IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 19, 2020, accepted January 17, 2021, date of publication February 22, 2021, date of current version March 3, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3060806

Predictive Syndrome Based Low Complexity Joint Iterative Detection-Decoding Algorithm for Non-Binary LDPC Codes

WAHEED ULLAH, (Senior Member, IEEE), LING CHENG, (Senior Member, IEEE), AND FAMBIRAI TAKAWIRA, (Member, IEEE)
School of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg 2000, South Africa
Corresponding author: Waheed Ullah (waheed@ieee.org)

ABSTRACT This paper addresses the problem of decoding non-binary low density parity check codes(LDPC) over finite field $GF(q)$ using symbol flipping approach. To achieve low complexity reliable communication, three new algorithms for improving the bit error rate performance of the non-binary LDPC decoder are presented. The first type is the symbol flipping decoding algorithm using a flipping function based on the channel reliability to identify the least reliable symbol position. In this algorithm, if the predicted symbol value satisfies the check sum, then the value is declared as correct otherwise the value is adjusted and sent back to the QAM detector. Algorithms 2 in this paper is an improvement to iterative joint detection-decoding algorithm by using the method of iterative hard decision based majority logic to select the new candidate symbol value. The feedback value to the QAM detector is adjusted by using Euclidean distance between the current symbol and the newly selected symbol value. Algorithm 3 is a low complexity version of Algorithm 2 which is derived by applying a majority voting scheme. In the majority voting scheme, symbols are short listed first by voting and all the computation are carried out only for the short listed least reliable symbols which significantly lowers the processing complexity. Numerical results and complexity analysis show that the proposed methods have good bit error rate versus complexity trade-off for various applications when compared with some existing algorithms.

INDEX TERMS Coded modulation, non-binary LDPC, joint iterative detection-decoding, symbol flipping decoding, sum product algorithm, majority logic decoding, reliability based decoding, QAM detector.

I. INTRODUCTION

Non-binary low density parity check (NB-LDPC) codes defined over $GF(q > 2)$ are an extension of binary LDPC codes [1], [2]. NB-LDPC codes have shown better bit error rate (BER) performance than binary LDPC codes. The NB-LDPC codewords can be transmitted as binary sequence. The codeword symbols defined over $GF(q)$ are mapped to a binary sequence and stream of bit vectors is transmitted over a binary input channel. At the receiver, the NB-LDPC decoder performs de-mapping of the received bit vectors to symbols. The main challenge of LDPC codes defined over high order Galois fields $GF(q > 2)$ is the high computational complexity. Fast Fourier transform (FFT) based sum product algorithm (FFT-SPA) [3] for decoding of non-binary LDPC was proposed to reduce the check node complexity in terms of hardware computation.

In the bit reliability [4] and symbol flipping NB-LDPC decoding algorithms [5], [6] the codeword is transmitted as binary sequence and the received binary sequence is then used as reliability information in the decoder. Reliability based majority logic decoding algorithm is similar to message passing decoding but it sends only the most reliable field message and is considered as the low computational algorithm. In the iterative majority logic decoding (MLgD) algorithm for non-binary LDPC codes, each symbol is iteratively updated by the extrinsic information-sums (EXIs) with the most reliable field element along each edge of the Tanner graph. The iterative reliability based hard (IHRB) and soft (ISRB) MLgD algorithms [7] have lower complexity but perform well only for parity check matrices with high column weights. The two algorithms in [7] are further improved by

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

introducing soft reliability information at the initialization to achieve better trade-off between complexity and bit error performance by proposing several modifications [8], [9]. One of the drawback of IHRB and ISRB algorithms in literature, is the requirement of large column weights. To overcome this drawback, an improvement to these algorithms was presented in [10] by introducing reliability updates in terms of bit rather than symbols. The bit reliability based decoding algorithm for NB-LDPC is comparatively more efficient and is termed as weighted bit reliability based (wBRB) algorithm [4]. This bit reliability based algorithm requires integer and Galois field operations only which helps to reduce the hardware implementation complexity and processing time. To further enhance the wBRB decoding algorithms, a full bit-reliability based decoding scheme was proposed in [4]. To lower the processing complexity, this algorithm uses the binary representation of non-zero entries of the parity check matrix. Symbol flipping decoding algorithms have low computational complexity but at the cost of reduced performance. A majority logic decision based algorithm was presented in [11], where the symbol position to be flipped is determined by majority decision while the flipped value is calculated from channel output by flipping individual bits of the symbol with low reliability. Another algorithm termed as weighted algorithm B (wt.Algo B) presented in [12] introduces the binary Hamming distance and plurality logic performance improvement. The parallel symbol flipping decoding (PSFD) algorithm in [13], has good performance only for parity check matrix of large column weight. A multiple voting based PSFD (MV-PSFD) algorithm was proposed in [14] for the improvement of PSFD algorithm. The performance of the SRBMP decoding algorithm [15] has been improved by the multiple voting symbol flipping (MV-SF) decoding algorithm [16].

Non-binary LDPC codes can also be transmitted using direct mapping to the high order modulation. The advantage of NB-LDPC codes using high-order q-ary modulations is the direct encoding and decoding over the q-ary constellation as the binary to non-binary mapping and de-mapping operations are not required. Also the non-binary symbol likelihoods are computed directly without any conversion. The mapping and de-mapping operations are cost effective in terms of complexity and introduce performance degradation that would have to be partially countered by a good choice of mapping and de-mapping at decoder. The NB-LDPC sum-product and its variants can be used to map to the higher order modulation to achieve high data rates. The binary Min-Sum decoding algorithm [17], [18] extended to the non-binary decoding, is known as Extended Min-Sum (EMS) algorithm [19]–[21], and it gives better trade-off between hardware complexity and bit error rate performance. The complexity in the Extended Min-Sum decoding largely arises from the computation of the check node (CN). To reduce the check node processing, a Forward-Backward scheme [22], [23] was proposed where a serial computation is carried in the hardware and intermediate results are used during that serial computation. However this scheme adds high latency and offers less throughput which becomes more significant when the finite field size of $GF(q)$ is increased. To overcome these challenges, a new hardware aware algorithm called syndrome based EMS algorithm was presented in [24]. This algorithm achieves better bit error rate performance as well as throughput due to increased parallelism.

An M-QAM based hard decision NB-LDPC algorithm was proposed in [25], [26] and is called the iterative joint detection-decoding (IJDD) algorithm. This algorithm uses the M-QAM modulation to get the higher throughput but its performance is poor compared to the q-ary LDPC (QSPA). This algorithm uses the hard decision symbol sequence as feedback in each iteration to the M-QAM detector to move the received messages towards correct direction and reduce the noise effect.

To enhance the bit error rate (BER) performance of the IJDD, an Improved IJDD (IIJDD) algorithm has been proposed in [27], [28] and it shows better bit error rate performance. The Improved IJDD algorithm takes into account the probability of symbols at initialization from channel information and then combines with the plurality logic based reliability information during iterative updates. This iterative reliability update of information is termed as accumulated reliability of a symbol. The accumulated reliability of symbols make the IIJDD algorithm computationally very complex. The IJDD algorithms are used with the majority-logic decodable non-binary LDPC codes with high column weight but show performance loss for low rate codes with low column weight [29]. As research in the literature is focused mostly on low column or ultra-low column weight LDPC codes, therefore, we have also chosen to use low/ultra-low column weight. In the literature, those symbol flipping NB-LDPC decoders, showing better BER performances for low column weight, are considered as good decoding algorithms LDPC codes using high column weight increase the decoding computational complexity and contribute towards error floor [7], [13], [25]. Therefore, ultra sparse LDPC codes are used to achieve low decoding latency and to overcome error as shown in [8], [9], [14], [28]. Other advantages of ultra-sparse LDPC codes like high girth and better BER performance are given in the references [19], [29], [34]

In this paper, symbol flipping decoding algorithms are proposed for the QAM based NB-LDPC codes. The proposed algorithms are inspired by the IJDD algorithms [27], [28] and use the concept of iterative hard reliability based majority logic decoding algorithm in [7], [30]. Euclidean distance is used to define the step size of the feedback to the detector to move the symbol towards correct position unlike to [27], [28]. Precise step size of the feedback message plays an important role in improving the BER performance. The IJDD algorithm can show oscillating behaviour if the feedback step size is not controlled properly and results in performance degradation. The IJDD algorithms in literature use the step size as the difference of the first and second maximum numbers of occurrences of a symbol in the extrinsic information-sum as a scaling factor for the symbol feedback information to the

QAM detector. This is not the efficient way of optimizing the received sequence to remove the noise by moving the channel output in the correct direction.

The followings are the main contributions in this paper:

- A new symbol flipping decoding algorithm is presented for QAM modulated NB-LDPC codes. Currently all the symbol flipping algorithms in literature are for BPSK modulation but in this paper, symbol flipping decoding algorithm for higher order modulation is introduced for the first time. Along with syndrome information, the flipping function of Algorithm 1 uses Euclidean distance with plurality logic to identify the positions of the short listed least reliable variable nodes. A majority voting scheme is utilised to short list the unreliable variable nodes. The value of the candidate symbol is selected by prediction from the set of the finite field $GF(q)$.
- To achieve better BER performance, an iterative hard decision based majority logic decoding scheme is introduced in Algorithm 2. Through an iterative approach, the feedback to the QAM detector is improved by the Euclidean distance based reliability of the received sequence to remove noise by moving the message in the correct direction. This differs from the IJDD algorithms which use a difference in number of votes between first and second highest voted candidates in conjunction with column weight to move the symbol in correct direction.
- Algorithm 3 is the low complexity version of the Algorithm 2. The majority voting scheme is applied to short list the least reliable nodes and then the symbol values are computed only for those short listed candidate symbols. This algorithm uses only the first reliability values of the hard decision symbols to update the extrinsic information-sums unlike IIJDD algorithm which uses the first and second reliability values of the hard decision symbols.

The rest of the paper is summarized as follows. Section II describes the relevant literature on non-binary LDPC decoding algorithms. The proposed algorithms on low complexity multiple symbol flipping decoding algorithm are presented in the section III. Section IV presents the results and discussion of the existing and proposed methods, while Section V gives complexity analysis. Section VI concludes the paper.

II. NON-BINARY LDPC CODES

Consider a regular parity check matrix H with dimension $m \times n$ defined over the Galois field (GF) size $q > 2$ where each element $h_{i,j} (1 \leq i \leq m, 1 \leq j \leq n)$ of H is an element over the $GF(q)$ where $q = 2^r$ such that $q = 2^r$ and r shows the number of bits in each symbol. Now consider a NB-LDPC code $\mathcal{C}$ of length n for a regular parity check matrix H, such that each row of H has weight of d_c and each column of H has weight of d_v. In the Tanner graph, for parity check matrix H, the non-zero entries ($h_{i,j} \neq 0$) show the i^{th} check node (CN) connected to the j^{th} variable node (VN) and the Tanner graph edge connection is given by $M(j) = \{i : 1 \leq i \leq m, h_{i,j} \neq 0\}$ and $N(i) = \{j : 1 \leq j \leq n, h_{i,j} \neq 0\}$.

A. NB-LDPC CODES USING BPSK

Let $\boldsymbol{c} = (\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_j, \ldots, \boldsymbol{c}_n)$ be a codeword in $\mathcal{C}$ and the binary form of j^{th} symbol in $\boldsymbol{c}$ is $\boldsymbol{c}_j = (c_{j,1}, c_{j,2}, \ldots, c_{j,t}, \ldots, c_{j,r}), 1 \leq t \leq r$. This binary sequence is modulated with binary phase shift keying (BPSK), where 1 is modulated as +1 and 0 is modulated as -1. The BPSK modulated j^{th} symbol $\boldsymbol{x}_j = (x_{j,1}, x_{j,2}, \ldots, x_{j,t}, \ldots, x_{j,r})$ is sent over additive white Gaussian noise (AWGN) channel. At receiver, the binary sequence $\boldsymbol{y}_j = (y_{j,1}, y_{j,2}, \ldots, y_{j,t}, \ldots, y_{j,n})$ is obtained from the transmitted sequence $\boldsymbol{x}_j$ by adding the additive white Gaussian channel noise $\mathcal{N}(0, \sigma^2)$ with two sided power spectral density as $\boldsymbol{y}_j = \boldsymbol{x}_j + \boldsymbol{n}_j$. The hard decision (HD) binary sequence $\boldsymbol{z}_j = (z_{j,1}, z_{j,2}, \ldots, z_{j,t}, \ldots, z_{j,r})$ for the j^{th} received symbol $\boldsymbol{y}_j$ is given by:

$$z_{j,t} = \begin{cases} 1, & \text{if } y_{j,t} > 0 \\ 0, & \text{if } y_{j,t} \leq 0 \end{cases} \tag{1}$$

The log likelihood ration (LLR) for NB-LDPC codes corresponding to the j^{th} symbol, is computed as follows for the BPSK signalling in AWGN channel [31].

$$\begin{aligned} LLR(\boldsymbol{c}_j) &= ln\left(\frac{Pr(\boldsymbol{y}_j|\boldsymbol{c}_j = \alpha)}{Pr(\boldsymbol{y}_j|\boldsymbol{c}_j = 0)}\right) \\ &= ln\left(\frac{Pr(\boldsymbol{y}_j|(c_{j,1}, c_{j,2}, \ldots, c_{j,t}) = \alpha)}{Pr(\boldsymbol{y}_j|(c_{j,1}, c_{j,2}, \ldots, c_{j,t}) = 0)}\right) \end{aligned} \tag{2}$$

The i^{th} syndrome of the j^{th} hard decision symbol sequence can be defined for the nonzero element of the parity check matrix $\boldsymbol{H} = [h_{i,j}]_{mn}$ with $h_{i,j} \in GF(q)$.

$$\boldsymbol{s}_i = \sum_{j \in N(i)} h_{i,j} \boldsymbol{z}_j \tag{3}$$

If $\boldsymbol{s}_i \neq \boldsymbol{0}$, it means that one or more of the variable nodes contributing to this check node are incorrect.

B. NB-LDPC CODES USING QAM

The codeword $\boldsymbol{c} \in GF(q)$ is mapped to the constellation χ of the quadrature amplitude modulation (QAM) before transmission. Defining $\Omega(\cdot)$ as the symbol mapping function for QAM constellation then the codeword $\boldsymbol{c}$ is mapped as $\boldsymbol{x}_j = \Omega(\boldsymbol{c}_j) \in \chi$ with complex signal vector $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_j, \ldots, \boldsymbol{x}_n)$ and is transmitted over the additive white Gaussian noise (AWGN) channel. The received symbol sequence $\boldsymbol{y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_j, \ldots, \boldsymbol{y}_n)$ is obtained from the transmitted sequence $\boldsymbol{x}$ after adding the complex additive white Gaussian channel noise $n \rightarrow \mathcal{N}(0, \sigma^2)$ with zero mean and two sided power spectral density $N_0/2$ as $\boldsymbol{y}_j = \boldsymbol{x}_j + \boldsymbol{n}_j$.

At the receiver side, the detector estimate each symbol $\hat{\boldsymbol{x}}_j$ using the maximum likelihood decision rule.

$$\hat{\boldsymbol{x}}_j = \arg\min_{\boldsymbol{x} \in \chi} \|\boldsymbol{y}_j - \boldsymbol{x}\| \tag{4}$$

for $j = 1, 2, \ldots\ldots n$.

The above equation (4) becomes very complex for high order QAM. This complexity can be reduced by defining a

radius D of a sphere and the received symbol is estimated for those symbols in the region of D.

$$\hat{\boldsymbol{x}}_j = \arg\min_{\boldsymbol{x}\in\chi} \|\boldsymbol{y}_j - \boldsymbol{x}\| \leq D \tag{5}$$

In this paper we use the QAM constellation size χ equal to the Galois field size q [27]. For the given code rate $r_c = k/n$ where k is the information length, the spectral efficiency for this coded scheme is $\zeta = r_c log_2|\chi|$ bits per symbol.

The i^{th} syndrome of the j^{th} hard decision symbol sequence for $h_{i,j} \neq 0$ can be defined as:

$$\boldsymbol{s}_i = \sum_{j\in N(i)} h_{i,j}\boldsymbol{z}_j \tag{6}$$

If $\boldsymbol{s}_i \neq \boldsymbol{0}$, it means that one or more of the variable nodes contributing to this check node are incorrect.

At the initialization of the non-binary sum-product and min-sum decoding algorithms, the LLR value for each symbol of the received sequences [32] is computed with the assumption that all symbols have equal probability.

$$LLR(c) = ln\left(\frac{Pr(\boldsymbol{y}|\hat{\boldsymbol{x}})}{Pr(\boldsymbol{y}|\boldsymbol{x})}\right) \tag{7}$$

where $\hat{\boldsymbol{x}}$ is the symbol value which maximizes the probability function $Pr(\boldsymbol{y}|\hat{\boldsymbol{x}})$, i.e.

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in GF(q)}{\text{argmax}}\{Pr(\boldsymbol{y}|\boldsymbol{x}\}. \tag{8}$$

When the value of $LLR(\boldsymbol{c}) = 0$ or close to 0, means more reliability and vice versa.

When M-QAM coded modulation is used, the received symbol sequence contains noise which shifts the symbol from the position in the QAM constellation. Therefore, it's hard to detect correctly the symbol value. The Euclidean distance based likelihood method is used to calculate the approximate position of the received symbol using LLR which is given as:

$$LLR(\alpha) = \frac{2}{\sigma^2}(d(\mathbf{y}, \alpha)^2 - d(\mathbf{y}, \hat{\alpha}^2)) \tag{9}$$

where $\hat{\alpha}^2$ is the closest QAM point of the received symbol $\boldsymbol{y}$ and α is the primitive element of $GF(q)$ in the QAM constellation.

C. NON-BINARY LDPC DECODERS

A review of NB-LDPC codes and related decoding algorithms are presented in this section but particular emphasis is on IJDD algorithms [25], [26] using QAM modulation.

For a given NB-LDPC code C over $GF(q)$, the j^{th} received symbol $\mathbf{y}_j$, can be any of the q element in the $GF(q)$. If α is the primitive element of $GF(q)$, then the q elements of the $GF(q)$ are expressed as $GF(q) = \{0, 1, \alpha, \alpha^2, \ldots, \alpha^{q-2}\}$. Therefore, there are q messages passed between the check and variable nodes in the classic belief propagation NB-LDPC decoding algorithm. There are q probable information for each of the channel received symbols. To satisfy the check sum in equation (6) for q symbols, the complexity is in order of $O(q^{d_c-2})$.

The low complexity hard decision symbol flipping decoding based on the majority logic decision is known as generalized algorithm B [12]. This algorithm is based on plurality logic which counts the number of occurrences of each symbol. For the preset threshold t, if the counts of more than one finite field symbol over $GF(q)$ exceed t, choose in favor of that received symbol, otherwise choose some common symbol. This algorithm is further improved by weighting coefficients based on the Hamming distance between finite field symbol and extrinsic information-sums. This improved algorithm is called weighted algorithm B [12]. In this algorithm, the counts of occurrences of a finite field symbol is multiplied by the weighting coefficients assigned to the Hamming distance. The decision is taken based on the largest product. The optimal set of the weights was not determined in the weighted algorithm B and was left as future work. For the given hard decision symbol $\boldsymbol{z}_j$, the normalized extrinsic information-sum (EXI) for the k^{th} iteration is calculated as:

$$\sigma_{i,j}^{(k)} = h_{ij}^{-1} \sum_{j'\in N(i)\setminus j} h_{i,j'}\boldsymbol{z}_{j'}^{(k)} \tag{10}$$

for $1 \leq i \leq m, j \in N(i)$.

The generalized algorithm B (Algo B) decision for estimated correct symbol is obtained using the plurality logic as:

$$\hat{v}_j^{(k)} = \begin{cases} \alpha & if \;\; \eta(\alpha) \geq T \\ \boldsymbol{z}_j^{(k)} & \text{else} \end{cases} \tag{11}$$

Here $\eta(\alpha)$ is the number of the occurrences of $\alpha \in GF(q)$ in the extrinsic information-sum $\sigma_{i,j}^{(k)}$ for α corresponding to the largest count of $\eta(\alpha)$. T is the preset threshold determined through simulations.

In the IJDD algorithm [25], [26], the main idea is to correct the symbol vector $\boldsymbol{y}$ by removing noise in an iterative way and moving the received vector $\boldsymbol{y}$ closer towards the transmitted symbol point.

At the receiver side, the detector estimates each symbol $\hat{\boldsymbol{x}}_j^{(k)}$ using the maximum likelihood decision rule. The hard decision symbol $\boldsymbol{z}_j^{(k)}$ is made available to the NB-LDPC decoder after de-mapping the detector output $\boldsymbol{x}_j$ as follow:

$$\boldsymbol{z}_j^{(k)} = \Omega^{-1}(\hat{\boldsymbol{x}}_j^{(k)}) \tag{12}$$

From equation (11), let η_{max} be the largest reliability and η_{sub} as the second largest reliability of the symbol in $\sigma_{i,j}^{(k)} \in GF(q)$ at the k^{th} iteration, then $\Delta\eta_j^{(k)} = \eta_{max}^{(k)} - \eta_{sub}^{(k)}$ which represents the voting difference between two high reliable candidates. The pair of information $(v_j^{(k)}, \Delta\eta_j^{(k)})$ is provided to the QAM detector as feedback. The detector updates the received sequence $\boldsymbol{y}_j^{(k)}$ with the information provided by the LDPC decoder. Let $d(\boldsymbol{y}_j^{(k)}, r)$ be a search sphere with radius r, centered at $\boldsymbol{y}_j^{(k)}$. The following is the update rule at detector:

- If $\Omega(v_j^{(k)}) \in d(\boldsymbol{y}_j^{(k)}, r)$, then

$$\boldsymbol{y}_j^{(k+1)} = \boldsymbol{y}_j^{(k)} + \lambda^{(k)}\boldsymbol{\beta}^{(k)}, \tag{13}$$

• Otherwise $\boldsymbol{y}^{(k+1)} = \boldsymbol{y}^{(k)}$.

Here $\lambda^{(k)}$ and $\boldsymbol{\beta}^{(k)}$ are calculated as follow:

$$\lambda^{(k)} = \frac{\Delta\eta^{(k)}}{d_v} \tag{14}$$

and

$$\boldsymbol{\beta}^{(k)} = \begin{cases} \hat{\boldsymbol{x}}_j^{(k)} - \boldsymbol{y}_j^{(k)}, & \text{if } \Omega(v^{(k)}) = \hat{\boldsymbol{x}}_j^{(k)} \\ \Omega(v^{(k)}) - \hat{\boldsymbol{x}}_j^{(k)}, & \text{otherwise.} \end{cases} \tag{15}$$

The IJDD algorithm [26] is further improved by combining the plurality logic based scheme with the accumulated reliability [27], [28] from channel. The reliability vector $\boldsymbol{P}_j^{(k)}$ of the j^{th} variable node at k^{th} iteration for each element $\alpha \in GF(q)$ is given by:

$$\boldsymbol{P}_j^{(k)} = [p_j^{(k)}(0), p_j^{(k)}(1), \ldots, p_j^{(k)}(q-1)] \tag{16}$$

where $p_j^{(k)}(\alpha)$ is the probability that the estimated value of $\hat{v}_j^{(k)}$ is equal to the element $\alpha \in GF(q)$. The reliability vector $\boldsymbol{P}_j^{(k)}$ at $k = 1$ is initialized with likelihood values from the channel.

The second reliability metric $\bar{p}^{(k)}(\alpha)$ for the j^{th} variable node is calculated by the following equation:

$$\bar{p}^{(k)}(\alpha) = \frac{\eta_j^{(k)}(\alpha)}{\Phi} \tag{17}$$

where $\eta_j^{(k)}(\alpha)$ is the number of the votes of element α and Φ denotes the sum of the votes obtained by each candidate symbol in $GF(q)$ for the j^{th} variable node. If any of the element α has got zero vote, then it is assigned a value equal to half of the minimum votes of an element in (17) to avoid premature exclusion.

In order to combine the effect of reliability at current iteration k with the reliability at $(k-1)^{th}$,the normalised reliability for an element α of the j^{th} variable node is given by:

$$p^{(k)}(\alpha) = \frac{\bar{p}_j^{(k)}(\alpha) p_j^{(k-1)}(\alpha)}{\sum_{\dot{\alpha} \in GF(q)} \bar{p}_j^{(k)}(\dot{\alpha}) p_j^{(k-1)}(\dot{\alpha})} \tag{18}$$

such that $\sum_{\alpha \in GF(q)} p^{(k)}(\alpha) = 1$. The candidate symbol value $\hat{v}^{(k)_j}$ is selected based on the highest probability and $\delta\eta_j^{(k)}$ is calculated based on the first and second highest reliability values of the two candidates.

The accumulated reliability based IJDD algorithm is further improved by using the first and second hard decision based symbols for each of the j^{th} variable node to improve the bit error rate performance and is termed as Improved IJDD(IIJDD) algorithm [27].

In the improved IJDD algorithm, for each check sum $\boldsymbol{s}_i$ connecting to d_c variable nodes, a variable node, say v_j with least reliability is chosen based on the Euclidean distance. Let $(\boldsymbol{z}_{j,1})$ and $(\boldsymbol{z}_{j,2})$ be the two most reliable symbols passed from the QAM detector to the variable node v_j in two configurations and $\boldsymbol{z}_l \in GF(q)$ be the least reliable symbol, then the two check-to-variable messages from the detector are $\{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{j,1}, \ldots \boldsymbol{z}_{d_c}\} \backslash \{\boldsymbol{z}_l, l \neq j\}$ and $\{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{j,2}, \ldots \boldsymbol{z}_{d_c}\} \backslash \{\boldsymbol{z}_l, l \neq j\}$. For the least variable node v_j, the most reliable symbol $\boldsymbol{z}_{j,1}^{(k)}$ and the next most reliable symbol $\boldsymbol{z}_{j,2}^{(k)}$ are delivered to the associated check node $\boldsymbol{s}_i$ while for other variable node connected to this check node $\boldsymbol{s}_i$, only the most reliable symbol $\boldsymbol{z}_{j,1}^{(k)}$ is delivered.

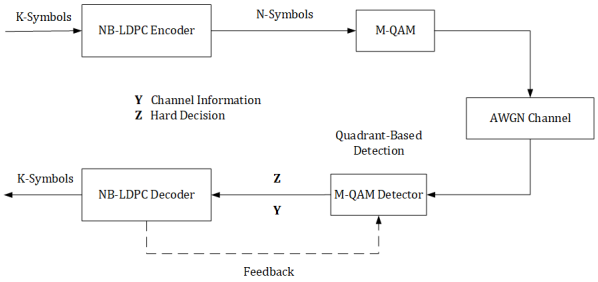

FIGURE 1. System Model: NB-LDPC coded QAM modulation.

III. PROPOSED NB-LDPC DECODING ALGORITHMS

A. SYSTEM MODEL

This paper uses the most common type of QAM modulation which is rectangular QAM, where the constellation points are arranged in a square grid. Figure 1 shows the receiver structure to reduce the complexity of the QAM detector where the received symbol quadrant is first identified and then the symbols in that quadrant are used to estimate the correct symbol.

The non-binary LDPC code defined over the $GF(q)$ is used in connection with the QAM constellation such that the size of the coded symbol is the same as the QAM constellation. The NB-LDPC encoder encodes information symbols to construct a codeword of length n. These n symbols are modulated by the QAM module and are transmitted over wireless channel. The complex AWGN channel $\mathcal{N}(0, \sigma^2)$ with zero mean and two sided power spectral density $N_0/2$, effects each of the transmitted symbol as shown below:

$$\boldsymbol{y}_j = \boldsymbol{x}_j + \boldsymbol{n}_j \tag{19}$$

The quadrant based QAM detector estimates the transmitted signal as follows:

$$\hat{\boldsymbol{x}}_j^{(k)} = \underset{\boldsymbol{x}_{\mathfrak{q}} \in \chi_{\mathfrak{q}}}{\arg\min} \|\boldsymbol{y}_j^{(k)} - \boldsymbol{x}_{\mathfrak{q}}\| \tag{20}$$

where $\chi_{\mathfrak{q}}$ contains all the constellation symbol in the $\mathfrak{q}^{th}$ quadrant. The quadrant $\chi_{\mathfrak{q}}$ is determined from the sign of the real and imaginary parts of the symbol y_j. This quadrant based QAM (QQAM) detection reduces the complexity enormously as only four symbols are required to estimate the correct symbol. The QAM ML detection used in [27] as shown in equation (4) requires all the symbols in constellation which makes the estimation of correct symbols very complex in computation.

This proposed method is very different than the QAM detector presented in [27], [28] where maximum likelihood

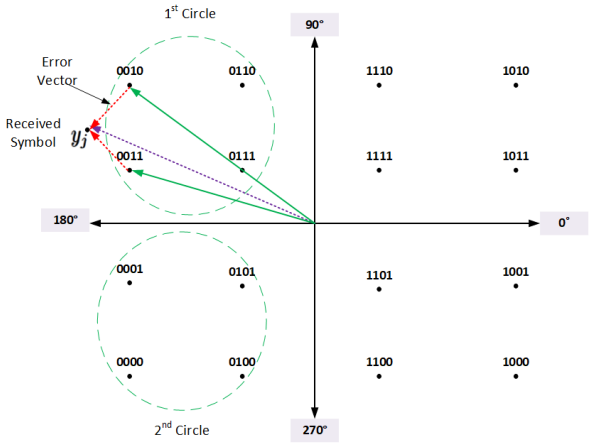

FIGURE 2. 16-QAM Gray coded constellation and the received symbol.

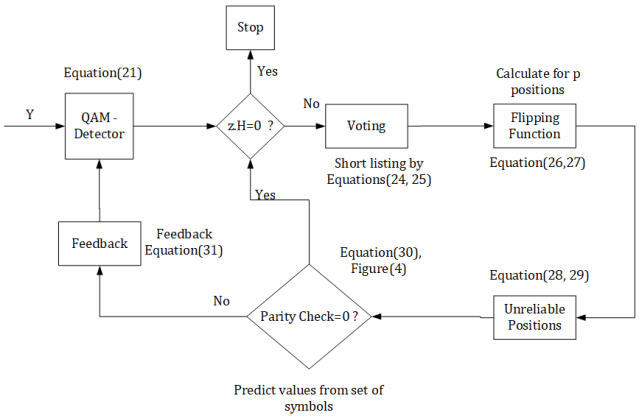

FIGURE 3. Proposed Algorithm 1 Execution Cycle.

detection is used. The received symbol sequence estimated using reduced complexity quadrant based QAM detection is sent to the symbol flipping NB-LDPC decoder. The decoder predicts the unreliable symbol value for each of the failed variable nodes contributing to failure of check-sum $\boldsymbol{s}_i$ at k^{th} iteration. Let the check node $\boldsymbol{s}_i$ have d_c number of connected variable nodes $v_1, v_2, \ldots, v_{d_c}$. For each unreliable node, a set of symbols from the $GF(q)$ table is predicted where each predicted symbol is different in one bit from the unreliable symbols with the assumption that AWGN channel has affected one bit in each symbol at the most. Also keeping in view that, in Gray coded QAM, the symbols in the same quadrant are different by one bit. Therefore, it is more likely to choose the correct symbol for the least reliable symbol of the connected failed check node.

In case the predicted symbol does not satisfy the check-sum, the NB-LDPC decoder updates the information and sends back the newly adjusted symbols to the$M$-QAM detector. The newly adjusted signal may move to another quadrant due to the iterative adjustment and then all the symbols in the new quadrant are used to estimate the newly adjusted symbol.

B. PROPOSED ALGORITHM 1

In this section, a new symbol flipping decoding algorithm for QAM modulated NB-LDPC codes is presented. Reliability information are added to the flipping function to decide the accurate symbol position in error as shown in Figure 3. For this selected position, a new candidate symbol value is predicted for the least reliable variable node connected to the failed check. In this proposed algorithm, only those variable nodes contributing to the failed checks are considered for flipping in order to maintain low complexity and fast convergence. The voting scheme adopted in parallel symbol flipping [13], [14] is used for selection of symbol positions to be flipped at the k^{th} iteration. In Figure 3, the short listing of least reliable symbols is completed first and the p number of variable nodes are sent to the next module for computing the flipping function. The flipping function decide the number of variable nodes to be flipped whereas the parity check module selects the flipping value for the positions selected by flipping function. The output of the QAM detector is fed into the check sum. The variable nodes contributing to the successful check sum are given as:

$$\tilde{\boldsymbol{\upsilon}} = \{i' : \boldsymbol{s}_i = \mathbf{0}, 1 \leq i \leq m, j \in M(i')\} \tag{21}$$

Those variable nodes contributing to the failed checks belong to ψ which is the vector containing the number of positions to be flipped, then it must be the subset of τ and is defined as:

$$\boldsymbol{\psi} \in \boldsymbol{\tau} = \{\boldsymbol{\upsilon}(j) \backslash \boldsymbol{\upsilon} \cap \tilde{\boldsymbol{\upsilon}}\}. \tag{22}$$

Here $\boldsymbol{\tau}$ is a set of erroneous variable nodes and equation (22) contains all variable nodes contributing to failed checks. If a single symbol is flipped in each iteration, then $\boldsymbol{\psi}$=1.

1) VARIABLE NODE SELECTION BY VOTING

In the voting scheme [13], [14], each unsatisfied check node gives one vote to the relevant variable node. The j^{th} variable node then collects all the votes, say $V_j^{(k)}$, from the failed check nodes at k^{th} iterations.

$$V_j^{(k)} = \sum_{i \in M(j)} V_{i,j}^{(k)} \tag{23}$$

where $V_{i,j}^{(k)} = 1$ if $\boldsymbol{s}_i^{(k)} \neq \mathbf{0}$, otherwise $V_{i,j}^{(k)} = 0$. For the accumulated voting $V_j^{(k)}$ equal or greater than a predefined threshold V_{th}, those variable nodes fulfilling the condition $V_j \geq V_{th}$ will be passed to calculate the flipping function $E_j^{(k)}$ through equation (25). This reduces the computational complexity from n number of variable nodes to just few variable nodes, say $\boldsymbol{\delta}$. The values of $\boldsymbol{\delta}$ at k^{th}can be calculated simply as follows:

$$\delta_{j'}^{(k)} = V_j^{(k)}, \quad \text{if } V_j \geq V_{th} \tag{24}$$

for $1 \leq j' \leq p$.

Here p shows the total number of variable nodes having maximum votes and $\boldsymbol{\delta}^{(k)}$ contains all positions for those variable nodes (VNs). In other words, these are the number of variable nodes having less reliable information and need to be

replaced with more reliable symbols. The positions of these variable nodes are stored as $\delta_{j'}^{(k)}$. The multiple voting method presented in [14] define two voting levels $\zeta_0 > \zeta_1 > 0$, using the same voting function defined in (24). For $\boldsymbol{s}_j^{(k)} \neq 0$, $V_{ij}^{(k)} = \zeta_0$ for the largest flipping function and $V_{ij}^{(k)} = \zeta_1$ for the VN with the second largest flipping function. In this paper, a voting scheme is used for short listing of least reliable symbols which reduces the memory consumption and computational complexity.

2) QAM BASED FLIPPING FUNCTION

Each unsatisfied syndrome(check-node) $\boldsymbol{s}_i^{(k)}$ provides one vote to the variable node and the variable node accumulates the votes. To find the position of variable nodes with less reliability, the following expression is used for $1 \leq j' \leq p$.

$$F_{j'}^{(k)} = \sum_{i \in M(j')} (2\boldsymbol{s}_i^{(k)} - 1)\xi_{j'}^{(k)} - \eta_{j'}^{(k)}(\alpha). \tag{25}$$

where $\eta_{j'}^{(k)}(\alpha)$ is number of occurrence of $\alpha \in GF(q)$ in the extrinsic information-sum $\sigma_{i,j}^{(k)}$ and $\xi_{j'}^{(k)}$ contains the reliability information derived from the channel received sequence by following mathematical expression.

$$\xi_{j'}^{(k)} = ||\boldsymbol{y}_{j'}^{(k)} - \hat{\boldsymbol{x}}_{j'}^{(k)}||. \tag{26}$$

The flipping function in equation (25) is derived from BPSK based symbol flipping decoding algorithm in [33]. The proposed algorithm finds iteratively the valid codeword from vector space over $GF(q)$. At the k^{th} iteration, if $\boldsymbol{s}_i^{(k)} = \boldsymbol{y}_j^{(k)} h_{i,j} \neq 0$, the aim is to perturb $\boldsymbol{y}^{(k)}$ and create a new sequence of candidate symbols $\boldsymbol{y}^{(k+1)}$. Two steps are required to choose a new candidate symbol value: 1) compute the flipping function to identify the flipped position; 2) candidate symbol value selection by predictive syndrome method as shown in Figure 4.

The position to be flipped is determined for the variable nodes having less reliability information by the following equation.

$$j'^{*(k)} = \underset{j'}{\operatorname{argmax}}(F^{(k)}). \tag{27}$$

Importantly, $j'^{*(k)}$ will belong to d_v number of check nodes. Therefore, to lower the complexity, d_v should be as small as possible. Let $\boldsymbol{\tau}$ be the threshold to determine the number of symbols to be flipped, then it can be determined in conjunction wih the column weight d_v. If ρ is the maximum number of symbols to be flipped per iteration then it can be determined as:

$$\rho = \begin{cases} \rho_1 & \text{if } \boldsymbol{\tau} \geq \varepsilon_1 d_v \\ \rho_2 & \text{else,} \end{cases} \tag{28}$$

where ε_1 is an integer value and $\rho_1 > \rho_2$. To find the position of the candidate symbol value, the equation for syndrome based symbol value prediction is written as:

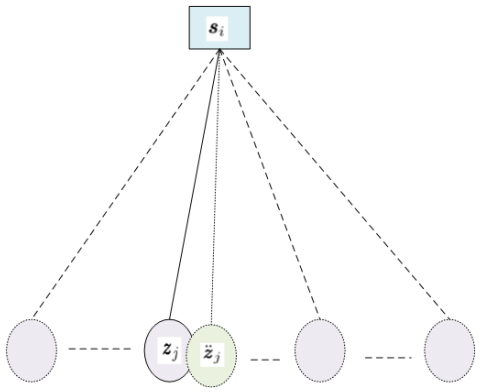

FIGURE 4. Symbol value prediction.

$$\boldsymbol{s}_i^{(k)} = \sum_{j'=1}^{d_c} h_{i,j'} \boldsymbol{z}_{j'}^{(k)} = h_{i,1}\boldsymbol{z}_1^{(k)} + h_{i,2}\boldsymbol{z}_2^{(k)} + \ldots + h_{i,d_c}\boldsymbol{z}_{d_c}^{(k)}. \tag{29}$$

In equation (29), let $\boldsymbol{z}_2^{(k)}$ be in error and contributes to the failure of the check node. To choose a correct candidate symbol, the symbol value is predicted from the set of the symbols which are different in one bit. If $\boldsymbol{s}_i^{(k)}$ is satisfied, then prediction process is terminated otherwise we choose another check node connected to $\boldsymbol{z}_{j'}^{(k)}$ until d_v number of check nodes. If all these check nodes fail, then we choose the last predicted value of the symbol and adjust $\boldsymbol{z}_2^{(k)}$ as shown in the following section.

3) SYMBOL VALUE SELECTION

To replace an unreliable symbol, the candidate symbol value selection is important. To choose a correct symbol value, a symbol prediction method is used from set of values in $GF(q)$. A symbol in the same quadrant as the received symbol is selected as correct symbol which satisfies the parity check equation $\boldsymbol{s}_i^{(k)} = \boldsymbol{0}$.

Let $\boldsymbol{z}_{j'}^{(k)}$ be the variable node symbol considered to be the least reliable symbol connected to the failed check node $\boldsymbol{s}_i^{(k)}$. To satisfy this parity check equation, a symbol is predicted for the selected position of hard decision symbol $\boldsymbol{z}_{j'}^{(k)}$ such that $\boldsymbol{z}_{j'}^{(k)} \neq \ddot{\boldsymbol{z}}_{j'}^{(k)}$ and $\ddot{\boldsymbol{z}}_{j'}^{(k)}$ has all possible symbol values of the $GF(q)$ inside the qudrant. Therefore, the symbol values from the quadrant for each position of $\ddot{\boldsymbol{z}}_{j'}^{(k)}$ which can be written as:

$$\ddot{\Gamma}_{j'}^{(k)} = \{\ddot{\boldsymbol{z}}_{j'}^{(k)} : \ddot{\boldsymbol{z}}_{j'}^{(k)} \in GF(q), \ddot{\boldsymbol{z}}_{j'}^{(k)} \neq \boldsymbol{z}_{j'}^{(k)}\} \tag{30}$$

The candidate symbol value prediction in equation (29) is illustrated in Figure 4. If $\boldsymbol{z}_{j'}$ is the least reliable symbol connected to check node $\boldsymbol{s}_i$ at k^{th} iteration, then symbol $\ddot{z}_{j'}$ is the predicted value such as $\boldsymbol{z}_{j'} \neq \ddot{\boldsymbol{z}}_{j'}$ and $\ddot{\boldsymbol{z}}_{j'} \in \ddot{\Gamma}_{j'}$ which satisfies the parity check equation i.e $\boldsymbol{s}_i = \boldsymbol{0}$.

4) FEEDBACK TO DETECTOR

In this proposed algorithm, if the predicted value does not satisfy the check equation, then the value is adjusted and sent

back to the QAM detector as follows:

$$\boldsymbol{y}_{j'}^{(k+1)} = \boldsymbol{y}_{j'}^{(k)} + f(\lambda^{(k)})\boldsymbol{\beta}^{(k)}. \tag{31}$$

Here f is a scaling factor and is determined through simulations. The other parameters $\lambda^{(k)}$ and $\boldsymbol{\beta}^{(k)}$ are calculated in the following equations. $\lambda^{(k)}$ is the $l2$-norm used to scale the second term of the equation (31).

For the selected candidate symbol value $v^{(k)}$, then $\lambda^{(k)}$ and $\boldsymbol{\beta}^{(k)}$ are calculated as follows:

$$\boldsymbol{\beta}^{(k)} = \begin{cases} \hat{\boldsymbol{x}}_{j'}^{(k)} - \boldsymbol{y}_{j'}^{(k)}, & \text{if } \Omega(v^{(k)}) = \hat{\boldsymbol{x}}_{j'}^{(k)} \\ \Omega(v^{(k)}) - \hat{\boldsymbol{x}}_{j'}^{(k)}, & \text{otherwise} \end{cases} \tag{32}$$

$$\lambda^{(k)} = \begin{cases} ||\hat{\boldsymbol{x}}_{j'}^{(k)} - \boldsymbol{y}_{j'}^{(k)}||, & \text{if } \Omega(v^{(k)}) = \hat{\boldsymbol{x}}_{j'}^{(k)} \\ ||\Omega(v^{(k)}) - \hat{\boldsymbol{x}}_{j'}^{(k)}||, & \text{otherwise} \end{cases} \tag{33}$$

Equations (32) and (33) give the step size to move the received symbol towards the correct position by removing the noise in a systematic way. The difference between the received symbol $\boldsymbol{y}_{j'}^{(k)}$ and the selected candidate symbol value determines the step size more accurately. The computed value of $\lambda^{(k)}$ can be large and needs to be truncated, otherwise this can lead to large step size and as a result the symbol will be moved to a far off location in the QAM constellation. The step size truncation is especially important in higher order QAM modulation as the number of symbols get congested and the distance between them decrease making it hard to estimate the correct symbol.

Continuous Loop Detection: A continuous loop detection procedure is adopted in this paper and positions of all the flipped symbols are recorded. The symbol flipped in an iteration is not flipped again in the successive iterations. In case the new candidate symbol does not satisfy the check equation, then it is re-adjusted and sent to the QAM detector. This loop detection procedure is applied to algorithm 1 only while the proposed algorithms 2 and 3 do not use any loop detection. For example, if a test flag (TF) is defined to store position of all flipping symbols, say $TF = [2, 4, 19]$ at k^{th} iteration and in next iterations these symbols are not flipped until the TF equal to total number of those variable nodes selected by the voting and then the TF is reset to zero for storing the symbol positions from next iteration.

C. PROPOSED ALGORITHM 2

In this proposed scheme, iterative hard reliability based majority logic decoding algorithm in [7], [30] is used which offers very low complexity at the initialization as well as to compute iteratively the hard reliability of the symbols as shown in Figure 5. Secondly Euclidean distance based feedback is used to control the step size of the symbol in the QAM constellation and move the symbol towards correct position by removing noise. Precise step size of the feedback scheme play an important role in improving the BER performance. The IJDD algorithm can show oscillating behaviour if the feedback step size is not controlled properly which can results in performance degradation.

Proposed Algorithm 1

1. Initialization: For $1 \leq j \leq n$, $1 \leq l \leq q$ and $1 \leq i \leq m$. The hard decision symbols $\boldsymbol{z_j}$ is obtained from the received symbol $\boldsymbol{y_j}$ after de-mapping by using (20) and (12).
2. Set max iteration $k = 1$ to I_{max}.
3. Syndrome Check-Sum: if $\boldsymbol{s}_i^{(k)} = \sum_{j \in N(i)} h_{i,j}\boldsymbol{z}_j^{(k)} = \boldsymbol{0}$ or if $k = I_{max}$, stop decoding and output $\boldsymbol{z}^{(k)}$ as codeword.
4. Determine the variable nodes contributing to failed check-sum by (21) and (22).
5. Calculate the votes for each of the variable node by using (23) and short list the least reliable symbol positions by (24).
6. For $1 \leq j' \leq p$, compute the flipping function for the selected number of positions by using (26) and (29).
7. Predict the symbol values $\ddot{\boldsymbol{z}}_{j'}^{(k)}$ by (30) for the chosen number of the symbol positions. If $\boldsymbol{s}_i^{(k)} = \boldsymbol{0}$, update the symbol $\boldsymbol{z}_{j'}^{(k)}$ and go to step 2.
8. If $\boldsymbol{s}_i^{(k)} \neq \boldsymbol{0}$, adjust the symbol by using (31).
9. The QAM detector updates the hard decision symbol $\boldsymbol{z}_{j'}^{(k)}$ based on the feedback.
10. $k = k + 1$ and go to step 3.

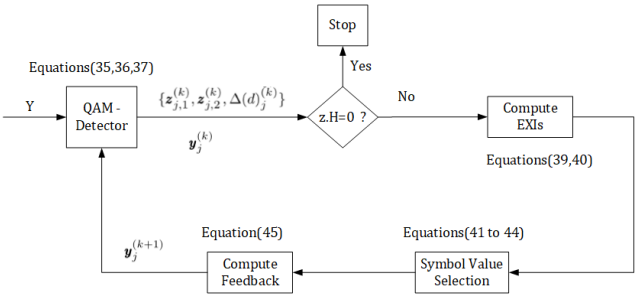

FIGURE 5. Proposed Algorithm 2 execution cycle.

In the proposed Algorithm 2, for each check sum $\boldsymbol{s}_i$ connecting to d_c variable nodes, a variable node, say v_j with least reliability is chosen based on the Euclidean distance same as in [27]. Let $\boldsymbol{z}_{j,1}$ and $\boldsymbol{z}_{j,2}$ be in the quadrant $\mathfrak{q}$ and are assumed as the two most reliable symbols passed from the QAM detector to the variable node v_j in two configurations. Suppose $\boldsymbol{z}_l \in GF(q)$ is the least reliable symbol, then the two check-to-variable messages from the detector are $\{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{j,1}, \ldots \boldsymbol{z}_{d_c}\} \backslash \{\boldsymbol{z}_l, l \neq j\}$ and $\{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{j,2}, \ldots \boldsymbol{z}_{d_c}\} \backslash \{\boldsymbol{z}_l, l \neq j\}$. For each least reliable variable node among the nodes connected by the same check node, the detector generates two most reliable symbols as follows:

$$\hat{\boldsymbol{x}}_{j,1}^{(k)} = \underset{\boldsymbol{x}_\mathfrak{q} \in \chi_\mathfrak{q}}{\arg\min} \|\boldsymbol{y}_j^{(k)} - \boldsymbol{x}_\mathfrak{q}\| \tag{34}$$

and the next reliable symbol is generated by:

$$\hat{\boldsymbol{x}}_{j,2}^{(k)} = \underset{\boldsymbol{x}_\mathfrak{q} \in \chi_\mathfrak{q}, \boldsymbol{x}_\mathfrak{q} \neq \hat{\boldsymbol{x}}_{j,1}^{(k)}}{\arg\min} \|\boldsymbol{y}_j^{(k)} - \boldsymbol{x}_\mathfrak{q}\| \tag{35}$$

Let $d_{j,1}^{(k)} = \|\boldsymbol{y}_j^{(k)} - \boldsymbol{x}_{\mathbb{q}}\|$ and $d_{j,2}^{k} = \|\boldsymbol{y}_j^{(k)} - \boldsymbol{x}_{\mathbb{q}}\|$ be the radiabilities for the two symbols $\hat{\boldsymbol{x}}_{j,1}^{(k)}$ and $\hat{\boldsymbol{x}}_{j,2}^{(k)}$ and the reliability metric for each variable node v_j is expressed as:

$$\Delta d_j^{(k)} = |d_{j,1}^{(k)} - d_{j,2}^{(k)}| \tag{36}$$

$\Delta d_j^{(k)}$ represent the reliability of the symbol for corresponding VN. Smaller value of the $\Delta d_j^{(k)}$ shows less reliability of the variable node and vice versa. After de-mapping, $\boldsymbol{z}_{j,1}^{k} = \Omega^{-1}\boldsymbol{x}_{j,1}^{(k)}$ and $\boldsymbol{z}_{j,2}^{(k)} = \Omega^{-1}\boldsymbol{x}_{j,2}^{(k)}$, the pair of information $\{\boldsymbol{z}_{j,1}^{(k)}, \boldsymbol{z}_{j,2}^{(k)}, \Delta(d)_j^{(k)}\}$ are sent to the NB-LDPC decoder at $k+1^{th}$ iteration. The position for least reliable variable node v_j among all others VNs connecting to the same check node s_i, is selected as:

$$j = \arg\min_{j \in N(i)} \Delta d_l, \quad l = 1, 2, \ldots, d_c \tag{37}$$

For this variable node v_j, most reliable symbol $\boldsymbol{z}_{j,1}^{(k)}$ and the next most reliable symbol $\boldsymbol{z}_{j,2}^{(k)}$ are delivered to the associated check node $\boldsymbol{s}_i$ while for other variable node connected to this check node $\boldsymbol{s}_i$, only most reliable symbol $\boldsymbol{z}_{j,1}^{(k)}$ is delivered. As there are two sets of message sequence from detector to NB-LDPC decoder, the extrinsic information-sum(EXI) in (10) results in the following configurations:

$$\sigma_{il}^{(k)} = \begin{cases} h_{ij}^{-1} \sum_{j' \in N(i)\setminus j} h_{ij'} \boldsymbol{z}_{j',1}^{(k)}, & \text{if } l = j \\ h_{il}^{-1} \sum_{l' \in N(i)\setminus\{j,l\}} h_{il'} \boldsymbol{z}_{l',1}^{(k)} + h_{i,j} \boldsymbol{z}_{j,2}^{(k)}, & \text{otherwise.} \end{cases} \tag{38}$$

and

$$\tilde{\sigma}_{il}^{(k)} = \begin{cases} h_{ij}^{-1} \sum_{j' \in N(i)\setminus j} h_{i,j'} \boldsymbol{z}_{j',1}^{(k)}, & \text{if } l = j \\ h_{il}^{-1} \sum_{l' \in N(i)\setminus\{jl\}} h_{il'} \boldsymbol{z}_{l',1}^{(k)} + h_{ij} \boldsymbol{z}_{j,2}^{(k)}, & \text{otherwise.} \end{cases} \tag{39}$$

Here (38) shows the extrinsic information-sums of the most reliable hard decision messages and (39) denotes the configuration for next most reliable hard decision based messages.

The concept of iterative hard reliability [7], [30] is used to find the correct symbol for all those positions having less reliable information. Suppose $\boldsymbol{z}_j$, being the most reliable symbol, then it is used for the initialization of the reliability of a symbol $\alpha_l \in GF(q)$ to a value γ for $1 \le l \le q$:

$$R_{j,l} = \begin{cases} \gamma, & \text{if } \boldsymbol{z}_{j,l} = \alpha_j, \\ 0, & \text{otherwise.} \end{cases} \tag{40}$$

Here γ is the predefined numerical value. Reliability information at the current iteration is used to find the most reliable symbol for the j^{th} position through the following expression:

If $\sigma_{i,j}^{(k)} = \alpha_l \in GF(q)$

$$R_{j,l}^{(k)} = R_{j,l}^{(k)} + 1 \tag{41}$$

and the reliability information are updated for the next $(k+1)^{th}$ iteration as follows:

$$R_{j,l}^{(k+1)} = R_{j,l}^{(k)}. \tag{42}$$

Proposed Algorithm 2 (E-IJDD)

1. Initialization: For $1 \le j \le n$, $1 \le l \le q$ and $1 \le i \le m$. Signal Detection: The hard decision symbols $\boldsymbol{z_j}$ is obtained from the received symbol $\boldsymbol{y_j}$ by (20) and (12). Also find $\hat{\boldsymbol{x}}_{j,1}^{(k)}, \hat{\boldsymbol{x}}_{j,2}^{(k)}$, and $\Delta d_j^{(k)}$ by using equations (34),(35) and (36). Decoder: If $\boldsymbol{z}_j = \alpha_l \in GF(q)$, then $R_{j,l} = \gamma$ else $R_{j,l} = 0$.
2. Set max iteration $k = 1$ to I_{max}.
3. Syndrome Check-Sum: if $\boldsymbol{s}_i^{(k)} = \sum_{j \in N(i)} h_{ij} \boldsymbol{z}_j^{(k)} = \boldsymbol{0}$ or if $k = I_{max}$, stop decoding and output $\boldsymbol{z}^{(k)}$ as codeword.
4. Calculate extrinsic information-sums $\sigma_{ij}^{(k)}$ by (38) and (39).
5. Calculate symbol reliability $R_{j,l}^{(k)}$ by (41) and update $R_{j,l}^{(k+1)}$ by (42).
6. Select the new candidate symbol value $\boldsymbol{z}_j^{(k+1)}$ by (43).
7. Adjust the symbol by using (44) and feedback to detector. Repeat equations (20),(12) (34),(35) and (36).
8. $k = k + 1$ for next iteration.

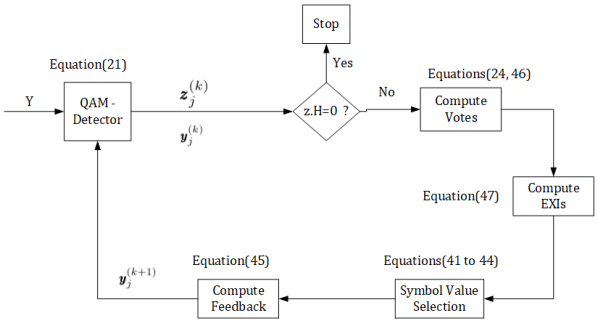

FIGURE 6. Proposed Algorithm 3 Execution Cycle.

Based on these reliability information, the new candidate symbol is selected as follow:

$$\boldsymbol{z}_j^{(k+1)} = \underset{\alpha_l}{\text{argmax}}\, R_{j,l}^{(k)}. \tag{43}$$

In the Algorithm 2, the update rule at detector is given by:

$$\boldsymbol{y}_j^{(k+1)} = \boldsymbol{y}_j^{(k)} + f(\lambda^{(k)})\boldsymbol{\beta}^{(k)}, \tag{44}$$

Here f is the scaling factor and $\lambda^{(k)}$ and $\boldsymbol{\beta}^{(k)}$ are calculated as given in equations (32) and (33) respectively.

D. PROPOSED ALGORITHM 3

This proposed Algorithm 3 further simplifies Algorithm 2 by introducing the concept of short listing of the candidate symbols by voting scheme as shown in Figure 6. In this proposed algorithm, the variable nodes contributing to the failure of the check-sum are first short listed and then only those short listed variable nodes are processed to estimate the their correct values. This process results in very less computational complexity as well as require less memory to store the processed data.

At the decoder, for the hard decision symbol sequence $\boldsymbol{z}^{(k)}$, the candidate symbols are short listed. Suppose $\boldsymbol{\delta}$ stores the

Proposed Algorithm 3 (V-IJDD)

1. Initialization: For $1 \leq j \leq n$, $1 \leq l \leq q$ and $1 \leq i \leq m$. The hard decision symbols $\boldsymbol{z_j}$ is obtained from the received symbol $\boldsymbol{y_j}$ after de-mapping by using (20) and (12).
if $\boldsymbol{z}_j = \alpha_l \in GF(q)$, then $R_{j,l} = \gamma$ else $R_{j,l} = 0$.
2. Set max iteration $k = 1$ to I_{max}.
3. Syndrome Check-Sum: if $\boldsymbol{s}_i^{(k)} = \sum_{j \in N(i)} h_{ij} \boldsymbol{z}_j^{(k)} = \boldsymbol{0}$ or if $k = I_{max}$, stop decoding and output $\boldsymbol{z}^{(k)}$ as codeword.
4. Calculate the votes for each of the variable node by using (23) and (45). Now short list the least reliable symbol positions by (24).
5. For $1 \leq j' \leq p$, calculate extrinsic information-sum $\sigma_{i,j'}^{(k)}$ by (46).
6. Calculate symbol reliability $R_{j',l}^{(k)}$ by (41) and update $R_{j',l}^{(k+1)}$ by (42).
7. Select the new candidate symbol value $v_{j'}^{(k+1)}$ by (43).
8. Adjust the symbol by using (44) and feedback to detector as in (20) and (12).
9. $k = k + 1$ for next iteration.

positions of all the variable nodes with maximum voting, then the values of $\boldsymbol{\delta}$ at k^{th} iteration can be calculated as follows:

$$\boldsymbol{\delta}_{j'}^{(k)} = V_j^{(k)}, \quad if \ V_j \geq V_{th} \tag{45}$$

for $1 \leq j' \leq p$ and p shows the total number of variable nodes with maximum votes. This reduces the computational complexity from n number of variable nodes to just few variable nodes p. Here $\delta_{j'}^{(k)} \in \boldsymbol{\delta}^{(k)}$ stores the position of each variable node with maximum votes and $\boldsymbol{\delta}^{(k)}$ contains all those positions of the variable nodes. In other words, $\boldsymbol{\delta}^{(k)}$ have all the positions of variable nodes having less reliable information and must be replaced with reliable symbols.

For the short listed candidate symbols $\boldsymbol{\delta}^{(k)}$, the normalized extrinsic information-sum (EXI) for the k^{th} iteration is calculated as:

$$\sigma_{i,j'}^{(k)} = h_{ij}^{-1} \sum_{\bar{j} \in N(i) \setminus j'} h_{i,\bar{j}} \boldsymbol{z}_{\bar{j}}^{(k)} \tag{46}$$

for $1 \leq i \leq m$, $1 \leq j' \leq p$. The extrinsic information-sum computation in the IIJDD algorithm [27] takes into account the first and the second most reliable symbols $\{\boldsymbol{z}_{j,1}^{(k)}, \boldsymbol{z}_{j,2}^{(k)}\}$ while the proposed algorithm 3 does not require the second reliability value. The symbols value is selected based on the IHRB algorithm as mentioned in algorithm 2. The feedback to the QAM detector is used as discussed in the algorithm 1 but only for the short listed variable nodes $\boldsymbol{\delta}$ at the k^{th} iteration.

IV. NUMERICAL RESULTS AND ANALYSIS

The proposed decoding algorithms for the NB-LDPC code (n, d_v, d_c) with d_v and d_c as the column and row weights respectively, has been simulated using QAM as the modulation technique and the performance parameters are compared with IJDD decoding algorithms in literature. The parity check

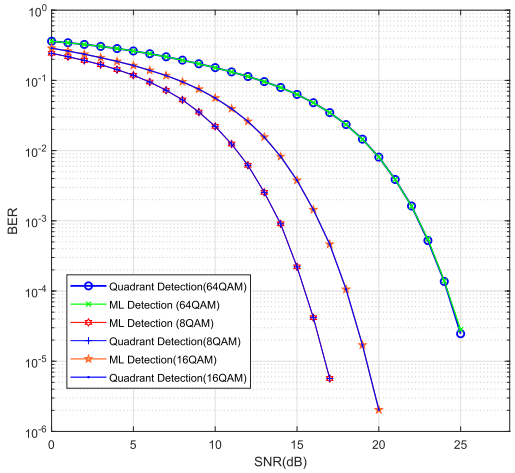

FIGURE 7. Comparison of ML and Quadrant QAM Detection.

matrices with $d_v = 2$ in this paper, is based on the construction method given in [29] and are available on the web site [34]. The ultra-spars parity check matrices are designed for multiple possible applications and more specifically for building good NB-LDPC codes in higher order Galois field. The maximum number of iteration in all the examples are kept as Imax=15 and the value of $\gamma = 1.75$. In the following examples, it is observed clearly from the BER performance curves that the proposed algorithm 2 performs better than the IJDD algorithms in litrature. It should be noted that in all these examples, the QAM modulation order is the same as Galois field size.

Example 1: In this example, BER performance of quadrant based detection scheme is compared with the ML QAM detection. The simulation results are shown for 8,16 and 64 QAM. In the Figure 7, it is clear that the performance curves of the proposed system model are similar and almost perfectly overlap on the ML QAM detection. For higher order QAM modulator like 128 or 256, the BER performance curves might not overlap perfectly as the symbols are close enough to be detected exactly in one quadrant of the constellation.

Example 2: The construction of NB-LDPC code (45, 2, 3) in this example over $GF(8)$ is based on the alist format given in [34]. In this example 8 QAM modulation has been used to simulate the proposed algorithms in comparison to the IJDD and Improved IJDD NB-LDPC decoding algorithms in literature [27]. The performance curves of various algorithms under AWGN are as shown in Figure 8. At BER of 10^{-6}, the algorithm 2 is outperforming about 3.5dB from Improved IJDD. The proposed algorithm 3 is performing about 4.2dB less than Improved IJDD while outperforming than algorithm 1 as well as IJDD. The proposed algorithm 1 based on the symbol flipping decoding, is performing around 2dB better than IJDD algorithm. The step size of the Algorithm 2 is based on the Euclidean distance between the received symbol and the newly selected candidate symbol which help in moving the received symbol in correct direction. Also the

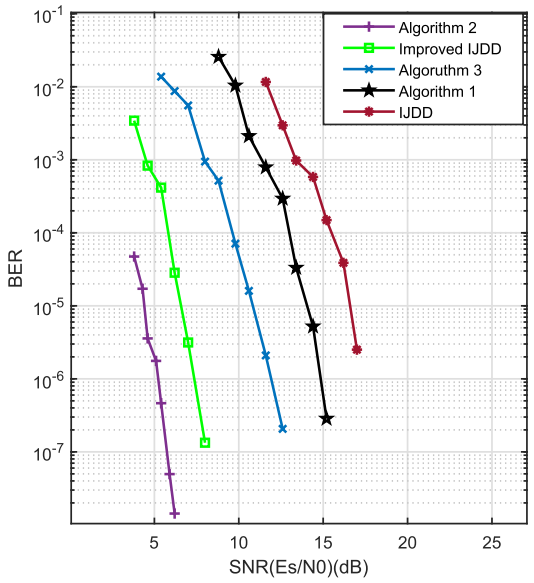

FIGURE 8. BER Performance of NB-LDPC Code (45,2,3) over GF(8).

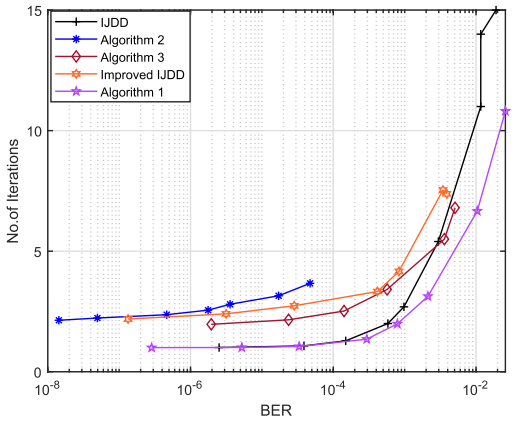

FIGURE 9. Average number of iterations verus BER for the (45,2,3) code over $GF(8)$.

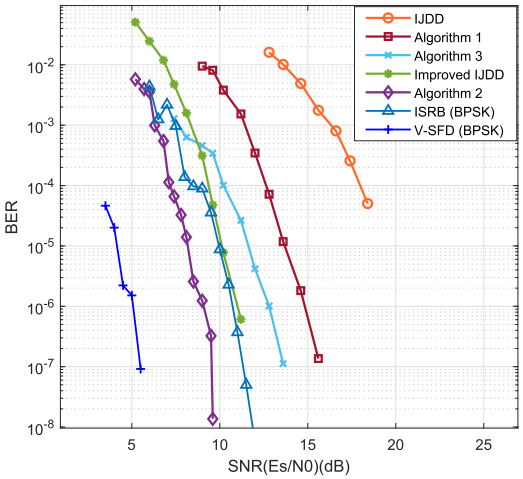

FIGURE 10. BER Performance of NB-LDPC Code (45,2,3) over $GF(16)$.

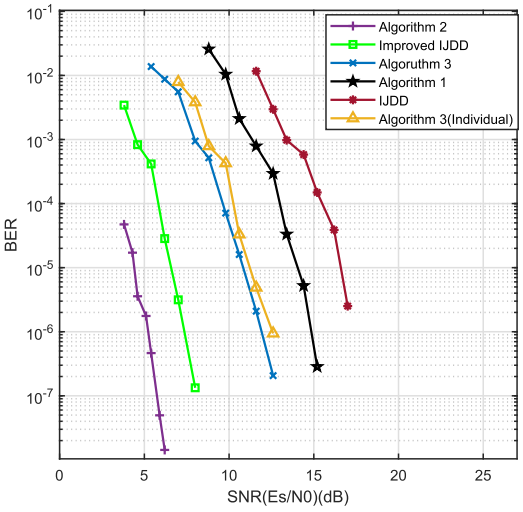

FIGURE 11. BER Performance of NB-LDPC Code(105,2,3) over GF(8).

adaptation of the IHRB algorithm to select the candidate symbol value helps to improve the BER performance of Algorithm 2.

In this example, Figure 9 depicts the curves for BER and average number of iterations of various algorithms. It is visible from the graph that the algorithm 1 converges faster than the other algorithms. Around the BER 10^{-4}, the average number of iterations of the IJDD algorithm and algorithm 1 are equal while the proposed algorithm 3 and the Improved IJDD algorithm are almost equal with average number of iteration less than 3 while algorithm 2 has slightly more than 3 iteration.

Example 3: In this example, the LDPC code(45,2,3) over $GF(16)$ is used to illustrate the BER performance of the proposed decoding algorithms in comparison to Improved IJDD and IJDD algorithms. From the BER curves illustrated in Figure 10, it is observed that the proposed algorithm 1 and 3 perform better than IJDD algorithm and close to the Improved IJDD algorithm. The performance gap at BER 10^{-6} between the proposed algorithm 2 is about 2.5dB while the algorithm 3 is around 2.5 less in performance than Improved IJDD and 2.5dB better than algorithm 1. Algorithm 1 shows an excellent performance in comparison to IJDD algorithm. It is worth mentioning that the algorithms in [6], [7] are using BPSK modulation while those presented in this paper are using QAM. The NB-LDPC code (45,2,3) over $GF(16)$ has been simulated using BPSK modulation to get the BER performance curves for ISRB and V-SFD algorithms as shown in Figure 10.

Example 4: Here a NB-LDPC code(105, 2, 5) over $GF(8)$ is simulated using the 8 QAM modulation. The performance curves for the proposed algorithms and IJDD algorithms are illustrated in Figure 11. The BER performance curves depicted in Figure 11, shows that algorithm 2 outperforms by 3.0dB than Improved IJDD at BER 10^{-6}. The performance gap between the algorithm 3 and Imrpoved IJDD algorithm is around 4.0dB while algorithm 1 is around 3.8dB away from algorithm 2. Algorithm 1 is performing better than IJDD algorithm by 2dB. An individual decoder and QAM

TABLE 1. Computational complexity of different non-binary LDPC decoding algorithms per iteation.

Algorithms	Number of operations				
	GM	GA	IA/RA	RC/IC	IM/RM
Algorithm 1	$\gamma + \frac{1}{4}qd_v(d_c)$	$\gamma + \frac{1}{4}qd_v(d_c-1)$	$p(d_v+1)+\rho$	$\frac{1}{4}q(p+\rho)$	$2(p+\rho)$
Algorithm 2	$3nd_v$	$3nd_v$	$2n(q+1)$	$2n(q-1)$	nq
IIJDD	$3nd_v$	$3nd_v$	$2nd_v+n+2n(q+1)$	$2n(q-1)$	$3nq+4n$
EMS	$2nd_vn_m$	$9n_m(nd_v-2m)$	$n_m(20nd_v-18m-12n)$	$n_m log_2 n_m(9nd_v-12m-4n)$	
IJDD	$2nd_v$	$2nd_v-m$	nd_v+n+4n	$n(2q-3)$	$4n$
wt-Algo.B	$2nd_v$	$3nd_v-m$	nd_v	nd_v	nd_v
ISRB	$2nd_v$	$2nd_v-m$	nd_v+n2^r	$2n2^r-2n$	$n2^r$
V-SFD	$d_v(2d_c-1)$	$pd_v(d_v+r)+2d_v(d_c-1)$	$p(d_v^2+2d_vr+4r+2d_v)$	$nr+p(d_v-1)+2(p-1)$	
M-QAM FFT-QSPA	$(2q+1)nd_v$	$5nd_vq$	$6m+2qnd_vlog_2q$	$7m+qnd_v(d_v+d_c-2)$	$n(q-1)$

IM/IC/IA: Integer Multiplication/Comparision/Adittion; p is the total number of variable nodes stored in $\boldsymbol{\delta}$.
RA/RM/RC: Real Addition/Multiplication/Comparision; GA/GM: Galois Field Addtion/Multiplication;
Notations used: $\gamma = nd_v = md_c$, $L = \boldsymbol{v}^{\xi}$ such that $\xi < d_c, n_m < q$.

detector is also added to this example. Algorithm 3 is simulated without using the feedback to the 8 QAM detector. The BER performance curves shows that at 10.6db, the individual decoder-detector has achieved BER of 1.603^{-5} while the feedback based joint decoder-detector has achieved BER of 3.304^{-5}. Also at 10^{-3}, there is around 0.9dB difference which shows an obvious advantage of feedback based IJDD algorithm.

Example 5: In this example, a quasi cyclic high column NB-LDPC code $(120, 4, 8)$ over $GF(8)$ is simulated using the 8 QAM modulation. The performance curves for the proposed algorithms and Improved IJDD algorithm are illustrated in Figure 12. The BER performance curves depicted in Figure 12, shows that algorithm 2 outperforms by 4.7dB than Improved IJDD at BER $10^{-5.7}$. The performance gap between the algorithm 3 and Imrpoved IJDD algorithm is around 5.0dB while algorithm 1 is around 1dB away from algorithm 2 at the beginning and the gap is getting increased to more than 2.0dB.

From the examples, it is observed that as the order of the QAM modulation is increased, the Euclidean distance based QAM detection gets more and more inefficient due to the symbols placed in more close vicinity. This property of the QAM modulation effects the iterative joint detection and decoding as well. The basic concept in IJDD algorithm is to move the received symbol towards the correct location by removing the noise in steps. This step size is very important and can lead to over or under estimation if it is not adjusted properly. It is important to note that the effect of the factor f becomes very sensitive for an increased size of the M-QAM where M is the modulation order. An adaptive step size can be investigated in further research to improve the BER performance.

Soft decision decoding algorithms have high complexity in comparison to the hard decision decoding [7, 9] which requires only integer's addition and multiplications and maintain hardware friendly architecture [35]. The soft decision iterative decoders achieve better BER performance but the main concerns is the high decoding complexity while on the other hand, hard decision decoding offer low complexity which is of interest for hardware implementation.

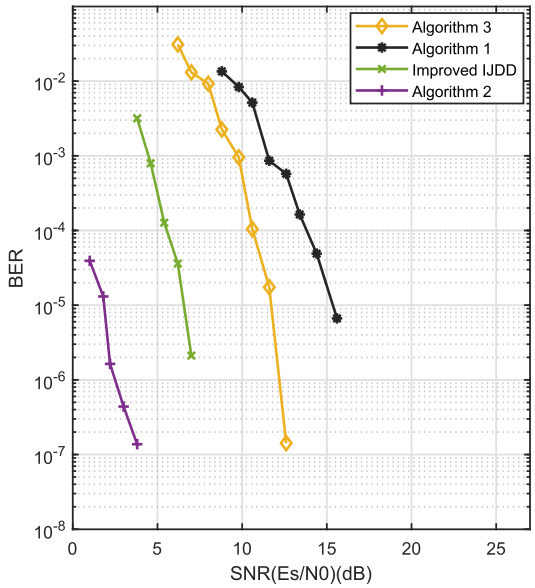

FIGURE 12. BER Performance of NB-LDPC Code(120,4,8) over GF(8).

Since in this paper, optimized LDPC codes from literature are used, therefore EXIT chart [2], [36] and density evolution [37] are not applied to do the code analysis.

V. COMPLEXITY ANALYSIS

In this section, the computation complexity of the proposed algorithms is analysed in detail and compared to that of the Improved IJDD algorithm. Consider an LDPC code (N, d_v, d_c) over GF(q) and let δ represents the nonzero elements in the parity check matrix $\boldsymbol{H}$ where $\gamma = nd_v = md_c$. At the ML QAM detector, q symbols of the QAM constellation are tested for each of the n received symbol sequence which results in nq real addition and multiplication. These operation are reduced to $(\frac{n}{4})q$ for the proposed quadrant based detection in this paper. The proposed Algorithms 1 and 3 require z_j only as opposite to IIJDD algorithm which require

the pair of $\{(\boldsymbol{z}_{j,1}, \boldsymbol{z}_{j,2}), \Delta d_j\}$ information. This results in further complexity reduction by $2nq + n$ real additions.

At the NB-LDPC decoder initialization, md_c Galois field multiplication and $m(d_v - 1)$ additions are required to compute the m check-sum and qn real comparisons are required to initialize the reliability information in case of Algorithm 2 and 3 only. For Algorithm 2, the update of the extrinsic information-sum $\sigma_{i,j}$ needs pair of $(\boldsymbol{z}_{j,1}, \boldsymbol{z}_{j,2})$, therefore, $2md_v$ Galois field multiplications and $2m(d_c - 1)$ Galois field additions are required. On the other hand Algorithms 1 and 3 require md_c multiplications and $m(d_c - 1)$ additions which means two times less computational complexity than IIJDD algorithm and Algorithm 2.

In the IIJDD algorithm, a pair of message $(v_j, \Delta\eta_j)$ is sent as feedback to the QAM detector while in the proposed algorithms, only the updated soft information $\boldsymbol{y}_j^{(k+1)}$ is used for the feedback. To compute the feedback $\boldsymbol{y}_j^{(k+1)}$, n real addition and multiplications are required for the proposed Algorithm 2 while for Algorithm 1, only ρ real addition and multiplications are required. To calculate the symbol flipping function in the Algorithm 1, pd_v real addition and p real multiplications are required. To compute the candidate symbol value for the position in error, d_v times the check-sum $\boldsymbol{s}_i$ is calculated. Therefore, to predict a single symbol value from the quadrant QAM constellation, $\frac{1}{4}qd_v(d_c - 1)$ Galois field addition and $\frac{1}{4}qd_v(d_c)$ multiplications are required for the symbols.

Table 1 shows the computational complexity of the proposed decoding algorithms in comparisons with other algorithms from the literature.

VI. CONCLUSION

In this paper, three algorithms are proposed for symbol flipping decoding of NB-LDPC over high order modulation. There is no symbol flipping decoding algorithm for the M-QAM in the literature. The results show significant performance improvement in terms of the bit error rate and complexity. The new concept of symbol flipping based IJDD is also showing better BER performance for the low column and row weights in comparison with the IJDD algorithms in literature. The proposed Algorithm 2 outperforms the Improved IJDD algorithm while Algorithm 1 and 3 perform better than IJDD. The step size of the Algorithm 2 is based on the Euclidean distance between the received symbol and the newly selected candidate symbol which help in moving the received symbol in correct direction. The numerical results and complexity analysis show that the proposed algorithms using the concept of iterative joint detection and decoding, offer an appealing trade-off between bit error rate and computational complexity.

ACKNOWLEDGMENT

The authors would like to thank Telkom South Africa and Sentech SOC LTD for sponsoring this research. They are also thankful to the anonymous reviewers and the Editor for their useful suggestions to improve this manuscript.

REFERENCES

[1] M. C. Davey and D. MacKay, "Low-density parity check codes over GF(q)," *IEEE Commun. Lett.*, vol. 2, no. 6, pp. 165–167, Jun. 1998.
[2] M. Sarvaghad-Moghaddam, W. Ullah, D. N. K. Jayakody, and S. Affes, "A new construction of high performance LDPC matrices for mobile networks," *Sensors*, vol. 20, no. 8, p. 2300, Apr. 2020.
[3] L. Barnault and D. Declercq, "Fast decoding algorithm for LDPC over GF(q)," in *Proc. Inf. Theory Workshop*, Mar. 2003, pp. 70–73.
[4] Q. Huang and S. Yuan, "Bit reliability-based decoders for non-binary LDPC codes," *IEEE Trans. Commun.*, vol. 64, no. 1, pp. 38–48, Jan. 2016.
[5] W. Ullah, L. Cheng, and F. Takawira, "Prediction and voting based symbol flipping non-binary LDPC decoding algorithms," in *Proc. IEEE 31st Annu. Int. Symp. Pers., Indoor Mobile Radio Commun.*, Aug. 2020, pp. 1–6.
[6] W. Ullah, L. Cheng, and F. Takawira, "Low complexity bit reliability and predication based symbol value selection decoding algorithms for non-binary LDPC codes," *IEEE Access*, vol. 8, pp. 142691–142703, 2020.
[7] C.-Y. Chen, Q. Huang, C.-C. Chao, and S. Lin, "Two low-complexity reliability-based message-passing algorithms for decoding non-binary LDPC codes," *IEEE Trans. Commun.*, vol. 58, no. 11, pp. 3140–3147, Nov. 2010.
[8] C. Xiong and Z. Yan, "Improved iterative soft-reliability-based majority-logic decoding algorithm for non-binary low-density parity-check codes," in *Proc. Conf. Rec. Forty 5th Asilomar Conf. Signals, Syst. Comput. (ASILOMAR)*, Nov. 2011, pp. 894–898.
[9] S. Yeo and I.-C. Park, "Improved hard-reliability based majority-logic decoding for non-binary LDPC codes," *IEEE Commun. Lett.*, vol. 21, no. 2, pp. 230–233, Feb. 2017.
[10] Q. Huang, M. Zhang, Z. Wang, and L. Wang, "Bit-reliability based low-complexity decoding algorithms for non-binary LDPC codes," *IEEE Trans. Commun.*, vol. 62, no. 12, pp. 4230–4240, Dec. 2014.
[11] B. Liu, J. Gao, G. Dou, and W. Tao, "Majority decision based weighted symbol-flipping decoding for nonbinary LDPC codes," in *Proc. 2nd Int. Conf. Future Comput. Commun.*, 2010, pp. 3–6.
[12] K. Jagiello and W. E. Ryan, "Iterative plurality-logic and generalized algorithm B decoding of Q-ary LDPC codes," in *Proc. IEEE Inf. Theory App. Workshop*, Feb. 2011, pp. 1–7.
[13] C.-C. Huang, C.-J. Wu, C.-Y. Chen, and C.-C. Chao, "Parallel symbol-flipping decoding for non-binary LDPC codes," *IEEE Commun. Lett.*, vol. 17, no. 6, pp. 1228–1231, Jun. 2013.
[14] N.-Q. Nhan, T. M. N. Ngatched, O. A. Dobre, P. Rostaing, K. Amis, and E. Radoi, "Multiple-votes parallel symbol-flipping decoding algorithm for non-binary LDPC codes," *IEEE Commun. Lett.*, vol. 19, no. 6, pp. 905–908, Jun. 2015.
[15] C. Chen, B. Bai, X. Ma, and X. Wang, "A symbol-reliability based message-passing decoding algorithm for nonbinary LDPC codes over finite fields," in *Proc. 6th Int. Symp. Turbo Codes Iterative Inf. Process.*, Sep. 2010, pp. 251–255.
[16] F. Garcia-Herrero, D. Declercq, and J. Valls, "Non-binary LDPC decoder based on symbol flipping with multiple votes," *IEEE Commun. Lett.*, vol. 18, no. 5, pp. 749–752, May 2014.
[17] W. Ullah and A. Yahya, "Comprehensive algorithmic review and analysis of LDPC codes," *Indonesian J. Electr. Eng. Comput. Sci.*, vol. 16, pp. 111–130, Oct. 2015.
[18] J. Chen and M. Fossorier, "Near optimum universal belief propagation based decoding of low-density parity check codes," *IEEE Trans. Commun.*, vol. 50, no. 3, pp. 406–414, Mar. 2002.
[19] D. Declercq and M. Fossorier, "Decoding algorithms for nonbinary LDPC codes over GF(q)," *IEEE Trans. Commun.*, vol. 55, no. 4, pp. 633–643, Apr. 2007.
[20] A. Voicila, D. Declercq, F. Verdier, M. Fossorier, and P. Urard, "Low-complexity decoding for non-binary LDPC codes in high order fields," *IEEE Trans. Commun.*, vol. 58, no. 5, pp. 1365–1375, May 2010.
[21] E. Li, D. Declercq, and K. Gunnam, "Trellis-based extended min-sum algorithm for non-binary LDPC codes and its hardware structure," *IEEE Trans. Commun.*, vol. 61, no. 7, pp. 2600–2611, Jul. 2013.
[22] V. Savin, "Min-max decoding for non binary LDPC codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2008, pp. 960–964.
[23] E. Boutillon and L. Conde-Canencia, "Bubble check: A simplified algorithm for elementary check node processing in extended min-sum non-binary LDPC decoders," *Electron. Lett.*, vol. 46, no. 9, p. 633, 2010.
[24] P. Schlafer, N. Wehn, M. Alles, T. Lehnigk-Emden, and E. Boutillon, "Syndrome based check node processing of high order NB-LDPC decoders," in *Proc. 22nd Int. Conf. Telecommun. (ICT)*, Apr. 2015, pp. 156–162.

[25] X. Wang, B. Bai, and X. Ma, "A low-complexity joint detection-decoding algorithm for nonbinary LDPC-coded modulation systems," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2010, pp. 794–798.
[26] S. Zhao, X. Wang, T. Wang, B. Bai, and X. Ma, "Joint detection–decoding of majority-logic decodable non-binary low-density parity-check coded modulation systems: An iterative noise reduction algorithm," *IET Commun.*, vol. 8, no. 10, pp. 1810–1819, Jul. 2014.
[27] M. Zhu, Q. Guo, B. Bai, and X. Ma, "Reliability-based joint detection-decoding algorithm for nonbinary LDPC-coded modulation systems," *IEEE Trans. Commun.*, vol. 64, no. 1, pp. 2–14, Jan. 2016.
[28] M. Zhu, Q. Guo, H. Xu, B. Bai, and X. Ma, "A multiple-voting-based decoding algorithm for nonbinary LDPC-coded modulation systems," *IEEE Access*, vol. 5, pp. 9739–9747, 2017.
[29] C. Poulliat, M. Fossorier, and D. Declercq, "Design of regular (2,dc)-LDPC codes over GF(q) using their binary images," *IEEE Trans. Commun.*, vol. 56, no. 10, pp. 1626–1635, Oct. 2008.
[30] X. Zhang, F. Cai, and S. Lin, "Low-complexity reliability-based message-passing decoder architectures for non-binary LDPC codes," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 11, pp. 1938–1950, Nov. 2012.
[31] H. Wymeersch, H. Steendam, and M. Moeneclaey, "Log-domain decoding of LDPC codes over GF (q)," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2004, pp. 772–776.
[32] O. Abassi, L. Conde-Canencia, A. Al Ghouwayel, and E. Boutillon, "A novel architecture for elementary-check-node processing in nonbinary LDPC decoders," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 64, no. 2, pp. 136–140, Feb. 2017.
[33] B. Liu, J. Gao, G. Dou, and W. Tao, "Weighted symbol-flipping decoding for nonbinary LDPC codes," in *Proc. 2nd Int. Conf. Netw. Secur., Wireless Commun. Trusted Comput.*, 2010, pp. 223–226.
[34] *Regular Ultra-Sparse Graphs and Related Non-Binary LDPC Codes*. Accessed: Aug. 2020. [Online]. Available: https://perso-etis.ensea.fr/~declercq/graphs.php
[35] Z. Xinmiao, *VLSI Architectures for Modern Error-Correcting Codes*. Boca Raton, FL, USA: CRC Press, Dec. 2017.
[36] G. J. Byers and F. Takawira, "EXIT charts for non-binary LDPC codes," in *Proc. IEEE Int. Conf. Commun.*, May 2005, pp. 652–657.
[37] G. Li, I. J. Fair, and W. A. Krzymien, "Density evolution for nonbinary LDPC codes under Gaussian approximation," *IEEE Trans. Inf. Theory*, vol. 55, no. 3, pp. 997–1015, Mar. 2009.

WAHEED ULLAH (Senior Member, IEEE) received the B.Sc. degree in electrical and electronics engineering from the University of Engineering and Technology Peshawar, Pakistan, and the master's degree in communication and information systems from the Nanjing University of Astronautics and Astronautics, China, in 2012. He is currently pursuing the Ph.D. degree with the University of the Witwatersrand, South Africa. His research interest includes channel coding for wireless communication. He is currently working on non-binary low density parity check (LDPC) codes.

LING CHENG (Senior Member, IEEE) received the B.Eng. degree *(cum laude)* in electronics and information from the Huazhong University of Science and Technology (HUST), in 1995, and the M.Ing. degree *(cum laude)* in electrical and electronics and the D.Ing. degree in electrical and electronics from the University of Johannesburg (UJ), in 2005 and 2011, respectively. In 2010, he joined the University of the Witwatersrand, where he was promoted to a Full Professor, in 2019. He has been a Visiting Professor at five universities and a Principal Advisor for more than 40 full research postgraduate students. He has published more than 100 research papers in journals and conference proceedings. His research interests include telecommunications and artificial intelligence. He was awarded the Chancellor's Medals in 2005, 2019, and the National Research Foundation rating, in 2014. The IEEE ISPLC 2015 Best Student Paper Award was made to his Ph.D. student in Austin. He is the Vice-Chair of the IEEE South African Information Theory Chapter. He serves as an Associate Editor for three journals.

FAMBIRAI TAKAWIRA (Member, IEEE) received the Ph.D. degree from Cambridge University, Cambridge, U.K., in 1986.

From 2012 to 2016, he was the Head of the School of Electrical and Information Engineering, University of the Witwatersrand. Prior to that, he was a Professor and the Dean of the Faculty of Engineering with the University of KwaZulu-Natal, Durban, South Africa. He is currently a Professor with the University of the Witwatersrand, South Africa. His research interests include wireless communication systems and networks. He has been an Active Volunteer in IEEE and ComSoc, the Director of Europe Middle East and Africa Region, from 2012 to 2013, an Editor of IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, from 2008 to 2010, and a Member of the COMSOC Board of Governors, from 2012 to 2013, the COMSOC Nominations and Elections Committee, from 2012 to 2017, and the COMSOC GIMS Committee, from 2013 to 2017. He is currently serving as a ComSoc Treasurer, from 2018 to 2021. He has also served as a General Co-Chair for IEEE-AFRICON 2015, an Executive Co-Chair for the International Conference on Communications 2010 (ICC2010), Cape Town South, Africa, and a member of Technical Committee for several IEEE AFRICON conferences.