Balasuriya and Wavegedara *EURASIP Journal on Wireless Communications and Networking* (2017) 2017:105
DOI 10.1186/s13638-017-0885-4

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Improved symbol value selection for symbol flipping-based non-binary LDPC decoding

Nuwan Balasuriya* and Chandika B. Wavegedara

Abstract

Symbol flipping-based hard decision decoding for non-binary low-density parity check (LDPC) codes has attracted much attention due to low decoding complexity even though the error performance of the symbol flipping decoder is inferior to that of the soft decision decoders. Standard symbol flipping decoding involves two steps, selection of the symbol position to be flipped and selection of the flipped symbol value. In this paper, an improved symbol value selection algorithm is developed for symbol flipping-based non-binary LDPC decoding. The key idea of the proposed algorithm is to use the complete information on correlation among the code symbols, in addition to their initial reliabilities when value of the flipped symbol is decided. The proposed algorithm offers improved error performance over the existing approaches of flipped symbol value selection which are solely based on the initial symbol reliabilities, with only a non-significant increase in complexity. At the same time, the proposed algorithm is low in complexity compared to other symbol flipping-based LDPC decoding algorithms which use the information on correlation among the code symbols in selecting the flipped symbol value.

Keywords: Non-binary low-density parity check codes, Symbol flipping, Flipped symbol value selection, Weighted voting

1 Introduction

Low-density parity check (LDPC) codes introduced by Galager [1] have capacity-approaching bit error rate (BER) performances. LDPC codes are characterized by sparse parity check matrices compared to other types of channel codes. There are numerous decoding techniques proposed in literature for LDPC codes, and they can be broadly categorized as soft decoding methods [1] and hard decoding methods [2, 3] as well as hybrid decoding methods which provide a trade-off between complexity and BER performance. Though soft decoding algorithms provide improved performance, for most practical applications, they may not be very appealing due to their high complexity. In particular, the use of soft decision-based decoding for finite geometry LDPC (FG-LDPC) codes is prohibitively complex. On the other hand, the hard decision-based bit flipping (BF) algorithm [2], which searches heuristically for a valid codeword in the multidimensional codeword space, is the simplest and most computationally efficient decoding algorithm.

In recent years, non-binary LDPC codes have drawn considerable interest due to their higher coding gains over binary LDPC codes [4]. In the literature, there are various algorithms proposed to decode non-binary LDPC codes with a complete hard decision decoding to retain the simplicity in decoding [5, 6]. These iterative non-binary LDPC decoding algorithms operate in two steps within each iteration. First, a symbol position is selected for flipping based on the current symbol reliabilities. Then, a new symbol value is selected based on the next highest reliability in replacing the existing symbol value at the selected symbol position. This procedure is carried out iteratively until a valid codeword is created or a predefined number of iterations are met. Existing research on hard decision-based non-binary LDPC decoding focuses only on improving the first step. To the best of our knowledge, none of the existing research attempts to improve the second step which also has a significant impact on

*Correspondence: nuwanbd@yahoo.com
Department of Electronic & Telecommunication Engineering, University of Moratuwa, Katubedda, Moratuwa, Sri Lanka

the performance of the overall decoding algorithm. On the other hand, [7–9] present three approaches in which the flipped symbol position selection and flipped symbol value selection steps are carried out together. However, we believe that there is a significant impact of the symbol value selection step on the performance of the overall decoding algorithm; thus, the performance of the overall symbol flipping algorithm can be improved by improving the flipped symbol value selection. To this end, in this paper, we propose a novel approach for selecting a new symbol value for the selected symbol position.

The rest of the paper is organized as follows. Sections 2 and 3 introduce the system model and the concept of symbol flipping in the domain of non-binary LDPC decoding. Section 4 presents the proposed flipped symbol value selection scheme and the overall decoding algorithm, which will be followed by a complexity analysis in Section 5. Section 6 presents the simulation results and the discussion. Finally, Section 7 concludes the paper giving suggestions for further research.

2 System model

We consider a wireless communication system shown in Fig. 1, where a continuous stream of bits is grouped into symbols drawn from a non-binary Galois field of order q, (GF($q = 2^p$)), where $p \in \mathbb{Z}^+$. A block of $n - m$ symbols is encoded using a $(n, n-m)$ non-binary LDPC code into an n dimensional code word $\mathbf{s} = [s_1, s_2, \ldots, s_n]$ over GF(q). The binary converted codeword $\mathbf{c} \in \{0,1\}^{np}$ is then binary phase shift keying (BPSK) modulated into an np dimensional modulated vector $\mathbf{x} = [x_1, x_2, \ldots, x_{np}]$, where $x_j \in \{-1, 1\}$ for $j \in [1, np]$. It should be noted that although we consider BPSK modulation for simplicity, any higher-order linear modulation technique can be used. We assume an additive white Gaussian noise (AWGN) channel. Then, the received signal vector $\mathbf{y} = [y_1, y_2, \ldots, y_{np}]$ corresponding to the codeword can be expressed as,

$$\mathbf{y} = \mathbf{x} + \mathbf{n}, \tag{1}$$

where $\mathbf{n} = [n_1, n_2, \ldots, n_{np}]$. n_i, $i \in \{1, \ldots, np\}$ are the iid complex AWGN samples with zero mean and σ^2 variance per real dimension. $\mathbf{y}$ is taken as the input to the symbol flipping LDPC decoder.

3 Non-binary LDPC decoding

The standard iterative non-binary LDPC symbol flipping decoding algorithm consists of three steps: initialization, flipped symbol position selection, and flipped symbol value selection, where the latter two steps are iteratively run until a valid codeword is found [5].

3.1 Initialization

With the received signal value y_j corresponding to the j^{th} bit position, the a posteriori probability of having sent x_j given that y_j has been received,

$$Pr(x_j|y_j) = \frac{f(y_j|x_j)Pr(x_j)}{Pr(y_j)}. \tag{2}$$

Furthermore, $f(y_j|x_j) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{\|y_j - x_j\|^2}{2\sigma^2}\right)$. We assume that $Pr(x_j = +1) = Pr(x_j = -1) = \frac{1}{2}$. Hence,

$$Pr(x_j|y_j) = \frac{1}{2C}\exp\left(-\frac{\|y_j - x_j\|^2}{2\sigma^2}\right), \tag{3}$$

where constant C can be calculated using the fact that $\sum Pr(x_j|y_j) = 1$.

With the assumption that the constituent bits within a symbol s_k, $k \in [1, n]$ are independent, $Pr(s_k|\mathbf{y}) = \prod_{j=(k-1)p+1}^{kp} Pr(x_j|y_j)$. The initial hard estimate of symbol s_k can be obtained using the following criterion:

$$\hat{s}_k = \mathcal{S}(\nu), \tag{4}$$

where $\mathcal{S}$ is the GF(q) symbol space and $\nu = \arg\max_\eta Pr(s_k = \mathcal{S}(\eta)|\mathbf{y})$. Furthermore, the log-likelihood ratio (LLR) of symbol $\mathcal{S}(\eta)$ at each symbol position s_k can be expressed as

$$llr(s_k = \mathcal{S}(\eta)|\mathbf{y}) = \log\left(\frac{Pr(s_k = \mathcal{S}(\eta)|\mathbf{y})}{1 - Pr(s_k = \mathcal{S}(\eta)|\mathbf{y})}\right). \tag{5}$$

3.2 Symbol position selection

In this work, we adopt the same concept as in the single-bit flipping algorithm of [2], where the flipped symbol position is selected based on the correlation among the

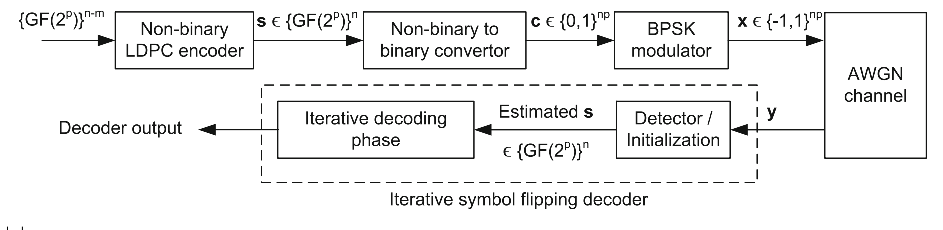

Fig. 1 System model

symbols which depends on the structure of the code and the initial reliabilities of the symbols captured from the received values.

Let the reliabilities of the n symbols at the tth iteration be $\mathbf{rel}^{(t)} = \left[rel_1^{(t)}, \ldots, rel_n^{(t)}\right] = \left[\left|llr(s_1|\mathbf{y})\right|, \left|llr(s_2|\mathbf{y})\right|, \ldots, \left|llr(s_n|\mathbf{y})\right|\right]$ where s_k now represents the current selected symbol at the k^{th} position. In each iteration, the decoder calculates a reliability metric ϕ_k, for each symbol $k = 1, \ldots, n$ separately, based on the set of parity check equations which depend on the considered symbol.

Let $\mathcal{N}(i)$, $i = 1, \ldots, m$, denote the set of symbols which contributes to the ith parity check equation and $l_i^{(t)} = \min_{k \in \mathcal{N}(i)} rel_k^{(t)}$ and $u_i^{(t)} = \max_{k \in \mathcal{N}(i)} rel_k^{(t)}$. Then,

$$\phi_{ki}^{(t)} = \begin{cases} rel_k^{(t)} - \frac{l_i^{(t)}}{2} & \text{if } k \in \mathcal{N}(i) \text{ and } \mathrm{z}_i^{(t)} = 0 \\ rel_k^{(t)} - \frac{l_i^{(t)}}{2} - u_i^{(t)} & \text{if } k \in \mathcal{N}(i) \text{ and } \mathrm{z}_i^{(t)} \neq 0, \end{cases} \tag{6}$$

where $\mathrm{z}_i^{(t)}$ is the i^{th} parity check syndrome symbol calculated from the input symbol vector to the decoder at the t^{th} iteration. Then, the metric value for the k^{th} symbol position at the t^{th} iteration is computed as

$$\phi_k^{(t)} = \sum \phi_{ki}^{(t)}, \quad k = 1, \ldots, n. \tag{7}$$

Note that the lower the $\phi_k^{(t)}$, the higher the chance for the symbol to be in error.

Thus, the corresponding position to be flipped during the t^{th} iteration is

$$k' = \arg\min_k \phi_k^{(t)}. \tag{8}$$

3.3 Flipped symbol value selection and finalization

In bit flipping-based binary LDPC decoding algorithms, after the selection of the bit to be flipped, the current bit value is inverted with the simple NOT rule and the resulting bit vector is tested for a valid codeword. If successful, the algorithm stops or otherwise the algorithm proceeds to the next iteration with the current hard decided bit vector. However, in the symbol flipping algorithm for non-binary LDPC codes, it is required to select the best symbol value from a set of $q - 1$ possible symbol values, as the flipped symbol.

As stated in Section 1, in the existing symbol flipping algorithms, there are two approaches for the selection of the new symbol value. In the approach of [5, 6], the selection of the flipped symbol value is based on the reliabilities of different symbol values calculated from the channel output values. Symbol value with the next best reliability is selected as the new flipping symbol value at the considered position. Note that the selected symbol position for flipping can be transformed in to p bits over GF(q) each corresponding to a channel output value given by y_j, $kp \leq j < (k + 1)p$. Reliability of the bit j is proportional to $|y_j|$ [5]; hence, a set of f, $(1 \leq f \leq p)$ bits corresponding to the lowest $|y_j|$ values are selected and flipped using the simple NOT rule. The value f known as *flagbit* is a parameter which starts from 1 at the first decoding iteration and then increments by 1 at each successive iteration. The resulting block of p bits corresponding to the considered symbol is converted back to non-binary to form the new flipped symbol. This approach considers only the initial realizing value-based reliabilities of the received symbols/bits. The correlation among the symbols depending on the structure of the channel code is not exploited in the aforementioned symbol value selection approach. Incorporation of this additional information in the flipped symbol value selection can be expected to produce an improved overall decoding performance. In the approach of [7–9], a search for a valid codeword is carried out in the extended symbol combination set formed by considering all possible symbols that a position can have. Although this second approach improves the decoding performance of the overall non-binary LDPC decoding algorithm, a dedicated flipped symbol value selection can be expected to further enhance the overall performance. Furthermore, a low complexity in the overall algorithm is highly desired, and this low complexity can be expected from a decoding algorithm with a dedicated symbol value selection step.

4 Proposed symbol value selection algorithm

In order to improve the performance of the existing non-binary LDPC decoding algorithms discussed in Section 3 while retaining a low complexity, we propose a symbol value selection algorithm which utilizes both the knowledge of the symbol reliabilities and the knowledge of the correlation introduced by the channel code.

At each iteration, for the selected position to be flipped, all other possible candidate symbol values are voted with a positive metric value for each parity check equation the considered symbol value satisfies. Furthermore, the voting metric value is selected to be a negative for each parity check the candidate symbol value does not satisfy. If the selected metric value is selected to be a fixed number, the reliabilities based on the initial realization of received values are not considered in selecting the new symbol. When a certain symbol position is considered for flipping, a certain parity check may not be satisfied due to the erroneous symbols in positions other than the selected position. This results in an erroneous metric value. However, these errors can be minimized by harnessing the information obtained from the initial reliabilities.

Let the selected position at the t^{th} iteration be k' and let $\mathcal{M}(k')$ denote the parity checks in which $s_{k'}$ is involved in.

Then, the metric value for the candidate symbol value $\mathcal{S}_v$ $\left(\mathcal{S}_v \in GF(q) \text{ and } \mathcal{S}_v \neq s_{k'}^{(t)}\right)$ from the i^{th} parity check

$$v_{k',i}^{(t)}(\mathcal{S}_v) = \begin{cases} \Gamma_i^{(t)} & \text{if } \mathrm{z}_i^{(t)}(\mathcal{S}_v) = 0 \\ -\Gamma_i^{(t)} & \text{if } \mathrm{z}_i^{(t)}(\mathcal{S}_v) \neq 0, \end{cases} \tag{9}$$

where $\Gamma_i^{(t)} = \left| rel_{k'}^{(t)} - \min_{j \in \mathcal{N}(i)} rel_j^{(t)} \right|$ and $\mathrm{z}_i^{(t)}(\mathcal{S}_v)$ represents the i^{th} element of the syndrome vector ($i \in \{1, 2, \ldots, m\}$) with $\mathcal{S}_v$ in the k'^{th} symbol position. Furthermore, the accumulated metric value is given as

$$v_{k'}^{t}(\mathcal{S}_v) = \sum_{i \in \mathcal{M}(k')} v_{k',i}^{t}(\mathcal{S}_v). \tag{10}$$

The new symbol for the selected k'^{th} position at the end of the t^{th} iteration is found as,

$$s_{k'} = \arg\max_{\mathcal{S}_v} v_{k'}^{(t)}(\mathcal{S}_v). \tag{11}$$

5 Computational complexity

In this section, we evaluate the computational complexity of the proposed symbol value selection algorithm and compare it to those of several existing symbol flipping-based non-binary LDPC decoding algorithms. Let w_c and w_r denote the column weight and row weight of the parity check matrix, respectively. For the purpose of analyzing the computational complexity, we consider any subsequent iteration, except the first iteration. We assume that the LLRs of the symbols and the initial syndrome vector are available for the symbol-value selection, and hence, the operations are not considered in the complexity calculation. Also, w_z denotes the weight of the syndrome vector at the considered iteration. Moreover, in order to reduce the complexity, the algorithm in [8] considers only a subset of the possible combinations of symbol vectors with n symbols, during the process of searching for a valid codeword. This is achieved by limiting the number of symbol positions and also the number of candidate symbol values per position considered during the process of calculating the *flipping function* [8]. Let the number of positions and symbol values per position considered while selecting the candidate set be η, ($\eta \leq w_r$) and ϵ, ($\epsilon \leq q$) respectively. Note that $\epsilon = 2$ in [8]. It can be shown that the number of real additions, multiplications, and comparisons required for the proposed and existing decoding algorithms are as given as in Table 1.

The proposed flipped symbol value selection step's complexity increases exponentially with the increased order of the Galois field. On the other hand, the computational complexity of the algorithm of [8] increases exponentially with the η value.

Table 1 Number of real operations for the symbol flipping algorithm

	Symbol position selection	Symbol value selection
RBA [5]	$(m + w_z)w_r + (w_c - 1)n$	0
PSF [7]		$7mw_r - m$
MVA [8]		$m\eta\epsilon^{\eta} + (w_c - 1)n\epsilon$
MVPSF [9]		$7mw_r - n + n\log_2(n)$
Proposed	$(m + w_z)w_r + (w_c - 1)n$	$2^p w_c + (w_c - 1)(2^p - 1)$ $+2^p w_c + (2^p - 1)w_c$

In Tables 2, 3, and 4, we compare the computational complexity of the proposed algorithm with those of the existing algorithms namely, reliability-based symbol value selection algorithm (RBA), parallel symbol flipping algorithm (PSF), multiple-vote symbol flipping algorithm (MVA), and multiple-vote parallel symbol flipping algorithm (MVPSF) in terms of the number of real operations required per iteration for the chosen 204 × 102 LDPC code over GF(4), 63 × 37 LDPC code over GF(16), and 1023 × 781 LDPC code over GF(16).

For the 204 × 102 LDPC code, with $w_c = 3$ and $w_r = 6$, we can see from Table 2 that the symbol position selection step requires approximately 1326 real operations per iteration on average. Compared to no operations required when the reliability-based algorithms of [5] is employed, the symbol selection step requires only 39 real operations per iteration in average when the proposed algorithm is employed. It is clear that when the total number of operations required per iteration (on average) for the symbol-flipping-based LDPC decoding algorithm is considered, increase in the computational complexity due the proposed symbol value selection algorithm is negligible for the 204 × 102 LDPC code in GF(4). Also compared to the algorithm of [8], the proposed algorithm has significantly less complexity for $\epsilon = 2$ and $\eta = \frac{w_r}{2} = 3$. The complexity of the algorithms of [7] and [9] are in the same order as the proposed algorithm, but are nearly 20 and 30% higher than the complexity of the proposed algorithm, respectively.

Table 2 Number of real operations for the symbol flipping algorithm for 204 × 102, GF(4) LDPC code

	Symbol position selection	Symbol value selection
RBA [5]	1326	0
PSF [7]		4182
MVA [8]		3264
MVPSF [9]		5645
Proposed	1326	39

Table 3 Number of real operations for the symbol flipping algorithm for 63 × 37, GF(16) LDPC code

	Symbol position selection	Symbol value selection
RBA [5]	753	0
PSF [7]		1430
MVA [8]		2722
MVPSF [9]		1769
Proposed	753	481

For the 63 × 37 LDPC code, with $w_c = w_r = 8$, Table 3 shows that the symbol position selection step requires approximately 753 real operations per iteration on average. Compared to no operations required when the reliability-based algorithm of [5] is employed, the symbol selection step requires only 481 real operations per iteration in average when the proposed algorithm is employed. Note that the higher Galois field order has contributed to a comparatively large number of operations required for the symbol value selection step. However, the increase in the overall computational complexity due the proposed symbol value selection algorithm is negligible for the 63 × 37 LDPC code in GF(16) too. Also compared to the algorithm of [8], the proposed algorithm is still less complex for $\epsilon = 2$ and $\eta = \frac{w_r}{2} = 4$. When the computational complexity per iteration is considered for the 1023 × 781 LDPC code, with $w_c = w_r = 32$, Table 4 shows that the symbol position selection step requires approximately 43,329 real operations per iteration on average while the proposed symbol value selection step requires 1969 real operations. Compared to the 559,042 operation required for the algorithm of [8] with $\eta = 8$, the proposed algorithm shows far less overall complexity. Furthermore, the computational complexities of algorithms of [7] and [9] are still in the same order as the proposed algorithm and are nearly 20 and 30% higher than the complexity of the proposed algorithm, respectively.

6 Simulation results and discussion

In this section, we investigate the error performance of the symbol flipping-based non-binary LDPC decoding algorithm consisting of the proposed symbol value selection step. Recall that the overall algorithm adopts the initialization and flipped symbol position selection of the symbol flipping-based decoding algorithm [2], along with the proposed symbol value selection scheme. Here onwards, the overall decoding algorithm is referred to as the proposed decoding algorithm. In our performance investigation we consider the following LDPC codes: a 204 × 102 LDPC code over GF(4), a 63 × 37 LDPC code over GF(16), and a 1023 × 781 LDPC code over GF(16). We assume an AWGN channel where iid zero-mean Gaussian noise samples are added to the received signal.

Table 4 Number of real operations for the symbol flipping algorithm for 1023 × 781, GF(16) LDPC code

	Symbol position selection	Symbol value selection
RBA [5]	43329	0
PSF [7]		53966
MVA [8]		559042
MVPSF [9]		63173
Proposed	43329	1969

We present the BER and frame error rate (FER) performances of the proposed algorithm in Figs. 2, 3, and 4. For comparison purposes, we also present the BER and FER performances of the symbol flipping LDPC decoding algorithm of [5], parallel symbol flipping algorithm of [7], multiple-vote symbol flipping LDPC decoding algorithm of [8], and multiple-vote parallel symbol flipping LDPC decoding algorithm of [9].

It can be observed from Figs. 2, 3, and 4 that the proposed algorithm offers an improvement of approximately 0.8 dB in the BER performance at 10^{-4} BER level compared to the two-stage algorithm in [5]. This improvement is due to the exploitation of the correlation among the symbols of the codeword in the flipped-symbol value selection step. Moreover, we can observe that the proposed algorithm outperforms the algorithm of [8] by about 0.3 dB and the algorithm of [7] by about 0.4 dB in the BER performance at 10^{-4} BER level. Meanwhile the FER performance of the proposed algorithm is better than the algorithms of [5, 7], and [8] by approximately 1, 0.5, and 0.3 dB, respectively at 10^{-3} FER level. However, the error performance of the proposed algorithm is very similar to that of the algorithm of [9]. Note that the algorithm of [9] is a combined algorithm of those in [7] and [8], thus harnessing the coding gains of both the algorithms. The dedicated flipped symbol value selection step in the proposed algorithm has resulted in a coding gain that matches the combined high coding gain of the algorithm of [9]. Furthermore, it was observed from simulation that the symbol-flipping decoding with the reliability-based symbol value selection algorithm of [5] achieves convergence after 6.5 iterations on average, whereas the proposed algorithm requires only 4.8 iterations for convergence. Thus, based on the complexity analysis carried out in section 5, the overall computational complexity of the proposed algorithm is only marginally higher than that of the algorithm of [5]. At the same time, the proposed algorithm is considerably simpler in terms of the real operations per iteration, compared to the algorithms of [8] and [9]. Specially, due to the employment of the dedicated flipped symbol value

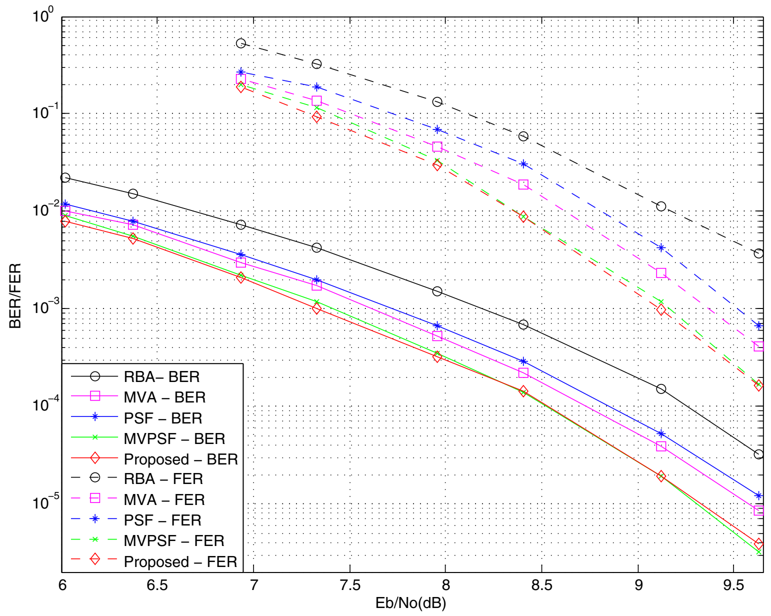

Fig. 2 Comparison of decoder BER performance for 204 × 102, GF(4) LDPC code

selection step, the proposed algorithm obtains an error performance similar to that of [9], with lesser decoder complexity.

7 Conclusions

We have proposed an improved symbol value selection algorithm which harnesses information from both code correlation and initial received value reliabilities to improve the overall BER performance. Simulation results have demonstrated that the proposed algorithm clearly outperforms all the existing algorithms without much added complexity except the algorithm of [9]. The proposed algorithm achieves the same error performance as in [9]. However, the proposed algorithm is considerably

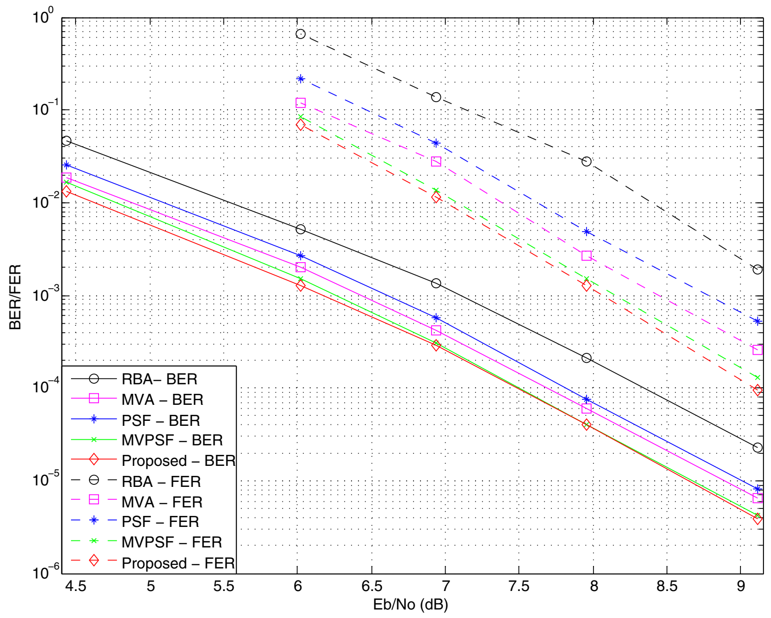

Fig. 3 Comparison of decoder BER performance for 63 × 37, GF(16) LDPC code

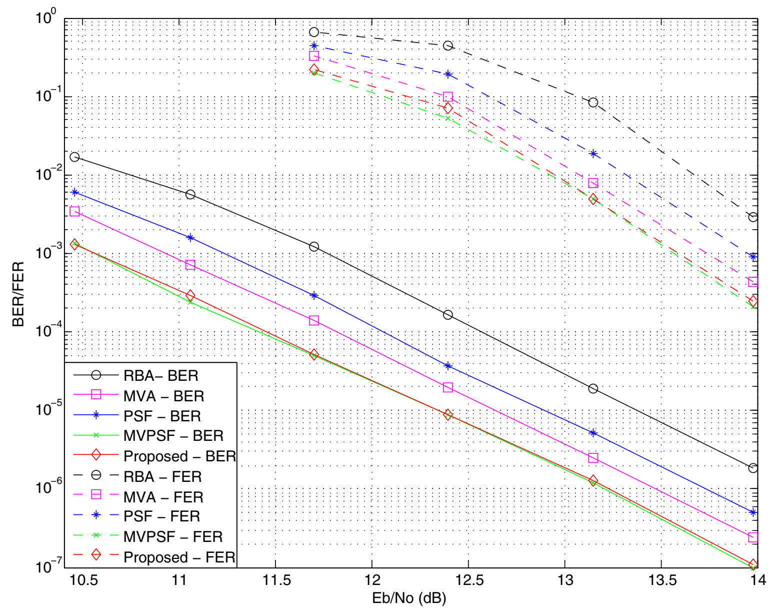

Fig. 4 Comparison of decoder BER performance for 1023 × 781, GF(16) LDPC code

superior when compared to [9] in terms of the decoder complexity.

Meanwhile, the computational complexity per iteration in the proposed algorithm exponentially increases with the increased order of the non-binary field. This constraint limits the use of higher-order non-binary fields which is a major drawback of the proposed algorithm. A technique to overcome this limitation would lead to a future research area. An algorithm with the relative probability-based LLR defined as the logarithm of the ratio between the best probability and the next best will also be an interesting future research area.

Acknowledgments
The authors would like to acknowledge the encouragements and suggestions provided by Prof. Dileeka Dias during this research.

Funding
Not applicable.

Authors' contributions
Both authors contributed to the searching of the literature and development of the proposed algorithm and also carried out the simulations. Both authors have read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Consent for publication
Not applicable.

Ethics approval and consent to participate
Not applicable.

Publisher's Note

Received: 25 October 2016 Accepted: 28 April 2017
Published online: 12 June 2017

References

1. R Gallager, Low-density parity-check codes. IRE Trans. Info. Theory. **8**, 21–28 (1962)
2. Z Liu, DA Pados, A decoding algorithm for finite-geometry LDPC Codes. IEEE Trans. Comm. **53**(3), 415–421 (2005)
3. T Ngatched, F Takawira, M Bossert, An improved decoding algorithm for finite-geometry LDPC Codes. IEEE Trans. Comm. **57**(2), 302–306 (2009)
4. MC Davey, D MacKay, Low density parity check codes over $GF(q)$. IEEE Comm. Lett. **2**(6), 165–167 (1998)
5. B Liu, J Gao, G Dou, W Tao, in *IEEE Intern. Conf. on Network Security, Wireless Communications and Trusted Computing*. Weighted symbol-flipping decoding for non-binary LDPC codes (IEEE, Wuhan, 2010), pp. 223–226
6. B Liu, J Gao, G Dou, W Tao, in *The 2nd Intern. Conf. on Future Computer and Communication*. Majority decision based weighted symbol-flipping decoding for non-binary LDPC codes (IEEE, Wuhan, 2010), pp. V3.6–V3.10
7. C Huang, C Wu, C Chen, C Chao, Parallel symbol-flipping decoding for non-binary LDPC codes. IEEE Comm. Lett. **17**(6), 1228–1231 (2013)
8. F Garcia Herrero, E Li, D Declercq, J Valls, Multiple-vote symbol-flipping decoder for non-binary LDPC codes. IEEE Trans. VLSI Sys. **22**(11), 2256–2267 (2014)
9. N Nhan, TMN Ngatched, OA Dobre, P Rostaing, K Amis, E Radoi, Multiple-votes parallel symbol-flipping decoding algorithm for non-binary LDPC codes. IEEE Comm. Lett. **19**(6), 905–908 (2015)