

Q-ary LDPC Decoders with Reduced Complexity

X. H. Shen & F. C. M. Lau

Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong

Email: shenxh@eie.polyu.edu.hk & encmlau@polyu.edu.hk

Abstract—Q-ary low-density parity-check (LDPC) codes achieve exceptional error performance at the expense of computation simplicity. Solutions to accelerate the decoding process have become one of the focuses in literature. In this paper, a decoding method is proposed, based on the subcode concept, to speed up the dominant iterative process. The method leads to speed improvement with moderate error-performance penalty.

Keywords—Bit error rate, complexity, LDPC code, q-ary LDPC code, subcode.

I. INTRODUCTION

Binary low-density parity-check (LDPC) codes are defined by very sparse parity-check matrices in which most of the elements are “0”s and the remaining entries are “1”s [1]–[3]. Q-ary LDPC codes (LDPC codes over finite fields $GF(q)$), demonstrated superior error performance over their binary counterparts [4], accompanied by a rapid increase in computation complexity. Log-domain and Fourier-domain interpretations of the belief propagation (BP) algorithm have greatly ameliorated the problem [5], [6]. The extended Min-Sum (EMS) decoder in [6], [7] selects the most probable codewords to further simplify the computation. [8] proposed Log-domain based selective Min-Max algorithm as an improved version of the Min-Sum algorithm in [5]. It accomplishes the task by selecting the most probable codewords based on the reliabilities assigned to various values of a symbol. [9] considers long error-correcting codes as constructed from shorter codes (referred to as *subcodes*) and a bipartite graph. The bipartite graph connects the symbols of the long code (variable nodes) to their corresponding subcodes (check nodes), where each of the subcodes serves as a local computation center [10]. The long code is an LDPC code when each subcode is a parity check. For a subcode with length d_r (i.e., the row weight of the parity-check matrix), there exist q^{d_r-1} valid subcode codewords to be examined in the maximum likelihood decoding. Therefore, even for a moderate value of q , the total number of computations involved in the check-node updating is enormous and hence the decoding of q-ary LDPC codes remains prohibitive.

In this paper, a new method of check-node updating based on Tanner’s subcode concept [9] is introduced. The processing in the check-node updating resembles the Chase algorithm [11] and the ordered statistical decoding (OSD) algorithm [12]. It results in substantial decoding-time improvement with moderate degradation in error performance. Our algorithm is useful under such scenarios as: i) fast evaluation of error performance of codes in the high SNR region; ii) fast comparison of various codes in terms of error performance.

The paper is organized as follows: Section II describes the proposed q-ary LDPC decoding method; Section III shows the simulation results concerning both the speed and error-performance aspects; and finally a conclusion is given in Section IV.

II. PROPOSED DECODING METHOD

In the following, our proposed the decoding mechanism for the subcode and then the decoding algorithm of the q-ary LDPC decoder is described. In particular, we propose a method that reduces the number of possible codewords to be considered for a subcode based on an algebraic decoder of the subcode.

A. Subcode in Q-ary LDPC Codes

For each check node in the bipartite graph of a q-ary LDPC code, let h_i ($i = 1, 2, \dots, d_r$) denote the non-zero entries in the parity-check matrix $\mathbf{H}$ corresponding to the given check node, and let c_i ($i = 1, 2, \dots, d_r$) be the symbols involved in the check-node computation. Assuming that $q = 2^p$, we denote the binary image of h_i by a square matrix $\mathbf{H}_i$ of size $p \times p$ and the symbol c_i by a binary vector $\mathbf{b}_i = (b_{i,1}, b_{i,2}, \dots, b_{i,p})$ of length p [13]. The parity-check equation for the given check node can then be written as

$$\sum_{i=1}^{d_r} \mathbf{b}_i \mathbf{H}_i = \mathbf{0} \quad (1)$$

where $\mathbf{b}_i$, $i = 1, 2, \dots, d_r$ and $\mathbf{0}$ are p -dimensional binary vectors.

Let $\mathbf{S}$ denote the concatenation of all $\mathbf{H}_i^T$ ($i = 1, 2, \dots, d_r$), i.e., $\mathbf{S} = [\mathbf{H}_1^T \ \mathbf{H}_2^T \ \dots \ \mathbf{H}_{d_r}^T]$, where $(\cdot)^T$ is the transpose operator, and $\mathbf{c}$ be the concatenation of all $\mathbf{b}_i$, i.e., $\mathbf{c} = [\mathbf{b}_1 \ \mathbf{b}_2 \ \dots \ \mathbf{b}_{d_r}]$. Consequently, the subcode can be regarded as a binary parity-check code of length pd_r with parity-check matrix $\mathbf{S}$ and code block $\mathbf{c}$. The subcode defined by $\mathbf{S}$ then takes the responsibility of choosing a relatively small portion of all possible codewords for a check node so as to ameliorate the computation complexity of the q-ary LDPC decoder.

B. Decoding Flow

We consider the Min-Sum decoding mechanism in our approach and a binary-input additive-white-Gaussian-noise (BIAGWN) channel is assumed. Note that to facilitate the check-node updating process, the messages passed during the iterative decoding will be reliabilities with regard to *bits* instead of *symbols*.

1) *Initialization*: Let M and N denote the number of rows and columns in the parity-check matrix $\mathbf{H}$ of a q -ary LDPC code. The received vector $\mathbf{r}$ can be written as an Np -dimensional vector, i.e.,

$$\mathbf{r} = (r_{1,1}, r_{1,2}, \dots, r_{1,p}, r_{2,1}, r_{2,2}, \dots, r_{2,p}, \dots, r_{N,1}, r_{N,2}, \dots, r_{N,p}), \quad (2)$$

where $\{r_{n,1}, r_{n,2}, \dots, r_{n,p}\}$ corresponds to the n th symbol ($n = 1, 2, \dots, N$) of the code block. For the n th variable node, the initialized message vector $\mathbf{s}_n = (s_{n,1}, s_{n,2}, \dots, s_{n,p})$ is computed using

$$s_{n,k} = \frac{2r_{n,k}}{\sigma^2}, \quad k = 1, 2, \dots, p; \quad (3)$$

where σ is the standard deviation of the channel noise.

Let $\alpha_n^{(m)} = (\alpha_{n,1}^{(m)}, \alpha_{n,2}^{(m)}, \dots, \alpha_{n,p}^{(m)})$ denote the message vector transmitted from the n th variable node to the m th check node, and let $\beta_m^{(n)} = (\beta_{m,1}^{(n)}, \beta_{m,2}^{(n)}, \dots, \beta_{m,p}^{(n)})$ represent the message vector sent from the m th check node to the n th variable node. After the initialization process, we set $\alpha_n^{(m)} = \mathbf{s}_n$ for all $n = 1, 2, \dots, N$.

2) *Check-node updating* $\beta_m^{(n)}$: We consider each check node as a subcode. For each variable node connected to the check node, we divide the process of determining the reliabilities of different symbol values of the variable node into two separate stages: (a) construction of a set of possible codewords based on the subcode's parity-check matrix and the received symbol vector; and (b) updating the messages from the check node to the variable node based on the set of codewords found in (a).

a) *Construction of a set of possible codewords*: Let V_m denote the set of variable nodes incident to the m th check node. Referring to Section II-A and using the received vector $\mathbf{r}$ (or the updated reliabilities $Q_{n,k}$ in Eq. (9) after each iteration), the symbols c_i and hence the vectors $\mathbf{b}_i$ ($i = 1, 2, \dots, d_r$) are determined using hard-decision. The subcode corresponding to this check node is fed with $\mathbb{C}$ and a set of reliabilities $\alpha_n^{(m)}$ ($n \in V_m$). In $\mathbb{C}$, we consider the g (an adjustable parameter) bits with the least reliabilities, called the "least reliable bits" (LRBs). By fixing the non-LRBs and letting the LRBs take on all possible values (1 or 0) in all combinations, a total of 2^g pd_r -dimensional binary vectors are obtained, denoted by $\mathbb{C}_t$ where $t = 1, 2, \dots, 2^g$.

Each of the vectors $\mathbb{C}_t$ ($t = 1, 2, \dots, 2^g$) is then decoded with the subcode's algebraic decoder into a codeword $\mathbb{C}_t$, i.e., $\mathbb{C}_t = \mathbf{f}(\mathbb{C}_t)$ where $\mathbf{f}(\cdot)$ denotes the algebraic decoder. Consequently, a set of 2^g codewords¹ $P = \{\mathbb{C}_1, \mathbb{C}_2, \dots, \mathbb{C}_{2^g}\}$ for the subcode are obtained. We evaluate the reliability R_t for each $\mathbb{C}_t$ ($t = 1, 2, \dots, 2^g$), using

$$R_t = \sum_{\substack{n \in V_m \\ k = 1, 2, \dots, p \\ \mathbb{C}_t(l_{n,k}) = 1}} \alpha_{n,k}^{(m)} \quad (4)$$

¹In the original q -LDPC decoder, $2^{p(d_r-1)}$ subcode codewords will have to be considered but here we only need to consider 2^g codewords where $2^g \ll 2^{p(d_r-1)}$.

where $l_{n,k}$ denotes the bit location in $\mathbb{C}_t$ that corresponds to the k th bit of the n th variable node; and $\mathbb{C}_t(l_{n,k})$ denotes the value of the $l_{n,k}$ th bit in $\mathbb{C}_t$.

Two problems may arise with the algebraic decoder. Firstly, some $\mathbb{C}_t$ vectors may give rise to more than one possible output codewords. Here, if it happens, we will simply compare the reliabilities of the possible output codewords and select the codeword with the maximum reliability R_t . Secondly, some $\mathbb{C}_t$ contains more error bits than the decoding capacity of the algebraic decoder. In this case, we will flip one or more non-LRBs with an aim to attaining a decodable vector. The flipping process is illustrated in Fig. 1. The bits in $\mathbb{C}_t$ are ordered according to their reliabilities, where more reliable ones are illustrated with lighter colors and the LRBs are marked in dark red color (see Fig. 1(a)). The process starts by flipping the non-LRB with the least reliability in $\mathbb{C}_t$ to generate a code block, as in Fig. 1(b), where the flipped bit is marked in blue color. If the code block is decodable, it will be used to replace $\mathbb{C}_t$ and the flipping process is completed. Otherwise, the non-LRB with the second least reliability in $\mathbb{C}_t$ will be flipped. The process continues until we flip the bits with the largest reliability. If a decodable code block still cannot be found, we will begin to flip two non-LRBs in $\mathbb{C}_t$. Similarly to the previous case, we aim to flip two non-LRBs in $\mathbb{C}_t$ with the least reliabilities (grey and blue ones in Fig. 1(c)), and so on, until a decodable code block is found to replace $\mathbb{C}_t$. In summary, the flipping process aims to flip a minimum number of non-LRBs with the least reliabilities such that a decodable code block can be found to replace $\mathbb{C}_t$.

b) *Message updating*: Consider the message $\beta_{m,k}^{(n)}$ ($k = 1, 2, \dots, p$) to be sent from the m th check node to the n th variable node. $\beta_{m,k}^{(n)}$ is calculated based on the reliabilities of the subcode codewords in P . Denoting $\theta(a)$ as the reliability that the bit associated with $\beta_{m,k}^{(n)}$ equals $a \in \{0, 1\}$, we have

$$\theta(1) = \max_{\substack{t = 1, 2, \dots, 2^g \\ \mathbb{C}_t(l_{n,k}) = 1}} (R_t - \alpha_{n,k}^{(m)}) \quad (5)$$

$$\theta(0) = \max_{\substack{t = 1, 2, \dots, 2^g \\ \mathbb{C}_t(l_{n,k}) = 0}} (R_t). \quad (6)$$

It is also worth noting that if there is only a small number of subcode codewords in P , it is possible that for some particular positions in the subcode, the set $\{t : t = 1, 2, \dots, 2^g; \mathbb{C}_t(l_{n,k}) = a\}$ is empty for some a . In other words, the $l_{n,k}$ th bit in $\mathbb{C}_t$ is always "0" or always "1" for all $t = 1, 2, \dots, 2^g$. Under this scenario, one of Eq. (5) and Eq. (6) cannot be evaluated. Suppose the $l_{n,k}$ th bit in $\mathbb{C}_t$ is always "1" for all $t = 1, 2, \dots, 2^g$. To evaluate Eq. (6), we have to create an extra subcode codeword $\mathbb{C}_{ex}$ with its $l_{n,k}$ th bit equal "0". In the proposed method, we select from P the codeword with the largest reliability R_t . The $l_{n,k}$ th bit of the selected codeword is flipped from "1" to "0". We will further flip the least number of other bits such that a valid codeword can be found. The extra valid codeword, i.e., $\mathbb{C}_{ex}$, will subsequently be used in Eq. (6) to evaluate the value of $\theta(0)$. (A similar

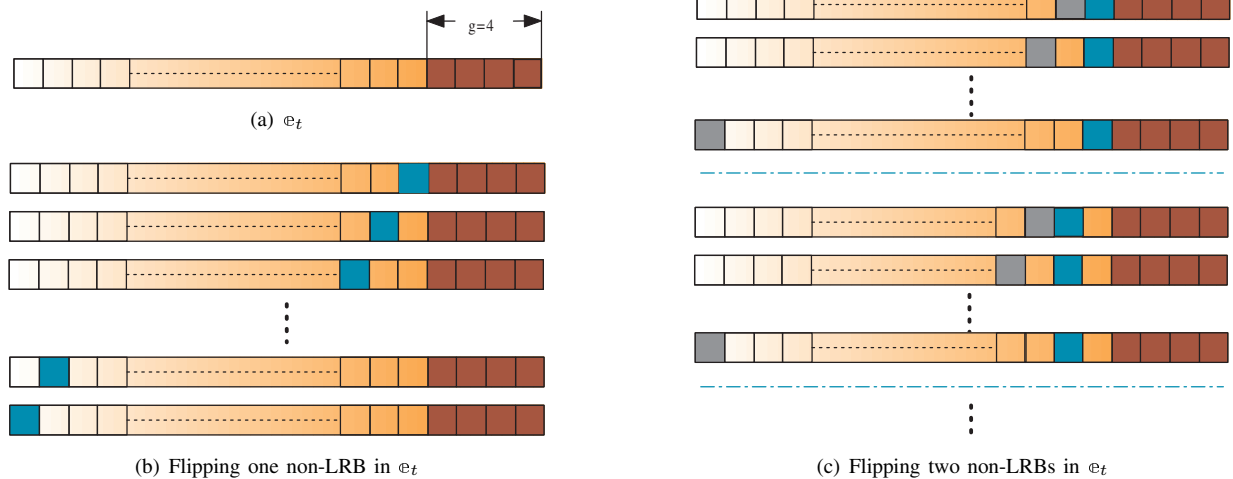


Fig. 1. An example of the flipping process in finding a decodable block for the subcode. (a) More reliable bits are illustrated with lighter colors and the LRBs are marked in dark red color. (b) Flipping one of the non-LRBs at a time to generate a set of code blocks, where the flipped bits are marked in blue color. (c) Flipping two non-LRBs (marked in grey and blue) at each time in e_t .

procedure can be used when a particular bit in e_t is always “0” for all $t = 1, 2, \dots, 2^g$). Having computed $\theta(1)$ and $\theta(0)$, the message $\beta_{m,k}^{(n)}$ is obtained using

$$\beta_{m,k}^{(n)} = \theta(1) - \theta(0). \quad (7)$$

After updating the reliability of each *bit* in the subcode, the reliabilities of the *symbols* in the q -ary codes can be easily determined from their binary representations.

3) *Variable-node updating* $\alpha_n^{(m)}$: Let C_n denote the set of check nodes incident to the n th variable node. The message vector transmitted from the n th variable node to the m th check node, i.e., $\alpha_n^{(m)}$, is updated using

$$\alpha_n^{(m)} = s_n + \sum_{j \in C_n/m} \beta_j^{(n)}. \quad (8)$$

4) *Tentative decoding*: An updated reliability for each bit in a symbol, denoted by $Q_{n,k}$, is computed using

$$Q_{n,k} = s_{n,k} + \sum_{j \in C_n} \beta_{j,k}^{(n)}. \quad (9)$$

The k th bit in the n th symbol, denoted by $w_{n,k}$, is then decoded according to the sign of $Q_{n,k}$. Based on the bit vector $(w_{n,1}, w_{n,2}, \dots, w_{n,p})$, the n th symbol, denoted by $w_n \in GF(q)$, can be further decoded. The decoded codeword, given by $\mathbf{w} = (w_1, w_2, \dots, w_N)$, is checked against the validity of the parity-check equation, i.e.,

$$\mathbf{w} \cdot \mathbf{H}^T = \mathbf{0}. \quad (10)$$

The iteration stops if the equation is satisfied. Otherwise the iteration process continues until Eq. (10) is satisfied or a pre-determined number of iterations have been executed.

C. Design of Subcode

In this section, we present the design criterion of the subcodes. Suppose the transmitted bits corresponding to the

TABLE I
NUMBER OF COMBINATIONS TO EVALUATE IN THE CHECK-NODE PROCESSING FOR A SINGLE EDGE FOR THE PROPOSED ALGORITHM, THE EMS DECODER, AND THE SELECTIVE MIN-MAX ALGORITHM.

Proposed Method	EMS Algorithm	Selective Min-Max
2^g	$\binom{d_r-1}{n_c} n_s^{n_c}$	$\left(\frac{q+1}{d_r-1}\right)^{d_r-1}$

variable nodes incident to a check node forms a codeword e' for the subcode. However, when the transmitted signals are corrupted by noise, the codeword has been determined as e after hard decisions are made at the decoder. Thus the error pattern is given by $\mathbf{y} = e' \oplus e$ and consequently the syndrome vector, denoted by $\mathbf{x}$, is given by

$$e \cdot \mathbf{S}^T = \mathbf{x}. \quad (11)$$

Hence, the number of different syndromes $\mathbf{x}$ determines the number of error patterns that an algebraic decoder can correct. To reduce the chance that an algebraic decoder decodes an input vector e_t into more than one subcode codeword, as mentioned in Sect. II-B2a, the matrix $\mathbf{S} = [\mathbf{H}_1^T \mathbf{H}_2^T \dots \mathbf{H}_{d_r}^T]$ should be designed in such a way that the number of distinct $\mathbf{x}$ can be maximized. In our simulations, we will follow this philosophy when designing the matrices $\mathbf{H}_i^T$ ($i = 1, 2, \dots, d_r$).

D. Complexity issue

The proposed decoding method aims to reduce the number of combinations in the check-node updating process, thus to accelerate the decoding process. The decoding complexity increases with g . When $g = pd_r$, the proposed algorithm is exactly the Min-Sum algorithm. The choice for g therefore offers an option to compromise between error performance and decoding simplicity.

1) *Another option in specifying $\{e_t\}$* : It is worth noting that apart from the process stated above, the proposed algorithm may also be implemented in a manner similar to

TABLE II
NUMBER OF OPERATIONS IN VARIOUS DECODING ALGORITHMS.

	Additions	Subtractions	Multiplications	Divisions	Comparisons
Proposed Method	$2^g Gp + d_c Gp$	$2^g Gp$	-	-	$2^g Gp$
FFT Algorithm	$2Gp$	$2G(q-1)$	$2Gp + 2Gq - Mq - Nq$	$2Gq$	-
Log-FFT Algorithm	$2Gp + 2Gq$	$2Gq$	-	-	$12Gp + 16Gq$
Min-Sum Algorithm	$Gq^{d_r-1}(d_r-1) + Gqd_c$	Gq^{d_r-1}	-	-	Gq^{d_r-1}
EMS Algorithm	$G\binom{d_r-1}{n_c} n_s^{n_c} d_r + Gqd_c$	-	-	-	$Gq^2 + G\binom{d_r-1}{n_c} qn_s^{n_c}$
Selective Min-Max	Gqd_c	-	-	-	$G\binom{q+1}{d_r-1} d_r^{d_r-1} (d_r-1)$

the selective Min-Max algorithm [8]. The process in forming a set of binary vector for the algebraic decoding may thus be modified as: i) search for the bit position in the subcode with the smallest reliability α , and denote $k = \lfloor \alpha \rfloor$; ii) identify all the bit positions with the integer parts of their reliabilities equal $k, k+1, \dots$, until the total number of identified bits is no less than g ; iii) flip the set of identified bits to form the binary vector set $\{\mathbf{e}_t\}$. In this manner, we may eliminate the need for sorting the all the bits in the subcode.

2) *Some discussions*: The proposed algorithm enjoys the flexibility in its complexity controlled through the parameter g . The proposed algorithm allows g bits in the subcode to flip between '1' and '0'. With a small g , it is possible that some of the d_r symbols take only one choice in $GF(q)$ in all the codewords formed in Section II-B2a. As an example, consider a code defined over $GF(32)$, with $d_r = 6$ (Code B in Section III); the subcode contains 30 bits, corresponding to 6 symbols. When $g = 4$, the number of symbols taking only one choice in the decoding process is at least 2 (the 4 bits are contained in 4 different symbols), at most 5 (the 4 bits are in a single symbol). The EMS decoder, on the other hand, controls the complexity through the parameter pair (n_s, n_c) [6]. It allows all the d_r symbols to take the n_s ($1 \leq n_s \leq q$) choices with largest reliabilities. The selective Min-Sum algorithm selects no less than $(q+1)$ elements from (d_r-1) variable nodes, and therefore has a fixed complexity. Table I summarizes the numbers of combinations in consideration during the three decoding algorithm for a single edge in the check-node processing. In Table I, G denotes the total number of connections in the Tanner graph, d_c is the column weight of $\mathbf{H}$, and (n_s, n_c) are the parameters in [6].

3) *Complexity comparison*: Table II summarizes the complexity of the proposed method against other decoding algorithms in terms of the number of computations required in a single iteration. It is observed that the proposed method contains no multiplication and division steps, which are required for the FFT algorithm. The transformations among the real domain, Log domain, and Fourier domain in the Log-FFT algorithm require a large number of table look-ups to complete. When compared with the Min-Sum and the EMS algorithm, the proposed approach requires fewer number of computations in each round of iteration. The proposed algorithm applies mainly additions whilst the selective Min-Max algorithm applies dominantly comparisons.

III. SIMULATION RESULTS

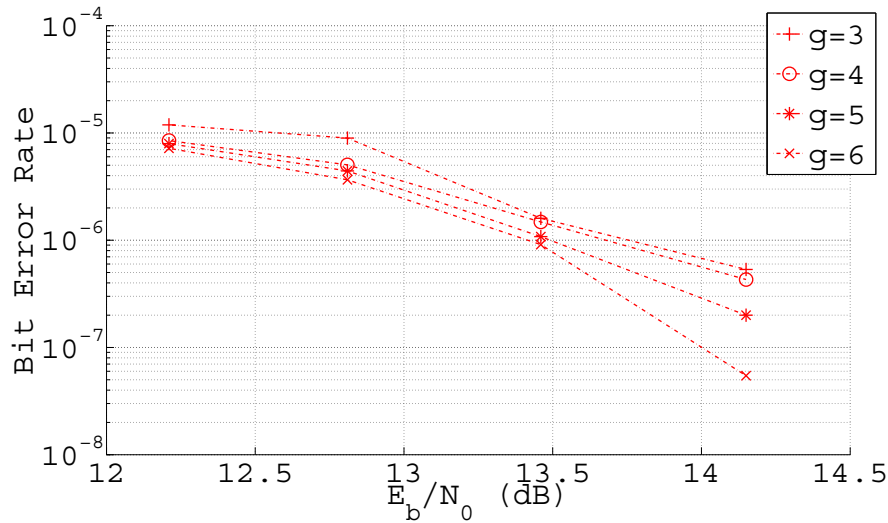
Two regular q-ary LDPC codes, Code A and Code B, have been simulated. Code A is a short-length code defined over $GF(16)$, with length 1,920 and code rate 1/3. The row weight is 4 which means that the subcode is a binary parity-check code with length 16 and code rate 1/4. Code B has a length of 20,000 with code rate 1/2. It is defined over $GF(32)$ and has a row weight of $d_r = 6$, leading to a binary parity-check subcode of length 30 and code rate 1/6. A BIAWGN channel is assumed. For both cases, the non-zero entries in each row of a parity-check matrix are selected² according to the criteria in Section II-C. The maximum number of decoding iterations is set at 50.

In the first set of simulations, a general estimation is made of the complexity difference among the proposed approach, the Min-Sum decoder and the EMS decoder by recording the decoding delay. 1,000 codewords are sent for each of Code A and Code B. The simulation time and the error performance of the proposed method is examined together with those of the Min-Sum decoder and the EMS decoder [6]. In Table III, the results show that the proposed approach can reduce the computation time substantially compared with the Min-Sum decoder and the EMS decoder. For Code A, the computation times of the proposed approach are only 21% to 43% of those needed by the EMS decoder; whereas the computation times of the Min-Sum decoder are several times those of the EMS decoder. For Code B, the speed improvement of the proposed approach is even more impressive, requiring only 1% to 9.7% of the computation times spent by the EMS decoder. No computation times have been recorded for the Min-Sum decoder because it takes unrealistically long to complete. Note that the proposed approach suffers from a degradation in error performance when g is too small.

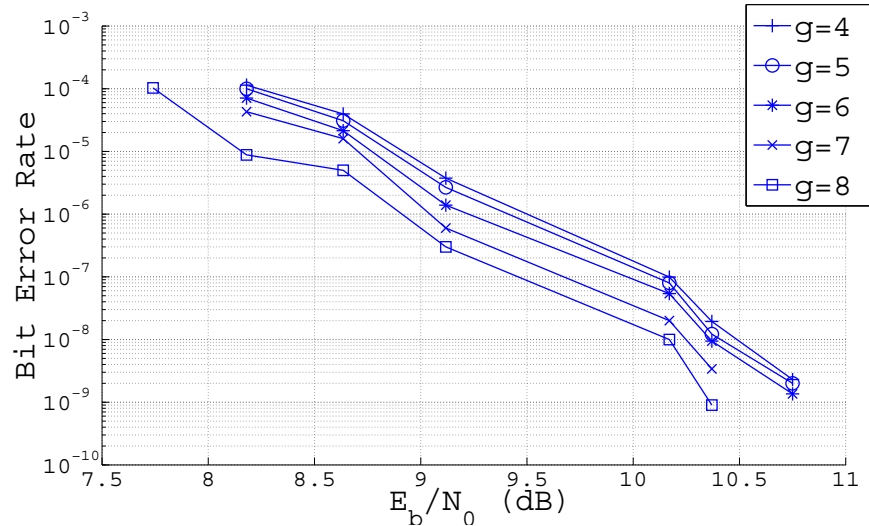
As indicated in Table III, the proposed algorithm clearly compromises error performance for simplicity. However, it can be observed that when E_b/N_0 is large, our algorithm works comparably with the EMS decoder with a much lower complexity. Therefore, in the high SNR region, we offer a fast method in evaluating the error performance of the codes.

We further simulate the bit error rates (BERs) of the two codes using our proposed approach, as demonstrated in Fig. 2. The results in Fig. 2 indicate that for both codes, a larger g

²For simplification, in our simulations, each row contains exactly the same set of non-zero entries. In other words, check nodes of a q-ary LDPC code are transformed into the same subcode in our simulations.



(a) Bit error rate of Code A with different choices of g



(b) Bit error rate of Code B with different choices of g

Fig. 2. Error performance of Code A and Code B under the proposed decoding algorithm

has led to a better error performance; e.g. in Fig. 2(a), the decoder for $g = 6$ outperforms that for $g = 3$ with more than 0.3dB, and in Fig. 2(b), $g = 8$ improves the BER by approximately 0.5dB from $g = 4$. As illustrated in Table I, the number of combinations considered in check-node updating increases exponentially with g , leading to lower BERs. In a practical implementation, it is favorable to have a dynamic choice for the parameter g , i.e., the decoder may choose to increase g if the E_b/N_0 is small, and vice versa.

It is also worth noting that Code B outperforms Code A with the same g values. Our algorithm may be used to compare various codes in terms of error performance, which is most useful when searching for the optimal codes.

IV. CONCLUSION

The paper proposes a decoding method for q-ary LDPC codes with a primary target of speeding up the decoding process. The proposed approach is based on the subcode concept and the decoding speed improvement is achieved with the help of the algebraic decoder of the subcodes. The algorithm offers another scheme to achieve the tradeoff between decoding complexity and the error performance.

The method demonstrated a significant improvement in decoding time with a moderate error-performance loss. It has been shown that the computation time can be reduced to a few percentages of that spent by an EMS decoder. Furthermore, the approach proposed may be applied to general non-binary LDPC codes with their subcodes defined.

TABLE III

DECODING TIME AND ERROR PERFORMANCE OF THE PROPOSED APPROACH, THE ORIGINAL MIN-SUM DECODER AND THE EMS DECODER. 1,000 CODE BLOCKS ARE SENT FOR EACH CODE OVER AN AWGN CHANNEL. T : COMPUTATION TIME IN SECONDS; t : NORMALIZED COMPUTATION TIME; N : NUMBER OF ERROR BLOCKS.

Code A	Method	g	SNR=14.1 dB			SNR=13.4 dB		
			T	t	N	T	t	N
	Proposed Approach	4	18	0.25	3	29	0.21	7
		5	21	0.29	1	45	0.32	7
		6	27	0.37	0	61	0.43	3
	Min-Sum	-	541	7.4	0	1125	8.0	0
	EMS	-	73	1	0	141	1	0
Code B	Method	g	SNR=11.1 dB			SNR=10.5 dB		
			T	t	N	T	t	N
	Proposed Approach	4	608	0.010	0	649	0.097	1
		5	1006	0.017	0	1083	0.015	1
		6	1887	0.032	0	1849	0.026	0
	Min-Sum	-	-	-	-	-	-	-
	EMS	-	59016	1	0	71052	1	0

ACKNOWLEDGMENT

The work described in this paper was supported by a grant from the Hong Kong Polytechnic University (Project No. G-YL22) and by the National Natural Science Foundation of China (Grant No. 60972037).

REFERENCES

- [1] M. C. Davey, *Error-Correction Using Low-Density Parity-Check Codes*. Cambridge University, 1999.
- [2] X. Zheng, F. C. M. Lau, and C. K. Tse, "Constructing Short-Length Irregular LDPC Codes with Low Error Floor," *IEEE Trans. Commun.*, vol. 58, no. 10, pp. 2823–2834, Oct. 2010.
- [3] W. M. Tam, F. C. M. Lau, and C. K. Tse, "A class of QC-LDPC codes with low encoding complexity and good error performance," *IEEE Commun. Lett.*, vol. 14, no. 2, pp. 169–171, Feb. 2010.
- [4] M. C. Davey and D. J. C. Mackay, "Low Density Parity Check Codes over $GF(q)$," *IEEE Communication Letters*, vol. 2, no. 6, pp. 165–167, 1998.
- [5] H. Wymeersch, H. Steendam, and M. Moeneclaey, "Log-Domain Decoding of LDPC Codes over $GF(q)$," in *Proc. IEEE International Conference on Communications*. Paris, France, June 2004, pp. 772–776.
- [6] D. Declercq and M. Fossorier, "Decoding Algorithm for Nonbinary LDPC Codes over $GF(q)$," *IEEE Trans. Comm.*, vol. 55, no. 4, pp. 633–643, 2007.
- [7] X. H. Shen and F. C. M. Lau, "Q-ary LDPC Decoder with Euclidean-distance-based sorting criterion," *IEEE Commun. Lett.*, vol. 14, no. 5, pp. 444–446, May 2010.
- [8] V. Savin, "Min-Max decoding for non binary LDPC codes," in *IEEE International Symposium on Information Theory*. Toronto, Canada, July 2008, pp. 960–964.
- [9] R. M. Tanner, "A Recursive Approach to Low Complexity Codes," *IEEE Trans. Inform. Theory*, vol. 27, no. 5, pp. 533–547, 1981.
- [10] Y. Min, F. C. M. Lau, and C. K. Tse, "Generalized LDPC Code With Single-Parity-Check Product Constraints At Super Check Nodes," in *Proc., The 7th International Symposium on Turbo Codes and Iterative Information Processing*. Gothenburg, Sweden, 2012.
- [11] D. Chase, "Class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. Inform. Theory*, vol. 18, no. 1, pp. 170–182, Jan. 1972.
- [12] M. Fossorier and S. Lin, "Error performance analysis for reliability-based decoding algorithms," *IEEE Trans. Inform. Theory*, vol. 48, no. 1, pp. 287–293, Jan. 2002.
- [13] C. Poulliat, M. Fossorier, and D. Declercq, "Design of Non-Binary LDPC Codes Using Their Binary Image: Algebraic Properties," in *Proc. International Symposium on Information Theory*. Seattle, USA, 2006, pp. 93–97.