E_{g_j}(x) = \Pi(x) K_j(x) + E_{\gamma_j}(x)$$

where $K_j(x)$ is some polynomial over \mathbb{F}_{p^m} . Since $g_j \in \ker(\psi_0^*)$, the constant term in $E_{g_j}(x)$ is zero, which makes the constant term in $E_{\alpha^{i_m} g_j}(x) E_{g_j}(x)$ also zero. Note that for $\gamma_j \in \ker(\psi_0^*)$, constant term of $E_{\gamma_j}(x)$ should be zero. As observed earlier, $\Pi(x)$ has a non-zero constant term, and therefore, for $\gamma_j \in \ker(\psi)$, $K_j(x)$ should be a polynomial with a zero constant term. For $\alpha^{i_m} \ker(\psi_0^*) = \ker(\psi_0^*)$, this should be true for $j = 1, \dots, p^{r-m}$.

Polynomial representations of elements in $\ker(\psi_0^*)$ contains at least one polynomial of each possible degree, from 0 to $\frac{r}{m} - 1$. Then, if $\deg(E_{\alpha^{i_m} g_j}(x)) > 0$, for at least one value of j , $E_{\alpha^{i_m} g_j}(x) E_{g_j}(x)$ would be of degree $\frac{r}{m}$. Since $\Pi(x)$ is also of degree $\frac{r}{m}$, this requires $K_j(x)$ to be a non-zero constant for that particular value of j , resulting in $\alpha^{i_m} \ker(\psi_0^*) \neq \ker(\psi_0^*)$. Therefore, for $\gamma_j \in \ker(\psi_0^*)$ for all $j = 1, \dots, p^{r-m}$, $\deg(E_{\alpha^{i_m} g_j}(x)) = 0$. In such a case, $K_j(x) = 0$ for all j . This requires $\alpha^{i_m} \in \mathbb{F}_{p^m}$, and since we require the minimum, $i_m = \frac{p^r - 1}{p^m - 1}$.

Thus, using the homomorphism ψ^* in Lemma 2, it is possible to construct an α -connected set of additive subgroups of order p^{r-m} , that has the minimum cardinality $\frac{p^r - 1}{p^m - 1}$, whose existence was proved in Lemma 1. ■

III. GRAPH EXPANSION

In this section, we present how a graph over \mathbb{F}_{p^r} can be expanded into a larger one over \mathbb{F}_{p^m} , where $m | r$, using the smallest set of α -connected subgroups of order p^{r-m} in \mathbb{F}_{p^r} , constructed as detailed in the previous section. We represent this special α -connected set by Θ_m from here onwards. Basic mathematical concepts used in the expansion are first briefly over-viewed in Subsection A, while the expansion is presented in Subsection B, along with an example.

A. Preliminaries

Consider some surjective homomorphism $\psi : H' \rightarrow H$, where $H' = \{\mathbb{F}_{p^r}, +\}$ and $H = \{\mathbb{F}_{p^m}, +\}$. As remarked earlier as well, $\ker(\psi)$ is a subgroup of H' , of order p^{r-m} . Since the homomorphism is surjective, according to the first isomorphism theorem [13], the quotient group $Q_\psi = H' / \ker(\psi)$ is isomorphic to H , i.e., Q_ψ contains the p^m cosets of $\ker(\psi)$, including the trivial coset ($\ker(\psi)$ itself). In the isomorphism between Q_ψ and H , this trivial coset maps to the identity

element of H (the additive identity of \mathbb{F}_{p^m}), and the other cosets map to the remaining elements of H .

Let $Q_\psi = \{C_\psi^0, C_\psi^1, \dots, C_\psi^{p^m-1}\}$, where each C_ψ^j represents some coset of $\ker(\psi)$, with C_ψ^0 representing the trivial coset. Cosets contain elements in \mathbb{F}_{p^r} , and using the multiplicative properties of the field, we define a ‘multiplication’ operation on Q_ψ as follows.

Definition 4. Operation βQ_ψ , for some $\beta \in \mathbb{F}_{p^r}$, is defined as $\beta Q_\psi = \{\beta C_\psi^0, \beta C_\psi^1, \dots, \beta C_\psi^{p^m-1}\}$.

If two subgroups in H' are α -connected, then the respective quotient groups are also related in a similar way, as shown in the following lemma.

Lemma 5. Let G_{ψ_1} and G_{ψ_2} be two α -connected subgroups of H' , with $G_{\psi_1} = \alpha^k G_{\psi_2}$. Let $Q_{\psi_1} = H' / G_{\psi_1}$ and $Q_{\psi_2} = H' / G_{\psi_2}$ be the corresponding quotient groups. Then, $\alpha^k Q_{\psi_2}$ is the same set as Q_{ψ_1} .

Proof: Let $Q_{\psi_1} = \{C_{\psi_1}^0, \dots, C_{\psi_1}^{p^m-1}\}$ and $Q_{\psi_2} = \{C_{\psi_2}^0, \dots, C_{\psi_2}^{p^m-1}\}$. Here the trivial cosets $C_{\psi_j}^0$ are the subgroups themselves, and all the cosets can be represented using the respective subgroup and some coset leader term as follows:

$$Q_{\psi_1} = \{G_{\psi_1}, G_{\psi_1} + l_{\psi_1}^1, \dots, G_{\psi_1} + l_{\psi_1}^{p^m-1}\}$$

$$Q_{\psi_2} = \{G_{\psi_2}, G_{\psi_2} + l_{\psi_2}^1, \dots, G_{\psi_2} + l_{\psi_2}^{p^m-1}\}$$

Using the multiplication operation on Q_{ψ_2} yields

$$\alpha^k Q_{\psi_2} = \{\alpha^k G_{\psi_2}, \alpha^k G_{\psi_2} + \alpha^k l_{\psi_2}^1, \dots, \alpha^k G_{\psi_2} + \alpha^k l_{\psi_2}^{p^m-1}\}$$

As cosets of any subgroup are mutually exclusive, and due to the multiplicative properties of the field, any $\alpha^k G_{\psi_2} + \alpha^k l_{\psi_2}^j$ is disjoint with any other. Since $G_{\psi_1} = \alpha^k G_{\psi_2}$ then

$$\alpha^k Q_{\psi_2} = \{G_{\psi_1}, G_{\psi_1} + \alpha^k l_{\psi_2}^1, \dots, G_{\psi_1} + \alpha^k l_{\psi_2}^{p^m-1}\}$$

$\alpha^k Q_{\psi_2}$ is a set containing G_{ψ_1} , and its $(p^m - 1)$ proper cosets, albeit the coset leader terms could have changed. Thus, Q_{ψ_1} and $\alpha^k Q_{\psi_2}$ are the *same sets*. When elements of the quotient groups are considered in some specific order, then $\alpha^k Q_{\psi_2}$ will be some permutation of Q_{ψ_1} . ■

The homomorphisms ψ_i^* we use in constructing the smallest α -connected set, Θ_m , are all surjective. Θ_m consists of the kernels of these homomorphisms, and it is possible to construct a *set of quotient groups* with those kernels. Let that set be $\Theta_m^Q = \{Q_{\psi_i^*} ; i = 0, \dots, \frac{p^r-1}{p^m-1} - 1\}$. Since all ψ_i^* 's are surjective, each $Q_{\psi_i^*}$ is isomorphic to H , the additive group of \mathbb{F}_{p^r} . Since Θ_m is α -connected, according to Lemma 5, multiplying some $Q_{\psi_i^*} \in \Theta_m^Q$ by some power of α results in a permutation of some other $Q_{\psi_j^*} \in \Theta_m^Q$.

These observations about Θ_m^Q provide some insights on how to decode a code over \mathbb{F}_{p^r} over one of its subfields \mathbb{F}_{p^m} . Instead of traditionally used symbol probabilities, we consider the probabilities of a variable node *belonging* to each coset of each quotient group in Θ_m^Q . Then, for each variable node, $\frac{p^r-1}{p^m-1}$ probability vectors of length p^m are required, which we refer to as ‘*coset probability vectors (CPVs)*’. Complexity

bottleneck in decoding NB-LDPC codes are the check node operations [6]-[7], and advantages of our approach become apparent when the impact on that step is assessed.

Check node operations in decoding NB-LDPC codes consist of two major sub-steps: permutation and convolution of probability vectors [4]. In the permutation sub-step, the simpler one of the two, symbol probability vectors received by the check node are permuted, where the permutations are defined by the respective edge weights. Since Θ_m^Q is constructed using the smallest α -connected set Θ_m , it is clear from Lemma 5 that CPVs will also have to be permuted similarly. Thus, complexity of the permutation step will not be significantly affected in the proposed approach.

In order to understand how our approach changes the convolution sub-step, consider the simple case of a degree 3 check node in a code over \mathbb{F}_{p^r} , where the parity-check equation is $v_1 + v_2 + v_3 = 0$. A convolution has to be carried out using the incoming symbol probability vectors of v_1 and v_2 for computing the outgoing symbol probability vector of v_3 , $\mathbf{p}_{v_3}^s$. Since these vectors are of length r , convolution will be of complexity order $\mathcal{O}(p^{2r})$. Now assume we have to compute some i 'th CPV of v_3 , $\mathbf{p}_{v_3,i}^c$. This computation also only requires the incoming i 'th CPVs of the remaining two variable nodes. Note that these are of length m , where $m \mid r$. As all quotient groups in Θ_m^Q are isomorphic to the additive group of \mathbb{F}_{p^m} , computation of $\mathbf{p}_{v_3,i}^c$ should be the same as the convolution sub-step at a check node of a code over \mathbb{F}_{p^m} . Thus, complexity is now only of order $\mathcal{O}(p^{2m})$. However, with $|\Theta_m^Q| = \frac{p^r-1}{p^m-1}$, that many CPVs will have to be computed, resulting in an overall complexity of $p^{2m} \times \frac{p^r-1}{p^m-1} \approx \mathcal{O}(p^{m+r})$. Nevertheless, particularly for the cases where $m \ll r$, this is a significant reduction of complexity.

Motivated by the observation that using coset probability vectors instead of symbol probability vectors can allow faster decoding of NB-LDPC codes, we will provide a more detailed analysis of these complexity advantages in Section VI. In the following subsection, we explain how to *expand* a graph over \mathbb{F}_{p^r} into one over \mathbb{F}_{p^m} so that CPVs can be used in decoding.

B. Graph Expansion

We assume that a Tanner graph of a code over \mathbb{F}_{p^r} is to be expanded into a graph over \mathbb{F}_{p^m} , where $m \mid r$. The set of quotient groups, Θ_m^Q , will be of cardinality $\frac{p^r-1}{p^m-1}$. Each $Q_i \in \Theta_m^Q$ is isomorphic to $\{\mathbb{F}_{p^m}, +\}$, and in decoding, an associated CPV has to be used. Observations on how CPVs impact decoding suggest that it is possible to simply replace each node in the original graph, i.e., the so-called \mathbb{F}_{p^r} nodes, with $\frac{p^r-1}{p^m-1}$ \mathbb{F}_{p^m} nodes. Each variable node over \mathbb{F}_{p^m} would represent some CPV, and check nodes would calculate their estimates. How the set of \mathbb{F}_{p^m} variable and check nodes of a single neighboring variable-check node pair of the original graph are connected will depend on the original edge weight, as evident from Lemma 5.

Consider a check node and a variable node in the original graph, connected with an edge of weight $\alpha^k \in \mathbb{F}_{p^r}$. According to Lemma 5, $Q_i \in \Theta_m^Q$ becomes a permutation of some $Q_j \in$

Θ_m^Q when multiplied with α^k . Then, in the expansion, the \mathbb{F}_{p^m} variable node representing the i 'th CPV should be connected to the \mathbb{F}_{p^m} check node calculating estimates of the j 'th CPV. As Q_i turns into a *permutation* of Q_j , CPVs transmitted along this edge is permuted as well. Thus, this is a 2-step process, where first the set of CPVs are permuted, and then each CPV is permuted within itself. From the point-of-view of expansion, it is equivalent to connecting the set of \mathbb{F}_{p^m} variable nodes with the set of check nodes using edges labeled with elements from \mathbb{F}_{p^m} .

As an example, consider parity-check equation ρ from a code over \mathbb{F}_{2^4} , where α denotes a primitive of the field.

$$\rho \Rightarrow \alpha^4 v_1 + \alpha v_2 = 0 \quad (3)$$

Fig. 1 presents the initial expansion for ρ . The shaded graph is the original Tanner graph over \mathbb{F}_{2^4} , and the graph beneath is the expansion over \mathbb{F}_{2^2} . In both graphs, circles denote variable nodes and squares denote check nodes. Note that ω is a primitive of \mathbb{F}_{2^2} and that edges in the expanded graph are labeled with \mathbb{F}_{2^2} elements.

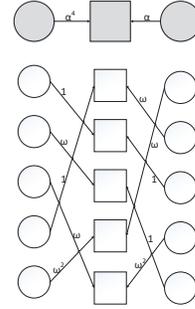


Fig. 1: Initial Expansion

Since each $Q_i \in \Theta_m^Q$ contains different groupings of the same set of symbols, the associated CPV contains some information about all other CPVs. Unfortunately, initial graph expansion is unable to capture these dependencies. In order to clearly visualize the relationships between CPVs, we propose an *alternate representation* of \mathbb{F}_{p^r} symbols below.

As each $Q_i \in \Theta_m^Q$ is isomorphic to $H = \{\mathbb{F}_{p^m}, +\}$, every coset in Q_i maps to some element of H . We define the *value* of some $\gamma \in \mathbb{F}_{p^r}$ with respect to some $Q_i \in \Theta_m^Q$ as the element of H that maps to the coset containing γ . Since $|\Theta_m^Q| = \frac{p^r-1}{p^m-1}$, using values defined with respect to each Q_i , γ can be uniquely represented as a vector of $\frac{p^r-1}{p^m-1}$ elements of H . For example, consider the case of \mathbb{F}_{2^4} and \mathbb{F}_{2^2} . Table I presents the $\frac{2^4-1}{2^2-1} = 5$ quotient groups in $\Theta_{2^2}^Q$, where each coset is listed under the $H_{2^2} = \{\mathbb{F}_{2^2}, +\}$ element we map to in the isomorphism between its quotient group and H_{2^2} . Alternative representations of \mathbb{F}_{2^4} elements as length 5 vectors over \mathbb{F}_{2^2} are listed in Table II. Note that a position i in these vectors map to quotient group Q_i as given in Table I.

Note that the sixteen vectors in Table II form a 2-dimensional space over \mathbb{F}_{2^2} . In channel coding terms, they are the 16 codewords of a $(2, 5)$ linear code over \mathbb{F}_{2^2} . Thus, values of some $\gamma \in \mathbb{F}_{2^4}$ with respect to 2 Q_i 's in $\Theta_{2^2}^Q$ are sufficient

	0	1	ω	ω^2
Q_0	$0, \alpha, \alpha^6, \alpha^{11}$	$1, \alpha^4, \alpha^{12}, \alpha^{13}$	$\alpha^2, \alpha^3, \alpha^5, \alpha^9$	$\alpha^7, \alpha^8, \alpha^{10}, \alpha^{14}$
Q_1	$0, 1, \alpha^5, \alpha^{10}$	$\alpha, \alpha^2, \alpha^4, \alpha^8$	$\alpha^6, \alpha^7, \alpha^9, \alpha^{13}$	$\alpha^3, \alpha^{11}, \alpha^{12}, \alpha^{14}$
Q_2	$0, \alpha^4, \alpha^9, \alpha^{14}$	$1, \alpha, \alpha^3, \alpha^7$	$\alpha^5, \alpha^6, \alpha^8, \alpha^{12}$	$\alpha^2, \alpha^{10}, \alpha^{11}, \alpha^{13}$
Q_3	$0, \alpha^2, \alpha^7, \alpha^{12}$	$1, \alpha^8, \alpha^9, \alpha^{11}$	$\alpha, \alpha^5, \alpha^{13}, \alpha^{14}$	$\alpha^3, \alpha^4, \alpha^6, \alpha^{10}$
Q_4	$0, \alpha^3, \alpha^8, \alpha^{13}$	$1, \alpha^2, \alpha^6, \alpha^{14}$	$\alpha^4, \alpha^5, \alpha^7, \alpha^{11}$	$\alpha, \alpha^9, \alpha^{10}, \alpha^{12}$

TABLE I: Quotient Groups in Θ_{24-2}^Q

0 : 00000	$\alpha^3 : \omega\omega^21\omega^20$	$\alpha^7 : \omega^2\omega10\omega$	$\alpha^{11} : 0\omega^2\omega^21\omega$
1 : 10111	$\alpha^4 : 110\omega^2\omega$	$\alpha^8 : \omega^21\omega10$	$\alpha^{12} : 1\omega^2\omega0\omega^2$
$\alpha : 011\omega\omega^2$	$\alpha^5 : \omega0\omega\omega\omega$	$\alpha^9 : \omega\omega01\omega^2$	$\alpha^{13} : 1\omega\omega^2\omega0$
$\alpha^2 : \omega1\omega^201$	$\alpha^6 : 0\omega\omega\omega^21$	$\alpha^{10} : \omega^20\omega^2\omega^2\omega^2$	$\alpha^{14} : \omega^2\omega^20\omega1$

TABLE II: Alternate Representations of Symbols in \mathbb{F}_{2^4}

to derive the remaining three. The dependencies could easily be captured through the parity-check equations of the code.

In the general case of \mathbb{F}_{p^r} and \mathbb{F}_{p^m} , alternate representation vectors would form a $\frac{r}{m}$ dimensional vector space, or in other words a $(\frac{p^r-1}{p^m-1}, \frac{r}{m})$ code, over \mathbb{F}_{p^m} . The $\frac{p^r-1}{p^m-1}$ \mathbb{F}_{p^m} nodes of every \mathbb{F}_{p^r} variable node would form this code, and since each such instance only involves the set of \mathbb{F}_{p^m} nodes of a single \mathbb{F}_{p^r} variable node, we refer to it as the ‘local code’. We propose using the parity-check matrix (PCM) of the local code, ${}^p\mathbb{H}_L^{r,m}$, to succinctly represent the dependencies between CPVs. Note that codes with parameters of the form $(\frac{p^r-1}{p^m-1}, \frac{r}{m})$ are from the family of non-binary simplex codes. Since dual of such a code is the $(\frac{p^r-1}{p^m-1}, \frac{p^r-1}{p^m-1} - \frac{r}{m})$ Hamming code over \mathbb{F}_{p^m} [14], parity-check equations would be Hamming codewords, and the PCM would contain $\frac{p^r-1}{p^m-1} - \frac{r}{m}$ of those. As an example, the ‘local’ PCM for the case of \mathbb{F}_{2^4} and \mathbb{F}_{2^2} , ${}^2\mathbb{H}_L^{4,2}$, which consists of 3 codewords of the (3, 5) Hamming code over \mathbb{F}_{2^2} , is given below.

$${}^2\mathbb{H}_L^{4,2} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & \omega & 0 & 1 & 0 \\ 1 & \omega^2 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

From the perspective of the expansion, parity-check equations due to the local code are a set of additional check nodes, which have to be added to the expanded graph. Since the set of \mathbb{F}_{p^m} nodes of every \mathbb{F}_{p^r} variable node forms one instance of the local code, $\frac{p^r-1}{p^m-1} - \frac{r}{m}$ additional check nodes representing the local PCM have to be added *per variable node of the original graph*. Adding such a large number of check nodes might seem to increase complexity, but it should be noted that these new nodes are of low degrees. Since the dual is a Hamming code, it should always be possible to find a set of degree 3 parity-check equations for the local PCM. With each new check node only being connected with a subset of the $\frac{p^r-1}{p^m-1}$ nodes of one \mathbb{F}_{p^r} variable node, they will be referred to as ‘local check nodes’ here onward. Check nodes resulting from expanding \mathbb{F}_{p^r} nodes will be called ‘regular check nodes’.

Performance with any form of iterative message passing decoding is dependant on features of the graph used, and

it is well-known that short cycles in the graph negatively impact decoding. While with binary codes, where the edge labels are all 1, effects of cycles depend only on their length, edge labels themselves have an impact in the non-binary case [15]. Particularly troublesome there are the cycles created by sub-matrices in the PCM that are not of full-rank [15]-[16]. Decoding performance of the expanded graph may be improved if the subgraph induced by the local PCM is free of these undesirable structures as much as possible. Although one could use a canonical generator matrix of a Hamming code as the local PCM, the graph induced may not entirely suit iterative decoding. In such a scenario, row operations can be carried out on the matrix until a ‘better’ one is obtained. This is particularly important when the expansion results in a binary graph ($p = 2$ and $m = 1$), since then *any* short cycle is detrimental for decoding. We have explored this case separately in [17], where the expansion was derived in an ad-hoc way, different from the more general approach presented here. In the non-binary case ($m > 1$), it might not be possible to remove all short cycles, and one might have to settle with a local PCM only free of short cycles not satisfying the ‘full-rank condition’, such as ${}^2\mathbb{H}_L^{4,2}$ given in (4). The decoding scheme we propose in the next section employs a technique to further reduce the possible negative effects of cycles among local check nodes.

Local check nodes enable us to adequately capture the various dependencies between CPVs, and adding those to the expanded graph wraps up the expansion. Different steps necessary for expanding a graph over \mathbb{F}_{p^r} into one over \mathbb{F}_{p^m} , where $m \mid r$, can be summarized as follows.

- 1) Obtain the smallest set of α -connected subgroups Θ_m , using the homomorphism in Definition 3, and following the steps outlined in Lemma 4. Use that to derive the set of quotient groups Θ_m^Q .
- 2) Map cosets of each $Q_i \in \Theta_m^Q$ with elements of $H = \{\mathbb{F}_{p^m}, +\}$. Use these isomorphisms to obtain the alternate representation vectors of \mathbb{F}_{p^r} elements.
- 3) Find a PCM more suited to iterative decoding for the code formed by alternate representation vectors.
- 4) Expand each node in the original graph into $\frac{p^r-1}{p^m-1}$ \mathbb{F}_{p^m} nodes. Connect the new variable and check nodes and label the edges, based on edge labels in the original graph.
- 5) Add local check nodes to represent the local PCM found in step 3.

Fig. 2 presents the complete expansion for the earlier example of parity-check equation ρ , given by (3). The \mathbb{F}_{2^4} graph is shaded grey, and the expansion to \mathbb{F}_{2^2} is depicted in white. Circles represent variable nodes, squares regular check nodes, and hexagons local check nodes. Note that in the interest of a clearer figure, edge labels are only shown for the instances where they are $\neq 1$.

IV. DECODING SCHEME

An iterative message passing decoding algorithm that utilizes the Tanner graph representation of NB-LDPC codes can be used with the expanded graph. Advantage herein is the

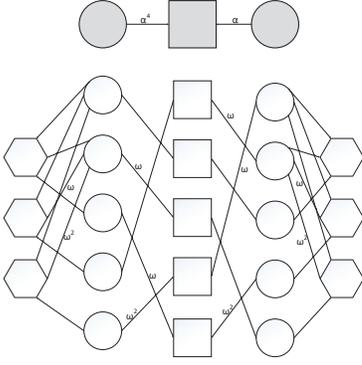


Fig. 2: Final Expansion

expansion being over a smaller field than the original graph, leading to a lower decoding complexity. Note that a few different options are available for expanding a graph over \mathbb{F}_{p^r} , one for each factor of r . Each of these would offer a different complexity-performance trade-off, which may suit different applications.

Any generic decoding algorithm can be applied straightforwardly to decode the expanded graph with some simple modifications. In the following, we present these modifications, and explain why they are required. Note that the explanation is from the perspective of a soft decision decoding (SDD) algorithm, such as QSPA [3], and its many variations [4]-[8], but can be also applied to other algorithms such as majority-logic decoding [18].

1) *Computing Channel Estimates*: Any SDD algorithm has to be initialized with probability estimates based on channel observations. In QSPA and its variants, for initializing the decoder, variable nodes compute channel estimates that are of the form of symbol probability vectors. In the proposed expansion, each variable node represents some CPV. Thus, when using SDD algorithms on expanded graphs, it is required to compute initial estimates for CPVs.

Note that each coset contains a subset of elements in \mathbb{F}_{p^r} . This makes computing initial estimates of CPVs quite straightforward, i.e., probability of a (\mathbb{F}_{p^r}) variable node belonging to a particular coset of some subgroup can be calculated by simply summing up probabilities of those symbols that belong to the coset. Equation (5) presents this computation, where \mathbf{p}_n^s is the symbol probability vector of original variable node n , $\mathbf{p}_{n,i}^c$ is the i 'th CPV of that node, C_i^j is the j 'th coset in i 'th quotient group, and a_k^j 's are \mathbb{F}_{p^r} elements in that coset.

$$\mathbf{p}_{n,i}^c(j) = \sum_{a_k^j \in C_i^j} \mathbf{p}_n^s(a_k^j) \quad j = 0, 1, \dots, (p^m - 1) \quad (5)$$

In most practical applications, decoders operate on either log or log-likelihood ratio (LLR) domain, due to hardware stability concerns [19]. In such a case, $\mathbf{p}_{n,i}^c$ has to be converted to the

desired domain, for example

$$\underline{L}_{n,i}^c(j) = \log \frac{\mathbf{p}_{n,i}^c(j)}{\mathbf{p}_{n,i}^c(0)} \quad j = 0, 1, \dots, (p^m - 1) \quad (6)$$

For one \mathbb{F}_{p^r} variable node, $\frac{p^r-1}{p^m-1} \mathbb{F}_{p^m}$ nodes that represent CPVs have to be initialized as in (5). Only a single symbol probability vector, corresponding to the single \mathbb{F}_{p^r} element transmitted through the channel, will be used for all those computations. This implies that channel observation is only sufficient to initialize $\frac{r}{m} \mathbb{F}_{p^m}$ symbols. However, in this approach, there are $\frac{p^r-1}{p^m-1}$ nodes that are initialized. Thus, channel observations are duplicated and dependencies are created between initial estimates of CPVs. Any error in channel estimates gets multiplied, and propagates through the graph, leading to performance losses.

Recall the fact that the set of \mathbb{F}_{p^m} nodes of a single \mathbb{F}_{p^r} variable node are 'connected' via the local code. Local code is an $(\frac{p^r-1}{p^m-1}, \frac{r}{m})$ code, and therefore, $\frac{r}{m}$ out of the $\frac{p^r-1}{p^m-1} \mathbb{F}_{p^m}$ nodes can be thought of as representing *information symbols*, and others *parity symbols*. We propose first picking a suitable set of $\frac{r}{m}$ nodes to represent information symbols, and initializing only these as in (5) and (6). For rest of the nodes, those that represent parity symbols of the local code, an additional scaling factor δ ($0 \leq \delta \leq 1$) will be used in (6). Our simulation results show that this modification helps in reducing propagation of errors in channel information, but δ has to be optimized per code. Equation (7) presents this modification.

$$\underline{L}_{n,i}^c(j) = \delta \cdot \log \frac{\mathbf{p}_{n,i}^c(j)}{\mathbf{p}_{n,i}^c(0)} \quad j = 0, 1, \dots, (p^m - 1) \quad (7)$$

After initialization, operations of the decoder would be similar to those of a decoder for a code over \mathbb{F}_{p^m} except for a couple of minor modifications that are explained in the following.

2) *Distinguishing Local Checks from Regular Checks*: Expanded graphs contain two different types of check nodes; local check nodes that represent dependencies between CPVs, and regular ones, resulting from expanding check nodes in the original graph. Here, local check nodes are only connected with \mathbb{F}_{p^m} nodes of a single \mathbb{F}_{p^r} variable node, whereas a regular check node will only be connected with one such. Thus, local check nodes do not represent relationships between different variables of the original code, and regular ones represent only those. This means that estimates from the two types of check nodes are based on two separate linear codes, and treating them similarly may not be the best approach to take.

As discussed in Section III B, local PCM may contain some short cycles, and these would be present in the expanded graph among the local check nodes. Estimates computed by a check node involved in one such cycle in two different iterations will be correlated with each other to some degree. This can make the estimates 'over-confident' of a variable node taking a particular value.

Taking into consideration the need to distinguish between estimates of local and regular check nodes, and also since local check nodes could be involved in short cycles, we propose using another scaling factor ψ ($0 < \psi < 1$) with estimates of local check nodes. In the literature, similar approaches have been taken to mitigate effects of short cycles with satisfactory results, for example in [20].

Combining probability estimates with this modification, at some variable node i of the expanded graph during k 'th decoding iteration, is given by (8). There, \underline{L}_i is the initial estimate for node i , $\underline{R}_i^{(k)}$ is the combined estimate, and $r_{j \rightarrow i}^{(k)}$ is the estimate sent from j 'th check node to i 'th variable node, in k 'th iteration. \underline{L}_i , $\underline{R}_i^{(k)}$ and $r_{j \rightarrow i}^{(k)}$ are all length p^m vectors of log or LLR values. N_i^r and N_i^l are, respectively, sets of regular and local check nodes in the neighborhood of node i .

$$\underline{R}_i^{(k)} = \underline{L}_i + \sum_{j \in N_i^r} r_{j \rightarrow i}^{(k)} + \psi \cdot \sum_{j \in N_i^l} r_{j \rightarrow i}^{(k)} \quad (8)$$

Similar to scaling factor δ used in initialization, ψ also has to be optimized per code.

3) *Testing for Convergence*: In iterative decoding of NB-LDPC codes, a tentative decision is taken by every variable node in each iteration to test whether the decoder has converged to a valid codeword. If so, then the check-sum at every check node should be zero, and the decoding process can be terminated. Same approach may be taken when decoding on expanded graphs. Tentative decision at each variable node would be the \mathbb{F}_{p^m} element most likely for the node, and check-sums would be computed at all check nodes, including local ones. Output of the decoder would be a vector of \mathbb{F}_{p^m} elements that's $\frac{p^r-1}{p^m-1}$ times longer than the original code length. Original codeword can be recovered by mapping each set of $\frac{p^r-1}{p^m-1}$ \mathbb{F}_{p^m} elements to a single \mathbb{F}_{p^r} element, via the 'local' code, as discussed in Section III B.

Even though we replace each \mathbb{F}_{p^r} node with $\frac{p^r-1}{p^m-1}$ \mathbb{F}_{p^m} nodes, just $\frac{r}{m}$ \mathbb{F}_{p^m} elements are sufficient to represent a single \mathbb{F}_{p^r} element, which is also evident from the local code. This observation leads to a slightly easier approach to checking convergence. Rather than deciding on all \mathbb{F}_{p^m} nodes of a single \mathbb{F}_{p^r} variable node, we propose only using the $\frac{r}{m}$ nodes selected as the 'information symbols' of the local code. Most likely \mathbb{F}_{p^m} elements of these would map to a single \mathbb{F}_{p^r} element, once more through the local code. Check-sums of original parity-check equations can then be computed with these \mathbb{F}_{p^r} elements. Note that even though now check-sums are computed over the larger field, computations involve only simple field arithmetic, and also there will be a significant reduction in the number of computations required when compared with the straight-forward approach.

With these three modifications, any iterative soft-decoding algorithm [3]-[8] proposed for NB-LDPC codes may be used with expanded graphs. This allows a large number of decoding strategies. For applications where decoding latency is the primary concern, a simplification of QSPA, such as min-max

decoding [7], can be used with an expanded graph, thereby achieving the complexity gains of both the simplification and the expansion. Section V presents some results from simulations where a few of these different strategies were evaluated.

V. SIMULATION RESULTS

In this section, we compare error-correcting performance of decoding schemes discussed in Section IV against some existing decoding algorithms for NB-LDPC codes. We consider different expansions of the same Tanner graph (different m for a fixed graph), and use QSPA [3], and one of its well-known simplifications, min-max decoding [7], with each expansion. QSPA and min-max decoding are also used on the original graph, along with max-log-SP algorithm [5], which is a special case of the extended min-sum (EMS) algorithm [6], where n_m and n_c are set to the maximum possible values of the size of the field and check-node degree, respectively. All algorithms were implemented in LLR domain [5], and simulations were done over the BI-AWGN channel, with maximum decoding iterations of 50 for all. Algorithms over expanded graphs were used with the modifications proposed in Section IV, and scaling factors δ and ψ were optimized through simulations. In the following, we use the algorithm along with the field size to refer to different decoding setups, for example, we let \mathbb{F}_{p^r} -QSPA denote QSPA on a graph over \mathbb{F}_{p^r} , and etc.

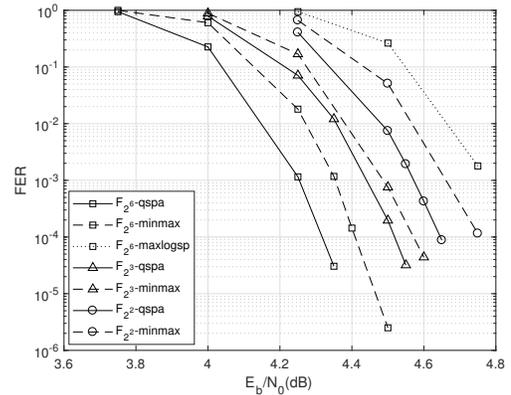


Fig. 3: FER Perf. with a (1998,1776) code over $GF(2^6)$ (\mathcal{C}_1)

Fig. 3 shows FER performance of decoding schemes with \mathcal{C}_1 , a rate 0.89 code over \mathbb{F}_{2^6} , of 1998 symbols in length. Code was generated through random re-labeling of a regular binary LDPC code of column weight 4, obtained from [21].

In Fig. 3, we observe that decoding algorithms over expanded graphs perform close to the best known decoder, QSPA over the original graph. In fact, QSPA over the \mathbb{F}_{2^3} expansion performs within 0.2dB of \mathbb{F}_{2^6} -QSPA, at a FER of 10^{-4} . When using the \mathbb{F}_{2^2} expansion, this widens slightly to 0.3dB. While min-max decoding over the original graph has a gap of only about 0.08dB with \mathbb{F}_{2^6} -QSPA, it should be noted that decoding is still over \mathbb{F}_{2^6} , and thus, it is more complex than QSPA over expanded graphs, as made evident in

Section VI. Interestingly, the max-log-SP algorithm (which is simpler than QSPA), is outperformed by all proposed decoding schemes, even though it operates in the original field. Max-log-SP shows a gap of about 0.55dB with \mathbb{F}_{2^6} -QSPA, at a FER of 10^{-3} . We also evaluate performance of min-max decoding over expanded graphs, which is quite satisfactory. In the case of \mathbb{F}_{2^3} expansion, min-max only has a gap of 0.06dB with \mathbb{F}_{2^3} -QSPA, while the gap between \mathbb{F}_{2^2} -QSPA and \mathbb{F}_{2^2} -min-max is around 0.1dB. These two decoding setups, which have complexity advantages of expansion and simplification, manage to outperform the max-log-SP algorithm over the original graph. Optimum values for scaling factors (δ, ψ) were found to be $(0.75, 0.25)$ for \mathbb{F}_{2^3} -QSPA and \mathbb{F}_{2^2} -QSPA, $(0, 0.3)$ for \mathbb{F}_{2^3} -min-max, and $(0, 0.4)$ for \mathbb{F}_{2^2} -min-max.

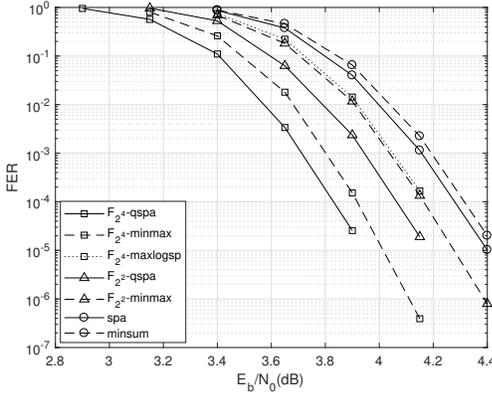


Fig. 4: FER Perf. with a (1000,861) code over $GF(2^4)$ (C_2)

Fig. 4 illustrates the FER performance of proposed schemes with a rate 0.861 code over \mathbb{F}_{2^4} , of 1000 symbols in length (C_2). The C_2 was generated by re-labeling a regular binary graph of column weight 3, constructed with the progressive edge growth algorithm [22]. For this code, we consider expansions over \mathbb{F}_{2^2} and \mathbb{F}_2 . Expansion over \mathbb{F}_2 is of special interest, since it results in a *binary* graph. When using this binary graph, we replace QSPA and min-max decoding with SPA and its well-known simplification, min-sum algorithm (MSA). Unique features and advantages offered by the binary expansion have been explored separately in [17].

Fig. 4 shows that the performance losses of the proposed schemes are quite small in this case as well. Gap between using QSPA on the original graph and its \mathbb{F}_{2^2} expansion is less than 0.3dB at a FER of 10^{-4} . Loss of replacing QSPA by its simplification min-max decoding is about 0.1dB for both original and expanded graphs. With C_2 , max-log-SP algorithm seems to perform a bit better than with C_1 . Here, its performance is very similar to that of using min-max algorithm on \mathbb{F}_{2^2} expansion, with a gap of close to 0.4dB with \mathbb{F}_{2^4} -QSPA, at a FER of 10^{-4} . When compared with QSPA on the original graph, using SPA on the binary graph results in a 0.5dB loss in performance. Simplifying SPA to MSA only loses a further 0.05dB. Although a 0.5dB loss seems significant, as explored in [17], decoding on a binary

expansion provides unique advantages in decoding complexity and hardware implementations. Optimum values for scaling factors (δ, ψ) here were $(0.5, 0.25)$ for \mathbb{F}_{2^2} -QSPA, SPA, and MSA, and $(0, 0.3)$ for \mathbb{F}_{2^2} -min-max.

Simulation results show that decoding algorithms implemented on proposed graph expansions are capable of performing quite close to those using the original graph. For any algorithm, performance gap of decoding on the expanded graph and using the original widens when the size of the field used for the expansion decreases. With a few different graph expansions possible, many decoding options become available for any given code. As discussed in the next Section, all these decoding schemes provide attractive complexity gains, with different levels of performance-complexity trade-offs.

VI. DECODING COMPLEXITY

In the following, we analyze the complexities of some decoding schemes on expanded graphs. We consider implementing the two popular versions of QSPA, LLR-QSPA [5] and FFT-QSPA [4], and also min-max decoding [7] on proposed expansions and compare them in terms of complexity with the same algorithms implemented on the original graph. Since NB-LDPC codes are most often defined over finite fields of characteristic 2 [3], a code over \mathbb{F}_{2^r} , where r has a factor m , is used in the complexity analysis. Complexities of the two major steps in iterative decoding, check node operations and variable node operations, are compared separately. For the comparison, we consider operations at a single node of each type during one iteration. Since the proposed expansions replace each node over \mathbb{F}_{2^r} with $E_f = \frac{2^r-1}{2^m-1}$ nodes, complexity of all those is the total complexity for the decoding schemes on expanded graphs. As explained in Section III B, these graphs also have the additional feature of *local* check nodes. Since $L_f = \frac{2^r-1}{2^m-1} - \frac{r}{m}$ such nodes are included per variable node of the original graph, their complexities are included with that of variable nodes.

At hardware level, apart from the number of operations, the type of operation also affects the complexity. It is well-known that operations such as multiplications are more complex than comparisons [19]. Therefore, we consider the number of operations of a few different types; comparisons (Comp), additions/subtractions (Add), multiplications/divisions (Mult) and table look-ups (LUT). Note that \max^* operation in LLR-QSPA can be performed with one comparison, two additions, and one table look-up [5], and that transformation between log and probability domain, required in FFT-QSPA, can be carried out with look-up tables. It has also been assumed that the forward-backward approach [7] is used in check node operations of the three algorithms. Further, cost of permuting probability vectors has been disregarded, since its impact on total complexity is negligible.

Table III lists complexities of check node operations in each decoding setup, while Table IV considers variable node operations. Average degrees of a check node and a variable node in the original graph are denoted with d_c and d_v , while d_l denotes the average degree of a local check node. As discussed in Section III B, local PCM is formed with Hamming codewords, and therefore it should always be possible to set $d_l = 3$. Due

to these new check nodes, average variable node degree would slightly increase in the expanded graphs, and we denote this new value with \tilde{d}_v , given by

$$\tilde{d}_v = d_v + \frac{L_f \times d_l}{E_f} \quad (9)$$

Substituting the values for E_f , L_f and d_v yields

$$\tilde{d}_v = d_v + 3 - 3 \times \frac{r(2^m - 1)}{m(2^r - 1)} \quad (10)$$

Note that degrees of *regular* check nodes in the expanded graphs remain d_c . When presenting complexities of decoding schemes on these graphs, we let E_f , L_f and \tilde{d}_v denote the number of new nodes per original node, number of local check nodes, and average variable node degree, respectively.

TABLE III: Check Node Complexity

Algorithm	Check Node Operations			
	Comp	Add	Mult	LUT
\mathbb{F}_{2^r} -LLR-QSPA	$(3d_c - 4) \times 2^r(2^r - 1)$	$(3d_c - 4) \times 2^r(3 \cdot 2^r - 2)$	0	$(3d_c - 4) \times 2^r(2^r - 1)$
\mathbb{F}_{2^m} -LLR-QSPA	$E_f(3d_c - 4) \times 2^m(2^m - 1)$	$E_f(3d_c - 4) \times 2^m(3 \cdot 2^m - 2)$	0	$E_f(3d_c - 4) \times 2^m(2^m - 1)$
\mathbb{F}_{2^r} -FFT-QSPA	0	$2d_c \times 2^r r$	$(2d_c - 1) \times 2^r$	$2d_c \times 2^r$
\mathbb{F}_{2^m} -FFT-QSPA	0	$E_f \cdot 2d_c \times 2^m m$	$E_f(2d_c - 1) \times 2^m$	$E_f \cdot 2d_c \times 2^m$
\mathbb{F}_{2^r} -Min-Max	$(3d_c - 4) \times 2^r(2 \cdot 2^r - 1)$	0	0	0
\mathbb{F}_{2^m} -Min-Max	$E_f(3d_c - 4) \times 2^m(2 \cdot 2^m - 1)$	0	0	0

From Table III, it can be seen that complexity gains of proposed schemes at check node operations depend on the decoding algorithm being used. For both LLR-QSPA and min-max decoding, using an expanded graph instead of the original results in a significant reduction in complexity, while for FFT-QSPA, the gains are modest. In the case of LLR-QSPA, using the original graph requires approximately $3d_c \times 2^{2r}$ comparisons, additions, and table look-ups, which results in an overall complexity of $\mathcal{O}(2^{2r})$. However, with the expansion over \mathbb{F}_{2^m} , there are only approximately $3d_c \times 2^{r+m}$ operations of each type, which reduces overall complexity to $\mathcal{O}(2^{r+m})$. This is a significant gain, especially in the cases with a large r , and we feel that, as a trade-off, the small performance losses observed in Section V are justifiable. Using an expanded graph can reduce the complexity order from $\mathcal{O}(2^{2r})$ to $\mathcal{O}(2^{r+m})$ in check node operations of min-max decoding as well. It should be noted that although they are of the same complexity order, min-max decoding is simpler than LLR-QSPA, since only comparisons are required. Gains of the proposed scheme reduce in the case of FFT-QSPA. Here, the number of multiplications and table look-ups required are almost the same (approximately $2d_c \times 2^r$) when using the original graph or an expanded one. There is a slight reduction in the number of additions though, from approximately $2d_c \times 2^r r$ to $2d_c \times 2^r m$. Thus, the overall complexity of FFT-QSPA on an expanded graph is $\mathcal{O}(2^r m)$, slightly lower than $\mathcal{O}(2^r r)$ on the original graph.

TABLE IV: Variable Node Complexity

Algorithm	Variable Node Operations			
	Comp	Add	Mult	LUT
\mathbb{F}_{2^r} -LLR-QSPA	$2^r - 1$	$2d_v \times 2^r$	0	0
\mathbb{F}_{2^m} -LLR-QSPA	$(r/m) \times \frac{(2^m - 1)}{5L_f \times}$	$E_f \cdot 2d_v \times \frac{2^m}{5L_f \times}$	0	0
Local Checks	$2^m(2^m - 1)$	$2^m(3 \cdot 2^m - 2)$	0	$2^m(2^m - 1)$
\mathbb{F}_{2^r} -FFT-QSPA	$2^r - 1$	$2d_v \times 2^r$	0	0
\mathbb{F}_{2^m} -FFT-QSPA	$(r/m) \times \frac{(2^m - 1)}{6L_f \times}$	$E_f \cdot 2d_v \times \frac{2^m}{6L_f \times}$	0	0
Local Checks	0	$\frac{2^m}{2^m m}$	$\frac{5L_f \times}{2^m}$	$\frac{6L_f \times}{2^m}$
\mathbb{F}_{2^r} -Min-Max	$(d_v + 1) \times 2^r$	$3d_v \times 2^r$	0	0
\mathbb{F}_{2^m} -Min-Max	$(E_f \cdot d_v + r/m) \times 2^m$	$E_f \cdot 3d_v \times \frac{2^m}{5L_f \times}$	0	0
Local Checks	$2^m(2 \cdot 2^m - 1)$	0	0	0

When considering variable node operations of decoding schemes on expanded graphs, we include the complexity of the L_f local check nodes added for each *original* variable node. Note that complexity of one such node can be derived by substituting $d_l = 3$ as the node degree, and 2^m as the field size, in the expressions for the respective algorithm in Table III. Due to this additional cost, complexity at variable nodes are *higher* in proposed schemes. However, this complexity increase is not sufficiently high to completely offset the gain obtained at check node operations, especially for LLR-QSPA and min-max decoding. As Table IV shows, complexity orders of these algorithms change from $\mathcal{O}(2^r)$ on the original graph to $\mathcal{O}(2^{r+m})$ on an expanded one, while in Table III, this change is from $\mathcal{O}(2^{2r})$ to $\mathcal{O}(2^{r+m})$ at check node operations. Hence, the overall gain is still significant for LLR-QSPA and min-max decoding, especially for larger values of r . In the case of FFT-QSPA, the complexity increase is comparatively smaller, from $\mathcal{O}(2^r)$ to $\mathcal{O}(2^r m)$. Since its gain at check nodes was also quite modest, the overall complexity gain would be minimal.

Tables III and IV demonstrate that decoding on expanded graphs is advantageous in terms of asymptotic complexity, while the actual performance gains would depend on parameters of the code used, such as field sizes, code length, rate, and average node degrees. In Table V, we consider complexities of some decoding schemes used in Section V with \mathcal{C}_1 , a code over \mathbb{F}_{2^6} with the codeword length 1998 and code rate 0.89. In this case, the original graph is over \mathbb{F}_{2^6} , and expansions over \mathbb{F}_{2^3} and \mathbb{F}_{2^2} are used for decoding. Table V presents complexities of using LLR-QSPA, FFT-QSPA, and min-max decoding on all three graphs, in terms of number of operations of each type per iteration. For decoding schemes over expansions, we also present the number of operations required as a percentage of the requirement when using the same algorithm with the original graph.

In Table V, we observe that using LLR-QSPA on expanded graphs offers exceptional complexity gains for \mathcal{C}_1 . Less than 20% of the operations for the original graph are required when

TABLE V: Number of Operations per Iteration with \mathcal{C}_1

Algorithm	Number of Operations ($\times 10^9$)			
	Comp	Add	Mult	LUT
\mathbb{F}_{2^6} -LLR-QSPA	932.17	2817.73	-	930.91
\mathbb{F}_{2^3} -LLR-QSPA	155.8 ($\approx 17\%$)	507.01 ($\approx 18\%$)	-	155.52 ($\approx 17\%$)
\mathbb{F}_{2^2} -LLR-QSPA	79.94 ($\approx 8\%$)	287.93 ($\approx 10\%$)	-	79.76 ($\approx 8\%$)
\mathbb{F}_{2^6} -FFT-QSPA	1.26	71.61	10.09	10.23
\mathbb{F}_{2^3} -FFT-QSPA	0.28 ($\approx 22\%$)	72.89 ($\approx 101\%$)	16.94 ($\approx 168\%$)	18.22 ($\approx 178\%$)
\mathbb{F}_{2^2} -FFT-QSPA	0.18 ($\approx 14\%$)	66.17 ($\approx 92\%$)	20.43 ($\approx 202\%$)	22.06 ($\approx 215\%$)
\mathbb{F}_{2^6} -Min-Max	1882.99	15.35	-	-
\mathbb{F}_{2^3} -Min-Max	342.7 ($\approx 18\%$)	27.33 ($\approx 178\%$)	-	-
\mathbb{F}_{2^2} -Min-Max	197.38 ($\approx 10\%$)	33.09 ($\approx 215\%$)	-	-

using the \mathbb{F}_{2^3} expansion. This reduces further with the \mathbb{F}_{2^2} expansion, to less than 10%. These gains correspond to speed-ups of *more than 5 times* in the \mathbb{F}_{2^3} case, and *more than 10 times* in the \mathbb{F}_{2^2} case. Considering that the performance losses, as shown in Section V, are only 0.2dB and 0.3dB, the complexity gains are very attractive. With FFT-QSPA though, using expansions are not particularly advantageous. Only gain of \mathbb{F}_{2^3} expansion, when compared with using the algorithm on the original \mathbb{F}_{2^6} graph, is in the number of comparisons required. Both decoding setups use a similar number of additions, while the setup on the expanded graph needs significantly more multiplications and table look-ups. This is due to the operations of local check nodes, which are absent in the original graph. With \mathbb{F}_{2^2} expansion, the number of comparisons reduces further, and the number of additions used is also slightly lesser than that of the \mathbb{F}_{2^6} case. Since \mathbb{F}_{2^2} expansion has more local check nodes than the \mathbb{F}_{2^3} one, the number of multiplications and table look-ups have increased significantly. Thus, for \mathcal{C}_1 , using FFT-QSPA with any of the two expansions is *more complex* than implementing on the original graph. The case of min-max decoding is very similar to that of LLR-QSPA; complexity gains are significant, and they are higher when the size of the field used is smaller. Due to local check node operations, the number of additions in proposed schemes is higher than in the original algorithm. Nevertheless, since the reduction in the number of comparisons is much higher in magnitude, min-max decoding on expanded graphs is significantly less complex.

Majority of existing algorithms are of complexity order $\mathcal{O}(2^{2r})$ for a code over \mathbb{F}_{2^r} , and implementing those algorithms on graph expansions results in significant complexity gains with minimal performance losses. For algorithms whose complexity order is not polynomial in field size, such as FFT-QSPA, the new strategy may not be advantageous. But as [19] pointed out, out of the two variants of QSPA, LLR-QSPA is more suitable for hardware implementations, due to better numerical stability of LLR domain operations. Therefore, the strategy proposed in this paper could be applied to reduce decoding complexity in most practical applications that adopt NB-LDPC codes. In particular, our proposed strategy enables

to decode a code defined over a large field using a graph over a much smaller field, while providing a good performance and complexity tradeoff, leading to a practical solution to decoding NB-LDPC codes.

VII. CONCLUSIONS

In this paper, we proposed a new method to expand a Tanner graph of a NB-LDPC code over \mathbb{F}_{p^r} into a graph over \mathbb{F}_{p^m} , where m is a factor of r . Most decoding algorithms proposed for NB-LDPC codes can be adapted to use these expanded graphs with simple modifications. This offers a number of different decoding options for any given code, with a different performance-complexity trade-off. Simulation results show that, in general, decoding on expanded graphs provide significant complexity gains, while performance losses are minimal. It may be interesting to note that the proposed expansion may find applications beyond decoding NB-LDPC codes.

REFERENCES

- [1] R. G. Gallager, "Low-density parity-check codes", *IRE Transactions on Information Theory*, vol. IT-8, pp. 21-28, Jan. 1962
- [2] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices", *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 399-431, Mar. 1999
- [3] M. C. Davey, and D. J. C. Mackay, "Low-density parity check codes over $GF(q)$ ", *IEEE Communication Letters*, vol. 2, no. 6, pp. 165-167, June 1998
- [4] L. Barnault, and D. Declercq, "Fast decoding algorithm for LDPC over $GF(2^q)$ ", *Proceedings of IEEE Information Theory Workshop*, Paris, France, Apr. 2003
- [5] H. Wymeersch, H. Steendam, and M. Moeneclaey, "Log-domain decoding of LDPC codes over $GF(q)$ ", *Proceedings of IEEE International Conference on Communications*, Paris, France, Jun. 2004
- [6] D. Declercq, and M. Fossorier, "Decoding algorithms for nonbinary LDPC codes over $GF(q)$ ", *IEEE Transactions on Communications*, vol. 55, no. 4, pp. 633-643, Apr. 2007
- [7] V. Savin, "Min-Max decoding for non binary LDPC codes", *Proceedings of IEEE International Symposium on Information Theory*, Toronto, Canada, July 2008
- [8] E. Li, D. Declercq, and K. Gunnam, "Trellis-based extended min-sum algorithm for non-binary LDPC codes and its hardware structure", *IEEE Transactions on Communications*, vol. 61, no. 7, pp. 2600-2611, July 2013
- [9] J. O. Lacruz, F. Garcia-Herrero, D. Declercq, and J. Valls, "Simplified trellis minmax decoder architecture for nonbinary low-density parity-check codes", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 9, pp. 1783-1792, Sep. 2015
- [10] V. Savin, "Binary linear-time erasure decoding for non-binary LDPC codes", *Proceedings of IEEE Information Theory Workshop*, Taormina, Italy, Oct. 2009
- [11] Y. Yu, W. Chen, J. Li, X. Ma, and B. Bai, "Generalized binary representation for the nonbinary LDPC code with decoder design", *IEEE Transactions on Communications*, vol. 62, no. 9, pp. 3070-3083, Sep. 2014
- [12] M. Zhang, K. Cai, Q. Huang, and S. Yuan, "On bit-level decoding of nonbinary LDPC codes", *IEEE Transactions on Communications*, vol. 66, no. 9, pp. 3736-3748, Sep. 2018
- [13] J. J. Rothman, "Advanced modern algebra", 1st ed. Prentice Hall, 2003, pp. 116-218
- [14] S. Lin, and D. J. Costello, "Error control coding", Upper Saddle River, NJ, USA: Pearson Education, 2004

- [15] B. Amiri, J. Kliewer, and L. Dolecek, "Analysis and enumeration of absorbing sets for non-binary graph-based codes", *IEEE Trans. on Comm.*, vol. 62, no. 2, pp. 398-409, Feb. 2014
- [16] S. Cho, K. Cheun, and K. Yang, "A message-passing algorithm for counting short cycles in nonbinary LDPC codes", *Proc. of IEEE ISIT*, Vail, CO, USA, June 2018
- [17] V. B. Wijekoon, Emanuele Viterbo, Yi Hong, R. Micheloni, and A. Marelli, "A Novel Graph Expansion and a Decoding Algorithm for NB-LDPC Codes", *IEEE Trans. on Comm.*, vol. 68, no. 3, pp. 1358 - 1369, Mar. 2020
- [18] Chao-Yu Chen, Qin Huang, Chi-chao Chao, and Shu Lin, "Two low-complexity reliability-based message-passing algorithms for decoding non-binary LDPC codes", *IEEE Transactions on Communications*, vol. 58, no. 11, Nov. 2010
- [19] C. Spagnol, E.M. Popovici, and W.P. Marnane, "Hardware implementation of $GF(2^m)$ LDPC decoders", *IEEE Trans. Circuits Syst. I*, vol. 56, no. 12, pp. 2609-2620, Mar. 2009
- [20] J. Jiang, and K. R. Narayanan, "Iterative soft-input soft-output decoding of Reed-Solomon codes by adapting the parity-check matrix", *IEEE Trans. on Inf. Th.*, vol. 52, no. 8, pp. 3746-3756, Aug. 2006
- [21] D. J. C. Mackay, "Encyclopedia of Sparse Graph Codes", [Online]. Available: <http://www.inference.org.uk/mackay/codes/data.html>.
- [22] X.-Y. Hu, E. Eleftheriou, and D.M. Arnold, "Regular and irregular progressive edge-growth tanner graphs", *IEEE Trans. on Inf. Th.*, vol. 51, no. 1, pp. 386-398, Jan. 2005