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 \( \mathbb{F}_{2^r} \) contains a unique subfield \( \mathbb{F}_{2^m} \), 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. \( \alpha \)-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 \( \alpha \). It is easy to verify that multiplying all the elements in \( G \) with some \( \alpha^t \in \mathbb{F}_{p^r} \) 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 \( \alpha \), are called \( \alpha \)-connected subgroups.

If subgroups \( G_1 \) and \( G_2 \) are \( \alpha \)-connected, and so are \( G_2 \) and \( G_3 \), then clearly \( G_1 \) and \( G_3 \) are also \( \alpha \)-connected. This yields the following definition.
Definition 2. A set of subgroups $S = \{G_1, \ldots, G_n\}$ of $\mathbb{F}_{p^m}$ is an $\alpha$-connected set if
1) each $G_i \in S$ is $\alpha$-connected with all other $G_j \in S$.
2) $G_i \in S$ is $\alpha$-connected with some $G$, then $G \in S$.

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

Lemma 1. Consider $\mathbb{F}_{p^r}$ and let $m$ divide $r$ (i.e., $m \mid r$). Then the smallest possible $\alpha$-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 $\alpha$-connected set is the minimum value $i \in \{1, \ldots, p^r - 2\}$ 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 $\alpha^i m$. Note that $i_m$ is the minimum non-zero power of $\alpha$ in $S_{\alpha^i m}$. If $\alpha^i m \cdot G = G$, then clearly $\alpha^{i_m} \cdot G = G$, for any $\alpha^{i_m} \in S_{\alpha^i m}$. Since we are focused on the minimum, we only consider $i_m$, for which the following relation holds.

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

Using homomorphism $\psi$ between the additive groups of $\mathbb{F}_{p^r}$ and $\mathbb{F}_{p^m}$, we can define a homomorphism $\psi^*$ between the subfield $\mathbb{F}_{p^m}$ and $\mathbb{F}_{p^r}$ as follows.

Lemma 2. Let $G$ be a subgroup of order $p^m$ in $\mathbb{F}_p^m$, and let $\psi$ be some surjective homomorphism $\psi : H \rightarrow G$. The kernels of the subfield $\mathbb{F}_{p^m}$, $\mathbb{F}_{p^r}$, and $\mathbb{F}_{p^r}$ are $\alpha$-connected sets.

Proof: For $\psi : H \rightarrow G$, $\ker(\psi)$ is an additive subgroup of order $p^r - m$ in $\mathbb{F}_{p^r}$. Since $\psi(h^i) = \psi(\alpha^i h^i)$, it is clear that $\ker(\psi) = \alpha^i \ker(\psi)$. Thus, $\ker(\psi)$ is an additive subgroup of order $p^r - m$ in $\mathbb{F}_{p^r}$. Lemma 2 depends on the homomorphism $\psi$.

The cardinality of an $\alpha$-connected set generated as in Lemma 2 depends on the homomorphism $\psi$. Therefore, to construct the smallest $\alpha$-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 $\mathbb{F}_{p^m}$ as an extension of $\mathbb{F}_{p^r}$.

Lemma 3. Let $S_{\beta}$ be the set of elements in $\mathbb{F}_{p^r}$ generated by $\beta = \frac{p^r - 1}{p^m - 1}$. 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, there should be only one unique subgroup of a specific order in $K'$. The set of elements generated by $\beta$, $S_{\beta}$, 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 by a polynomial $E_{\alpha^i}(x)$ over $\mathbb{F}_{p^m}$, of degree at most $(\frac{p^r - 1}{p^m - 1})$. In the case of elements belonging to $\mathbb{F}_{p^m}$ (for $\beta$), the polynomials would be of degree 0. The primitive polynomial $\Pi(x)$ for the representation is of degree $\frac{p^r - 1}{p^m - 1}$ and, since $\Pi(x)$ is irreducible, it must have a non-zero constant term. Based on this representation, we define a homomorphism $\psi^*$ 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 the homomorphism $\psi^*$ in the method proposed in Lemma 2 generates an $\alpha$-connected set of minimum cardinality.

Lemma 4. The set of kernels of homomorphisms $\psi^*_i(h') = \psi^*(\alpha^i h')$, for $i \in \{0, 1, \ldots, p^r - 2\}$ form an $\alpha$-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 $H' = \{F_{p^r}, + \}$ and $H = \{F_{p^m}, + \}$. Since $|H| = p^m$, $\ker(\psi^*)$ is a subgroup of order $p^{\prime - m}$ in $H'$. Using $\psi^*$ in Lemma 2, it is clear that kernels of $\psi^*_i(h') = \psi^*(\alpha^{-i}h')$ form an $\alpha$-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 $\alpha^{i_m} g_j = \gamma_j$, $j = 1, \ldots, p^{\prime - m}$. Let polynomial representations (in the extended form) of $\alpha^{i_m} g_j$ and $\gamma_j$ be $E_{\alpha^{i_m}}(x), E_g(x)$ and $E_{\gamma_j}(x)$, respectively. These are related by the mod $\Pi(x)$ polynomial multiplication

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

where $K_j(x)$ is some polynomial over $\mathbb{F}_{p^m}$. Since $g_j \in \ker(\psi^*_0)$, the constant term in $E_{\alpha^{i_m}}(x)$ is zero, which makes the constant term in $E_{\alpha^{i_m}}(x) E_g(x)$ also zero. Note that for any $\gamma_j \in \ker(\psi^*_0)$, constant term of $E_{\gamma_j}(x)$ should be zero as observed earlier, $\Pi(x)$ has a non-zero constant term, and therefore, for $\gamma_j \in \ker(\psi^*_0)$, $K_j(x)$ should be a polynomial with a zero constant term. For $\alpha^{i_m} \ker(\psi^*_0) = \ker(\psi^*_0)$, this should be true for $j = 1, \ldots, p^{\prime - m}$.

Polynomial representations of elements in $\ker(\psi^*_0)$ contains at least one polynomial of each possible degree, from $0$ to $\frac{p^m - 1}{p^r - 1} - 1$. Then, if $\deg(E_{\alpha^{i_m}}(x)) > 0$, for at least one value of $j$, $E_{\alpha^{i_m}}(x) E_g(x)$ would be of degree $\frac{p^m - 1}{p^r - 1}$. Since $\Pi(x)$ is also of degree $\frac{p^m - 1}{p^r - 1}$, this requires $K_j(x)$ to be a non-zero constant for that particular value of $j$. Therefore, for any $\gamma_j \in \ker(\psi^*_0)$ for all $j = 1, \ldots, p^{\prime - m}$, $\deg(E_{\alpha^{i_m}}(x)) = 0$. Thus, for any $j$, $K_j(x) = 0$. This requires $\alpha^{i_m} \ker(\psi^*_0) = \ker(\psi^*_0)$, and since we require the minimum, $i_m = \frac{p^m - 1}{p^r - 1}$.

Thus, using the homomorphism $\psi^*$ in Lemma 2, it is possible to construct an $\alpha$-connected set of additive subgroups of order $p^{\prime - m}$, that has the minimum cardinality $\frac{p^m - 1}{p^r - 1}$, 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 $\alpha$-connected subgroups of order $p^{\prime - m}$ in $\mathbb{F}_{p^r}$, constructed as detailed in the previous section. We represent this special $\alpha$-connected set by $\Theta_m$ 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 $p^{\prime - m}$. 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., $Q_\psi$ contains the $p^m$ cosets of $\ker(\psi)$, including the trivial coset (ker(ψ)) itself. In the isomorphism between $Q_\psi$ 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 $Q_\psi = \{C_0^\psi, C_1^\psi, \ldots, C_{p^m - 1}^\psi\}$, where each $C_j^\psi$ represents some coset of $\ker(\psi)$, with $C_0^\psi$ representing the trivial coset. Cosets contain elements in $\mathbb{F}_{p^r}$, and using the multiplicative properties of the field, we define a multiplicative operation on $Q_\psi$ as follows.

Definition 4. Operation $\beta Q_\psi$, for some $\beta \in \mathbb{F}_{p^r}$, is defined as $\beta Q_\psi = \{\beta C_0^\psi, \beta C_1^\psi, \ldots, \beta C_{p^m - 1}^\psi\}$.

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

Lemma 5. Let $G_{\psi_1}$ and $G_{\psi_2}$ be two $\alpha$-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 $Q_{\psi_1}$.

Proof: Let $Q_{\psi_1} = \{C_0^{\psi_1}, \ldots, C_{p^m - 1}^{\psi_1}\}$ and $Q_{\psi_2} = \{C_0^{\psi_2}, \ldots, C_{p^m - 1}^{\psi_2}\}$. Here the trivial cosets $C_0^{\psi_2}$ are the subgroups themselves, and all the cosets can be represented using the respective subgroup and some coset leader term as follows:

$$Q_{\psi_1} = \{G_{\psi_1}, G_{\psi_1} + l_1^{\psi_1}, \ldots, G_{\psi_1} + p_{p^m - 1}^{\psi_1}\}$$

$$Q_{\psi_2} = \{G_{\psi_2}, G_{\psi_2} + l_1^{\psi_2}, \ldots, G_{\psi_2} + p_{p^m - 1}^{\psi_2}\}$$

Using the multiplication operation on $Q_{\psi_2}$ yields

$$\alpha^k Q_{\psi_2} = \{\alpha^k G_{\psi_2}, \alpha^k G_{\psi_2} + \alpha^k l_1^{\psi_2}, \ldots, \alpha^k G_{\psi_2} + \alpha^k p_{p^m - 1}^{\psi_2}\}$$

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

$$\alpha^k Q_{\psi_2} = \{G_{\psi_1}, G_{\psi_1} + \alpha^k l_1^{\psi_2}, \ldots, G_{\psi_1} + \alpha^k p_{p^m - 1}^{\psi_2}\}$$

is a set containing $G_{\psi_1}$, and its $(p^m - 1)$ proper cosets, albeit the coset leader terms could have changed. Thus, $Q_{\psi_1}$ 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 $Q_{\psi_1}$.

The homomorphisms $\psi^*_i$ we use in constructing the smallest $\alpha$-connected set, $\Theta_m$, are all surjective. $\Theta_m$ consists of the kernels of these homomorphisms, and it is possible to construct a set of $\alpha$-connected subgroups with those kernels. Let that set be $\Theta_m = \{Q_{\psi_1}^i : i = 0, \ldots, \frac{p^m - 1}{p^r - 1} - 1\}$. Since all $\psi^*_i$’s are surjective, each $Q_{\psi_1}^i$ is isomorphic to $H$, the additive group of $\mathbb{F}_{p^r}$. Since $\Theta_m$ is $\alpha$-connected, according to Lemma 5, multiplying some $Q_{\psi_1}^i \in \Theta_m^Q$ by some power of $\alpha$ results in a permutation of some other $Q_{\psi_1}^j \in \Theta_m^Q$.

These observations about $\Theta_m^Q$ 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 $\Theta_m^Q$. Then, for each variable node, $p^m - 1$ 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 $\Theta_m^Q$ is constructed using the smallest $\alpha$-connected set $\Theta_m$, 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 $v_3$, $p_{c,v_3}$. Since these vectors are of length $r$, convolution will be of complexity order $O(p^{2r})$. Now assume we have to compute some $i$’th CPV of $v_3$, $p_{c,v_3}$. 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 $\Theta_m^Q$ are isomorphic to the additive group of $\mathbb{F}_{p^m}$, computation of $p_{c,v_3}$ 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 $O(p^{2m})$. However, with $|\Theta_m^Q| = p^{r-1}$, that many CPVs will have to be computed, resulting in an overall complexity of $p^{3m} \times p^{r-1} \approx O(p^{m+r})$. 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, $\Theta_m^Q$, will be of cardinality $p^{r-1}$. 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 CPV’s 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 $p^{r-1}$ $\mathbb{F}_{p^m}$ 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 $Q_j \in \Theta_m^Q$ when multiplied with $\alpha^k$. 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 $Q_j$, 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 $\rho$ from a code over $\mathbb{F}_{2^4}$, where $\alpha$ denotes a primitive of the field.

$$\rho = \alpha^4 v_1 + \alpha v_2 = 0$$  \hspace{1cm} (3)

Fig. 1 presents the initial expansion for $\rho$. 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 $\omega$ 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](image)

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

![Table I](image)

![Table II](image)
TABLE I: Quotient Groups in $\mathbb{F}_{2^2-2}^Q$

<table>
<thead>
<tr>
<th>$Q_0$</th>
<th>0, $\alpha, \alpha^2, \alpha^4$</th>
<th>$\alpha^2, \alpha^4, \alpha^7$</th>
<th>$\alpha^4, \alpha^7, \alpha^{11}$</th>
<th>$\alpha^7, \alpha^{11}, \alpha^{14}$</th>
<th>$\alpha^{11}, \alpha^{14}, \alpha^{19}$</th>
<th>$\alpha^{14}, \alpha^{19}, \alpha^{22}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q_1$</td>
<td>0, $\alpha, \alpha^2, \alpha^4$</td>
<td>$\alpha^2, \alpha^4, \alpha^7$</td>
<td>$\alpha^4, \alpha^7, \alpha^{11}$</td>
<td>$\alpha^7, \alpha^{11}, \alpha^{14}$</td>
<td>$\alpha^{11}, \alpha^{14}, \alpha^{19}$</td>
<td>$\alpha^{14}, \alpha^{19}, \alpha^{22}$</td>
</tr>
<tr>
<td>$Q_2$</td>
<td>0, $\alpha^2, \alpha^4, \alpha^7$</td>
<td>$\alpha^4, \alpha^7, \alpha^{11}$</td>
<td>$\alpha^7, \alpha^{11}, \alpha^{14}$</td>
<td>$\alpha^{11}, \alpha^{14}, \alpha^{19}$</td>
<td>$\alpha^{14}, \alpha^{19}, \alpha^{22}$</td>
<td>$\alpha^{19}, \alpha^{22}, \alpha^{25}$</td>
</tr>
<tr>
<td>$Q_3$</td>
<td>0, $\alpha^4, \alpha^7, \alpha^{11}$</td>
<td>$\alpha^7, \alpha^{11}, \alpha^{14}$</td>
<td>$\alpha^{11}, \alpha^{14}, \alpha^{19}$</td>
<td>$\alpha^{14}, \alpha^{19}, \alpha^{22}$</td>
<td>$\alpha^{19}, \alpha^{22}, \alpha^{25}$</td>
<td>$\alpha^{22}, \alpha^{25}, \alpha^{28}$</td>
</tr>
<tr>
<td>$Q_4$</td>
<td>0, $\alpha^7, \alpha^{11}, \alpha^{14}$</td>
<td>$\alpha^{11}, \alpha^{14}, \alpha^{19}$</td>
<td>$\alpha^{14}, \alpha^{19}, \alpha^{22}$</td>
<td>$\alpha^{19}, \alpha^{22}, \alpha^{25}$</td>
<td>$\alpha^{22}, \alpha^{25}, \alpha^{28}$</td>
<td>$\alpha^{25}, \alpha^{28}, \alpha^{31}$</td>
</tr>
</tbody>
</table>

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

\[
\begin{array}{ccc}
0 & : & 00000 \\
1 & : & 10111 \\
\alpha & : & 01100 \\
\alpha^2 & : & \omega001 \\
\alpha^4 & : & 01000 \\
\alpha^8 & : & 1100\omega \\
\alpha^{16} & : & \omega \omega \omega \\
\alpha^{32} & : & 00110 \\
\end{array}
\]

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 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 $2^4_{H_L}$ in 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 dependancies 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 $m | r$, can be summarized as follows.

1) Obtain the smallest set of $\alpha$-connected subgroups $\Theta_m$, using the homomorphism in Definition 3, and following the steps outlined in Lemma 4. Use that to derive the set of quotient groups $\Theta^Q_m$.

2) Map cosets of each $Q_i \in \Theta^Q_m$ with elements of $H = \{\mathbb{F}_{p^m}+, \mathbb{F}_{p^m} \}$, use 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 $\rho_i$ given by (3). The $\mathbb{F}_{2^4}$ graph is shaded grey, and the expansion to $\mathbb{F}_{2^5}$ 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
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 $\mathbf{p}_n^c$ 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.

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

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

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

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 $\frac{p^r - 1}{p^m - 1}$ $\mathbb{F}_{p^m}$ 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 $(\frac{p^r - 1}{p^m - 1}, \frac{p^r - 1}{p^m})$ code, and therefore, $\frac{p^r - 1}{p^m}$ 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 $\frac{p^r}{p^m}$ 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 $\delta$ ($0 \leq \delta \leq 1$) will be used in (6). Our simulation results show that this modification helps in reducing propagation of errors in channel information, but $\delta$ has to be optimized per code. Equation (7) presents this modification.

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

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 $\psi$ ($0 < \psi < 1$) 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, $L_i$ is the initial estimate for node $i$, $R^{(k)}_{j \rightarrow i}$ is the combined estimate, and $r^{(k)}_{j \rightarrow i}$ is the estimate sent from $j$'th check node to $i$'th variable node, in $k$'th iteration. $L_i$, $R^{(k)}_{j \rightarrow i}$ and $r^{(k)}_{j \rightarrow i}$ are all length $p^m$ vectors of $\log$ or LLR values. $N'_i$ and $N_i^j$ are, respectively, sets of regular and local check nodes in the neighborhood of node $i$.

$$R^{(k)}_{j \rightarrow i} = L_i + \sum_{j \in N'_i} r^{(k)}_{j \rightarrow i} + \psi_1 \sum_{j \in N_i^j} r^{(k)}_{j \rightarrow i}$$

(8)

Similar to scaling factor $\delta$ used in initialization, $\psi$ 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 $F_{p^m}$ 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 $F_{p^m}$ elements that’s $p^m-1$ times longer than the original code length. Original codeword can be recovered by mapping each set of $F_{p^m}$ elements to a single $F_{p^r}$ element, via the ‘local’ code, as discussed in Section III B.

Even though we replace each $F_{p^r}$ node with $F_{p^m}$ nodes, just $\frac{p^m-1}{p^r-1}$ $F_{p^m}$ elements are sufficient to represent a single $F_{p^r}$ 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 $F_{p^m}$ nodes of a single $F_{p^r}$ variable node, we propose only using the $\frac{p^m}{p^r}$ nodes selected as the ‘information symbols’ of the local code. Most likely $F_{p^m}$ elements of these would map to a single $F_{p^r}$ element, once more through the local code. Check-sums of original parity-check equations can then be computed with these $F_{p^r}$ 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 $m_{nc}$ 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 $\delta$ and $\psi$ 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 $F_{p^r}$-QSPA denote QSPA on a graph over $F_{p^r}$, and etc.

Fig. 3 shows FER performance of decoding schemes with $C_1$, a rate 0.89 code over $F_{2^{6}}$, 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 $F_{2^6}$ expansion performs within 0.2dB of $F_{2^6}$-QSPA, at a FER of $10^{-4}$. When using the $F_{2^3}$ expansion, this widens slightly to 0.3dB. While min-max decoding over the original graph has a gap of only about 0.08dB with $F_{2^6}$-QSPA, it should be noted that decoding is still over $F_{2^6}$, 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 0.55 dB with $F_{22}$-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 $F_{23}$ expansion, min-max only has a gap of 0.06 dB with $F_{23}$-QSPA, while the gap between $F_{22}$-QSPA and $F_{22}$-min-max is around 0.1 dB. 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 $(\delta, \psi)$ were found to be (0.75, 0.25) for $F_{23}$-QSPA and $F_{22}$-QSPA, (0, 0.3) for $F_{23}$-min-max, and (0, 0.4) for $F_{22}$-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 $F_{2^4}$, 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 $F_{2^2}$ and $F_{2^3}$. Expansion over $F_2$ 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 $F_{2^2}$ expansion is less than 0.3 dB at a FER of $10^{-4}$. Loss of replacing QSPA by its simplification min-max decoding is about 0.1 dB 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 $F_{2^2}$ expansion, with a gap of close to 0.4 dB with $F_{2^2}$-QSPA, at a FER of $10^{-4}$. When compared with QSPA on the original graph, using SPA on the binary graph results in a 0.5 dB loss in performance. Simplifying SPA to MSA only loses a further 0.05 dB. Although a 0.5 dB 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 $(\delta, \psi)$ here were (0.5, 0.25) for $F_{2^2}$-QSPA, SPA, and MSA, and (0, 0.3) for $F_{2^2}$-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 $F_{2^r}$, 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 with $E_f = \frac{2^{r-1} + 1}{2^r - 1}$ 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 $L_f = \frac{2^r - 1}{2^r - 1} - \frac{r}{2^r}$ 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

\[
\tilde{d}_v = d_v + \frac{L_f \times d_i}{E_f}
\]

Substituting the values for \( E_f, L_f \) and \( d_v \) yields

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

Note that degrees of regular check nodes in the expanded graphs remain \( d_v \). When presenting complexities of decoding schemes on these graphs, we let \( E_f, L_f \) and \( d_v \) 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**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Comp</th>
<th>Add</th>
<th>Mult</th>
<th>LUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathbb{F}_2 )-LLR-QSPA</td>
<td>( (3d_v - 4) \times 2^r )</td>
<td>( (3d_v - 4) \times 2^r )</td>
<td>0</td>
<td>( (3d_v - 4) \times 2^r )</td>
</tr>
<tr>
<td>( \mathbb{F}_2 )-LLR-QSPA</td>
<td>( (3d_v - 4) \times 2^r )</td>
<td>( (3d_v - 4) \times 2^r )</td>
<td>0</td>
<td>( (3d_v - 4) \times 2^r )</td>
</tr>
<tr>
<td>( \mathbb{F}_2 )-FFT-QSPA</td>
<td>0</td>
<td>( 2d_v \times 2^r )</td>
<td>( (2d_i - 1) \times 2^r )</td>
<td>2 ( 2^m )</td>
</tr>
<tr>
<td>( \mathbb{F}_2 )-FFT-QSPA</td>
<td>0</td>
<td>( 2d_v \times 2^r )</td>
<td>( (2d_i - 1) \times 2^r )</td>
<td>0</td>
</tr>
<tr>
<td>( \mathbb{F}_2 )-Min-Max</td>
<td>( (3d_v - 4) \times 2^r )</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( \mathbb{F}_2 )-Min-Max</td>
<td>( (3d_v - 4) \times 2^r )</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

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 \( 3d_v \times 2^{2r} \) comparisons, additions, and table look-ups, which results in an overall complexity of \( O(2^{2r}) \). However, with the expansion over \( \mathbb{F}_2 \), there are only approximately \( 3d_v \times 2^{2r} \) operations of each type, which reduces overall complexity to \( O(2^{2r}) \). 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 \( O(2^{2r}) \) to \( O(2^{2r}) \) 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 \( 2d_v \times 2^r \)) when using the original graph or an expanded one. There is a slight reduction in the number of additions though, from approximately \( 2d_v \times 2^r \) to \( 2d_v \times 2^r \). Thus, the overall complexity of FFT-QSPA on an expanded graph is \( O(2^{2r}) \), slightly lower than \( O(2^{2r}) \) on the original graph.

**TABLE IV: Variable Node Complexity**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Comp</th>
<th>Add</th>
<th>Mult</th>
<th>LUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathbb{F}_2 )-LLR-QSPA</td>
<td>( 2^t - 1 )</td>
<td>( 2d_v \times 2^r )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( \mathbb{F}_2 )-FFT-QSPA</td>
<td>( (r/m) \times 2^m )</td>
<td>( E_f \times 2^m \times \frac{2^m}{2^m} )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( \mathbb{F}_2 )-FFT-QSPA</td>
<td>( (r/m) \times 2^m )</td>
<td>( E_f \times 2^m \times \frac{2^m}{2^m} )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( \mathbb{F}_2 )-Min-Max</td>
<td>( (d_v + 1) \times 2^t )</td>
<td>( 3d_v \times 2^m )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( \mathbb{F}_2 )-Min-Max</td>
<td>( (d_v + r/m) \times 2^m )</td>
<td>( E_f \times 2^m \times \frac{2^m}{2^m} )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( \mathbb{F}_2 )-Min-Max</td>
<td>( (d_v + r/m) \times 2^m )</td>
<td>( E_f \times 2^m \times \frac{2^m}{2^m} )</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

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 \( d_i = 3 \) 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 \( \mathcal{O}(2^t) \) on the original graph to \( \mathcal{O}(2^{2r}) \) on an expanded one, while in Table III, this change is from \( \mathcal{O}(2^t) \) to \( \mathcal{O}(2^{2r}) \) 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 \( \mathcal{O}(2^t) \) to \( \mathcal{O}(2^{2r}) \). 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 \( C_1 \), a code over \( \mathbb{F}_{26} \) with the codeword length 1998 and code rate 0.89. In this case, the original graph is over \( \mathbb{F}_{26} \), and expansions over \( \mathbb{F}_{26} \) and \( \mathbb{F}_{29} \) 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.
using the $F_{23}$ expansion. This reduces further with the $F_{22}$ expansion, to less than 10%. These gains correspond to speed-ups of more than 5 times in the $F_{23}$ case, and more than 10 times in the $F_{22}$ 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 $F_{23}$ expansion, when compared with using the algorithm on the original $F_{22}$ 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 $F_{22}$ expansion, the number of comparisons reduces further, and the number of additions used is also slightly less than that of the $F_{20}$ case. Since $F_{22}$ expansion has more local check nodes than the $F_{23}$ one, the number of multiplications and table look-ups have increased significantly. Thus, for $C_1$, 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 $O(2^{2r})$ for a code over $F_{2^r}$, and implementing these 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 $F_{2^r}$ into a graph over $F_{2^m}$, 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


[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


