

# Symbol-Flipping Based Decoding of Generalized Low-Density Parity-Check Codes Constructed over $GF(q)$

Fang-Chun Kuo and Lajos Hanzo

School of Electronics and Computer Science, University of Southampton, SO17 1BJ, UK, lh@ecs.soton.ac.uk

**Abstract**—An efficient symbol-flipping based decoding algorithm designed for nonbinary Generalized Low-Density Parity-Check (GLDPC) codes is proposed. By extending the concept of the Weighted Bit Flip Voting (WBFV) algorithm designed for binary Hamming-code based GLDPC codes, the symbol-flipping decoding algorithm can be beneficially employed for decoding the family of GLDPC codes constructed from nonbinary constituent codes, such as nonbinary Bose Chaudhuri Hocquenghem (BCH) codes or Reed Solomon (RS) codes. The simulation results demonstrate that improvements of 1 dB and 2.7 dB are achieved by the proposed coding scheme in comparison to the more conventional binary GLDPC codes using the WBFV decoding algorithm, when using the Galois Field  $GF(32)$  for communicating over AWGN and uncorrelated Rayleigh fading channels, respectively.

## I. INTRODUCTION

Low-Density Parity-Check (LDPC) codes were originally devised by Gallager [1] in the early 1960s, but have not been exploited in practice until the 1990s [2], [3]. During the past decade, however, LDPC codes have received substantial research attention as a benefit of their excellent performance and hence numerous new developments have taken place in this area. As an evolution of classic LDPC codes [1], Generalized Low-Density Parity-Check (GLDPC) codes were introduced by Tanner [4] and then Hamming-code based GLDPC codes were further explored by Boutros *et al.* [5] as well as by Lentmaier and Zigangirov [6]. GLDPC codes are constructed by replacing each single parity check of regular LDPC codes with the parity check matrix of a small linear block code referred to as the constituent code. It has been shown that Hamming-code based GLDPC codes are asymptotically good in the sense of minimum distance and exhibit an excellent performance over both AWGN and Rayleigh channels [5]–[7]. Pothier [8] also demonstrated that GLDPC codes can be considered as a generalization of product codes and as a benefit of their higher flexibility in terms of the selection of code length, GLDPC codes constitute a promising design alternative to replace product codes in many applications, such as digital audio and TV broadcasting, high speed packet data transmission and deep space applications. Furthermore, the GLDPC decoder of a Hamming-code based scheme has a regular parallel structure, which renders them amenable to systolic array based practical integrated circuit (IC) implementations.

In this paper, we investigate the attainable performance of symbol-based hard decision decoding algorithms designed for GLDPC codes constructed over  $GF(q)$ , since there is

a paucity of results on GLDPC codes employing nonbinary constituent codes. By contrast, binary constituent codes have more often been used for constructing GLDPC codes [9]–[14]. Moreover, perhaps the best known classic codes are the maximum-minimum-distance nonbinary Reed Solomon (RS) codes, which are used in numerous standards, such as the Digital Audio Broadcast (DAB) and Digital Video Broadcast (DVB) schemes or in Compact Disc (CD) players. It might therefore be worth investigating, how RS codes behave when they are embedded in GLDPC coding schemes. A particular further advantage of GLDPC codes is that their iterative decoding is based on the decoding of modest-complexity constituent codes, hence the total decoding complexity may be expected to be low.

For symbol-based hard decision decoding, a symbol-flipping algorithm is considered, which may be considered to be an extension of the bit-flipping algorithm [1], [11], [15]. A simple bit flipping scheme was originally proposed by Gallager for LDPC codes [1]. Based on the appealing conceptual and implementational simplicity of the bit-flipping algorithm, the Weighted Bit-Flipping (WBF) algorithm was developed in [15] for the sake of achieving an improved performance by exploiting some bit-reliability information, which results in an attractive tradeoff between the achievable performance and the decoding complexity imposed [15]. The concept of bit flipping algorithms using votes [11] was generalised for employment in GLDPC codes using binary Hamming constituent codes and hence it was termed as Weighted Bit Flip Voting (WBFV) [11]. Based on the philosophy of the WBFV algorithm developed for binary Hamming-code based GLDPC codes, here we propose a symbol flipping algorithm for employment in nonbinary GLDPC codes. Similar to the WBFV algorithm of [11], the error correcting capability of the constituent codes is exploited for more accurately determining the position of the least reliable symbols. However, in the context of the symbol-flipping algorithm, not only the error positions, but also the  $(q - 1)$  legitimate error magnitudes have to be evaluated. This can be achieved, if the classic algebraic decoders of the nonbinary constituent codes of the GLDPC codes are applied. In each decoding iteration, the least reliable symbols are corrected according to the error magnitude provided by the algebraic decoder of the nonbinary constituent code. We will provide simulation results to demonstrate that symbol-flipping algorithms can be successfully employed for the decoding of nonbinary GLDPC codes. We will also demonstrate that

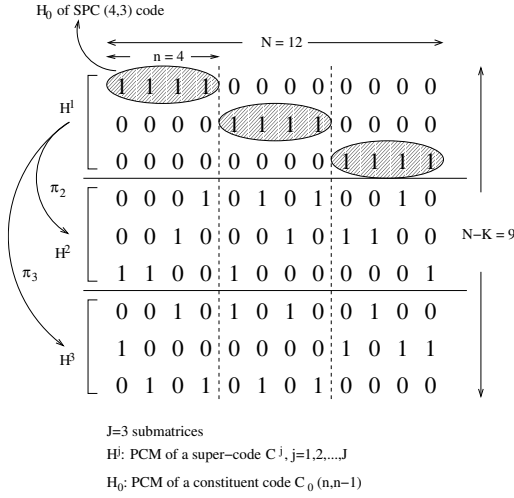


Fig. 1. Parity-check matrix  $H$  of an  $R = K/N = 1/4$ -rate LDPC (12,3) code having  $J = 3$  levels, which uses the single parity check code SPC(4,3) as its constituent code.

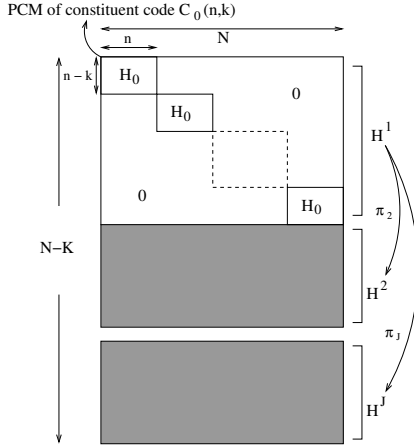


Fig. 2. Parity-check matrix  $H$  of a GLDPC  $(N, K)$  code

the proposed coding scheme results in an improved error rate performance in comparison to binary GLDPC codes using the WBFV decoding algorithm of [11], when communicating over both AWGN and uncorrelated Rayleigh fading channels.

The remainder of this paper is organized as follows. In Section II, a brief review of GLDPC codes is provided. The symbol-flipping algorithm proposed for nonbinary GLDPC codes is outlined in Section III. Our simulation results are presented in Section IV and, finally, our conclusions are offered in Section V.

## II. THE STRUCTURE OF GLDPC CODES

In this section, the structure of GLDPC codes is introduced [8]. There are two appealing ways of describing the structure of GLDPC codes. The first one is based on the construction of the Parity Check Matrix (PCM), which may be interpreted as an extension of the PCM of LDPC codes [1], while the other is based on the concept of Tanner graphs [4].

### A. Description of the Parity-Check Matrix

Consider the example shown in Fig. 1, portraying the PCM  $H$  of size  $9 \times 12$  of a classic  $R = K/N$ -rate LDPC  $(N, K)$

code [1] using the parameters  $N = 12, K = 3$  and  $J = 3$ , where  $H$  is constructed by concatenating  $J = 3$  submatrices, namely  $H^1, H^2$  and  $H^3$ . The three submatrices  $H^1, H^2$  and  $H^3$  seen in Fig. 1 are of dimension  $3 \times 12$ , which are the PCMs of the super-codes  $C^1, C^2$  and  $C^3$ , respectively, that are constructed from the single parity check (SPC) codes  $(n, n - 1)$ . More explicitly, the first submatrix  $H^1$  seen in Fig. 1 is a *block diagonal matrix* having the matrix elements  $H_0$ , which constitutes the PCMs of the SPC( $n, n - 1$ ) codes associated with  $n = 4$  along its main diagonal. Accordingly, each group of 4 bits constituting a codeword of the super-code  $C^1$  is only related to a SPC(4,3) code. Therefore, the super-code  $C^1$  is constituted by the direct concatenation of  $N/n = 3$  number of SPC(4,3) codes, which are hence referred to as the constituent codes  $C_0$  seen in the top third of Fig. 1. All the other submatrices, namely  $H^2$  and  $H^3$  are formed by the pseudo-random permutation of all the columns of the submatrix  $H^1$  without interleaving the elements of the columns. This operation is formulated as  $H^j = \pi_j(H^1)$  for  $j = 2, 3$ , explicitly indicating that the super-codes  $C^2, C^3$  are constructed by random interleaving the super-code  $C^1$ . The codeword  $C$  is the intersection, i.e. the common symbols of the super-codes  $C^1, C^2$  and  $C^3$ . More explicitly, the codeword  $C$  of the LDPC  $(N, K)$  code should be checked by the PCM  $H$ , which is the concatenation of the  $J = 3$  PCMs of the super-codes  $C^1, C^2$  and  $C^3$ , therefore we have  $C \cdot H^1 = C \cdot H^2 = C \cdot H^3 = 0$ .

This example of a classic LDPC code can be generalized for the sake of constructing GLDPC codes. The SPC  $(n, n - 1)$  code is used as the constituent code, when constructing classic LDPC codes, while a more general class of  $(n, k)$  block codes may be used as constituent codes, when constructing GLDPC codes. As illustrated in Fig. 2, the matrix  $H_0$  of dimension  $(n - k) \times n$  is the PCM of a constituent code  $C_0(n, k)$ . The first submatrix  $H^1$  of the super-code  $C^1$  portrayed in the top segment of Fig. 2 produces the direct concatenation of  $N/n$  number of constituent codes  $C_0(n, k)$  according to [8]. Hence we have  $C^1 = \bigoplus_{l=1}^{N/n} C_0$ , where  $N$  is the codeword length of the GLDPC  $(N, K)$  code and  $n$  is the codeword length of the constituent code  $C_0(n, k)$ . Finally, the PCM  $H$  of the GLDPC  $(N, K)$  code is constructed from the concatenation of the  $J$  number of submatrices  $(H^1 \cdots H^J)$ , which are the PCMs of the super-codes  $(C^1 \cdots C^J)$ , respectively. Thus, a GLDPC  $(N, K)$  code may be viewed as the intersection  $C = \bigcap_{j=1}^J C^j$  of the  $J$  super-codes [8]. Furthermore, since the submatrices  $H^2, \dots, H^J$  are derived by interleaving the columns of the first submatrix  $H^1$ , the codewords of the super-codes  $C^j, j \in \{2 \cdots J\}$  are constituted by random permutations of the codewords of the super-code  $C^1$ , which is expressed as  $C^j = \pi_j(C^1)$ , where  $\pi_j$  represents the corresponding symbol-interleaver.

### B. Graphical Concept

Fig. 3 portrays the bipartite graph of the classic LDPC (12,3) code [1] defined in Fig. 1. As shown in Fig. 3, the upper part contains  $N = 12$  codeword symbol nodes, while the lower

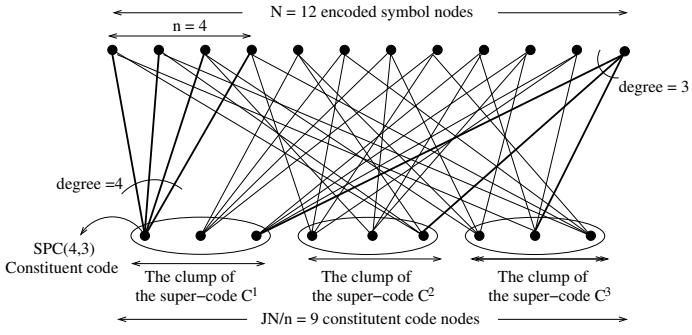


Fig. 3. The bipartite graph of the LDPC (12,3) code using the PCM of Fig. 1.

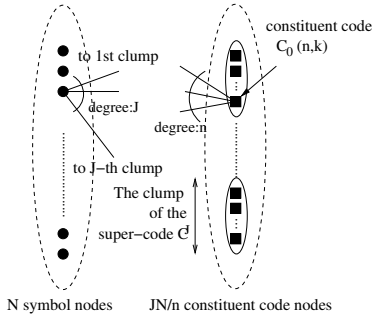


Fig. 4. The bipartite graph of the GLDPC  $(N, K)$  code.

part represents  $J \times N/n = 9$  constituent code nodes. The 9 constituent codes are simple binary SPC  $(4, 3)$  codes. An edge between a symbol node and a constituent code node indicates that the corresponding symbol belongs to particular constituent code. Therefore, a 12-symbol word included in the upper part is a codeword of the resultant LDPC codes, if and only if the 9 lower nodes of 4 incoming symbols belong to the SPC  $(4, 3)$  codes. Furthermore, the  $J = 3$  edges stemming from every single symbol node are connected to specific constituent code nodes belonging to different so-called clumps [8] of super-codes  $C^1, \dots, C^3$ . We can also see in Fig. 3 that the degrees of the symbol node and constituent code node are always  $J = 3$  and  $n = 4$ , respectively, which are defined as the number of the super-codes and the codeword length of the constituent code  $C_0(n, k)$ .

Similar to the classic LDPC code of Fig. 3, the graphical representation of the GLDPC  $(N, K)$  code is depicted in Fig. 4. The SPC  $(n, n-1)$  code is replaced by a more general  $(n, k)$  block code used as the constituent code  $C_0$ . The GLDPC code's length is  $N$  symbols, which is the number of symbol nodes in the left part of Fig. 4. The right part of Fig. 4 holds  $J \times N/n$  constituent code nodes. The degree of the constituent code node is equal to  $n$ , while the constituent code itself is defined as  $C_0(n, k)$  in Fig. 4. The degree of the symbol nodes is  $J$ , which implies that every symbol node is connected to  $J$  constituent codes represented by  $J$  different clumps of super-codes  $C^j, j \in \{2 \dots J\}$ , respectively. In other words, every symbol is determined by the  $J$  constituent codes, which belong to  $J$  super-codes, respectively.

### C. Summary

The code rate of a GLDPC code can be lower-bounded by observing its parity-check matrix as [8]:

$$R = K/N \geq 1 - J(1 - r_0), \quad (1)$$

where  $r_0$  denotes the code rate of the constituent code  $C_0(n, k)$ , where the equality holds, when the parity-check matrix  $H$  has full rank, i.e. when all of its rows are independent. In practice,  $\pi_j$  is chosen at random, but by avoiding that two (or more) symbol nodes are connected to the same  $J$  constituent code nodes, since this would create short cycles of length 4 in the Tanner graph [8]. Moreover, GLDPC codes having  $J = 2$  levels have the highest possible code rate [8] as well as a low-complexity decoder structure, which are desirable properties in practical applications. Thus, in our study, we consider only GLDPC codes having  $J = 2$  levels. Note that for practical  $(N, 2, n)$  GLDPC codes, it is only possible to construct a meritorious PCM  $H$  in which no undesirable short cycles of length 4 appear in the graph, if we have  $N/n \geq n$  [16]. Therefore, the  $(N, 2, n)$  GLDPC codes should always satisfy  $N/n \geq n$ .

### III. SYMBOL-FLIPPING BASED DECODING ALGORITHM

Given the definition of the GLDPC codes in terms of  $J$  number of interleaved super-codes [5]–[7], the decoding philosophy of the  $J = 2$ -level GLDPC  $(N, K)$  code is similar to that of the product code, where every symbol of the GLDPC  $(N, K)$  codeword could be decoded by two constituent codes, which belong to two independent super-codes, respectively [5]–[7]. Accordingly, the WBFV [11] algorithm decodes the GLDPC codes using an iterative method, in which a hard decision decoder is applied by each constituent code  $C_0(n, k)$  and passes back  $n$  individual votes to the symbols it covers. The magnitude of a vote indicates how reliable a constituent code node considers the current symbol's value to be. Here we extend the concept of the WBFV algorithm designed for the binary Hamming-code based GLDPC codes [11] to symbol-flipping and invoke it for the GLDPC codes using either nonbinary BCH or RS constituent codes. Again, in the symbol-flipping algorithm not only the error positions, but also the error magnitudes defined over  $GF(q)$  have to be evaluated, where the algebraic decoders of the nonbinary constituent codes provide both the error positions as well as the error magnitudes.

For the  $J = 2$ -level nonbinary GLDPC construction employed, all  $GF(q)$  symbols will belong to both of the  $(n, k)$  constituent codes in the super-code  $C^1$  and  $C^2$ , respectively. Fig. 5 shows the symbols' vote generation process for the GLDPC  $(21, 9)$  code constructed over  $GF(8)$  using the constituent code RS  $(7, 5)$ . Note that for the sake of plausible and straightforward explanation, the GLDPC code considered in Fig. 5 was constructed without avoiding the short cycles of length 4. The votes concerning the specific values of the symbol are generated by the Peterson-Gorenstein-Zierler (PGZ) or the Berlekamp-Massey (BM) [17] hard decision decoders (HDDs) of the RS constituent codes in the super-code

$C^1$  and  $C^2$ . In nonbinary HDDs, owing to the limited error-correction capability of the RS code, a *decoding failure* occurs, when the corrupted codeword is not within the so-called decoding sphere of a valid codeword. Hence, for the nonbinary algebraic RS constituent decoders used in the symbol-flipping algorithm, there are three possible decoding scenarios, which are also featured in Fig. 5 and discussed below:

- All-zero syndromes: this implies the presence of a valid ( $V$ ) codeword, hence all the  $n$  constituent RS code symbols are labelled by the character  $V$ .
- Non-zero syndromes and decoding success: this indicates the presence of an invalid but correctable received word. Since successful decoding took place, the corresponding error positions are labelled with  $E$  indicating that the symbols are in error. By contrast, the correct symbol label  $e$  is assigned to all other symbols.
- Non-zero syndromes and decoding failure: no error positions were identified owing to decoding failure, therefore no corrective action may be carried out and no useful information may be gleaned from this decoder. Hence all the  $n$  RS code symbols of the codeword are also labelled by  $e$ , similarly to the correctly received symbols of the correct decoding scenario.

To elaborate a little further, the various votes  $V$ ,  $e$ ,  $E$  are given numerical values so that the votes arriving from the  $J = 2$  decoders for all the  $N = 21$  symbols, which are either  $VV$ ,  $eV$ ,  $EV$ ,  $ee$ ,  $eE$  or  $EE$ , may be ranked in terms of their reliability according to the sum of the  $J = 2$  constituent votes. It was suggested in [11] that using the weight of  $V = 0$ ,  $e = 1$  and  $E = 2$  is capable of producing the lowest possible BERs, where having higher weights represents a lower reliability. Table I shows the weights of the various vote pairs adopted from Hirst and Honary [11] for our symbol-flipping algorithm. Noting that in our future research, we intend to specifically optimize the weights for nonbinary RS constituent codes operating over different GFs. As depicted in Fig. 5, a vote pair  $EE'$  indicates that different error magnitudes were suggested by the  $J = 2$  constituent decoders. In this case, we will randomly opt for the error magnitude suggested by one of the two constituent codes and re-calculate the corresponding weight, which has to satisfy  $EE > EE' > Ee$ . Accordingly, in each decoding iteration, after ranking the vote weight of each symbol, the symbols deemed to be least reliable by the vote weights or confidence measures are then corrected according to the error magnitudes determined by the PGZ or BM HDDs.

Based on the above elaborations, the symbol-flipping aided decoding algorithm designed for the family of the GLDPC codes using nonbinary RS constituent codes is summarized as follows:

- 1) Invoke the PGZ or BM HDDs of the nonbinary RS constituent codes, which belong to the super-code  $C^1$  and  $C^2$ , respectively.
- 2) Compute the vote weight for each of the  $N$  symbols according to Table I for the sake of quantifying the

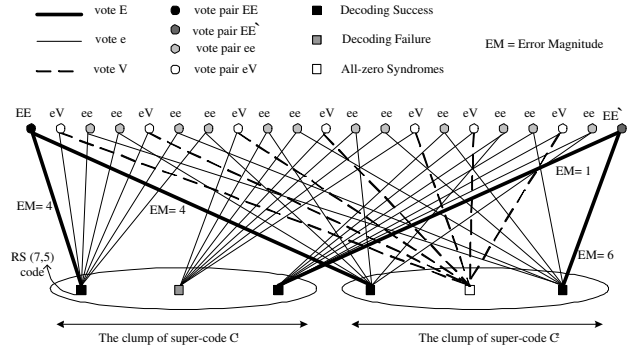


Fig. 5. Example of the voting process for the  $J = 2$ -level GLDPC (21, 9) code constructed over  $GF(8)$  using the constituent RS(7, 5) codes in one iteration. The 21 encoded symbol nodes are seen in the upper part, while the 6 constituent code nodes are portrayed in the lower part of the figure.

TABLE I

VOTE WEIGHTS FOR THE SYMBOL-FLIPPING BASED DECODING ALGORITHM FOR THE NONBINARY GLDPC CODES [11].

$V = 0, e = 1, E = 2$						
Vote pair	$EE$	$EE'$	$Ee$	$eV, ee$	$eV$	$VV$
Vote weight	4	3.5	3	2	1	0

reliability of all symbols.

- 3) Rank the symbols according to their reliability based on their vote weights.
- 4) Correct the lowest reliability symbols, i.e. those having the maximum vote weights, according to the error magnitudes. If the vote pair assumes the values of  $ee$  or  $eV$ , the corresponding symbol will not be corrected, since no valid error magnitude was calculated.
- 5) Repeat steps (1) to (4). This process of symbol flipping continues, until we either arrive at the vote pair  $VV$  for all the  $N$  symbols or the maximum affordable number of GLDPC iterations is reached.

#### IV. SIMULATION RESULTS

Five nonbinary GLDPC codes defined over  $GF(32)$  were employed in our simulations using BPSK modulation for communicating over both AWGN and uncorrelated Rayleigh

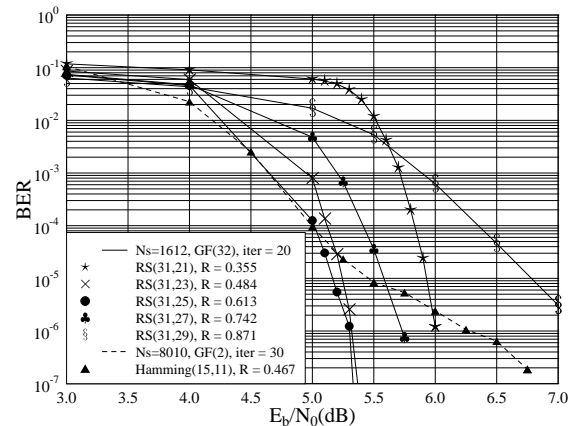


Fig. 6. BER performance of GLDPC codes using both linear Hamming and RS constituent codes. BPSK modulation is used when communicating over an AWGN channel. The codeword length is around 8000 bits for all the codes of Table II.

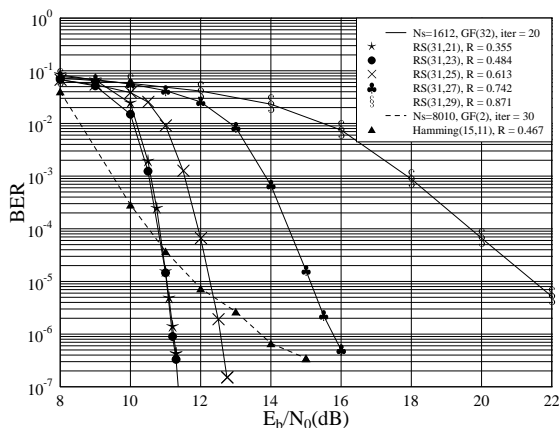


Fig. 7. BER performance of GLDPC codes using both linear Hamming and RS constituent codes. BPSK modulation is used when communicating over an uncorrelated **Rayleigh fading channel**. The codeword length is around 8000 bits for all the codes of Table II.

TABLE II

CODING GAIN AT A TARGET BER OF  $10^{-5}$ , EXTRACTED FROM THE RESULTS SHOWN IN FIG. 6 AND FIG. 7. BPSK MODULATION IS USED, WHEN COMMUNICATING OVER BOTH AWGN AND UNCORRELATED RAYLEIGH FADING (URF) CHANNELS.

Constituent code	Rate	Coding gain	
		AWGN	URF
RS(31,21)	0.355	3.7 dB	33.0 dB
RS(31,23)	0.484	4.2 dB	33.0 dB
RS(31,25)	0.613	4.3 dB	31.8 dB
RS(31,27)	0.742	4.0 dB	28.9 dB
RS(31,29)	0.871	2.8 dB	22.8 dB

fading channels. All codes had a codeword length of  $N = 1612$  5-bit symbols defined over  $GF(32)$  (8060 bits), and the RS codes (31, 21), (31, 23), (31, 25), (31, 27), (31, 29) were used as our constituent codes, which have error correcting capabilities  $t = 5, 4, 3, 2$  and 1 5-bit symbols, respectively. It can be observed in Fig. 6 that as expected, the GLDPC code using the RS (31,25) constituent code achieves the highest coding gain at a BER of  $10^{-5}$ , when communicating over an AWGN channel. On the other hand, the best constituent code for the GLDPC codes defined over  $GF(32)$  for transmission over an uncorrelated Rayleigh fading channel was seen in Fig. 7 to be the RS (31, 23) code, since it had the highest coding gain, outperforming the RS (31, 21) code, although the latter one had the edge over the AWGN channel. A range of further conclusions can be drawn from Table II.

Our GLDPC coding scheme defined over  $GF(32)$  was also benchmarked against a binary GLDPC code using the (15, 11) Hamming constituent codes, as shown in both Fig. 6 and Fig. 7. At the BER of  $10^{-6}$ , the GLDPC codes using the RS (31, 21) constituent code of rate  $R = 0.484$  has an  $E_b/N_0$  improvement of 1 dB in comparison to the binary GLDPC code employing the (15, 11) Hamming constituent code based scheme having a similar rate of  $R = 0.467$ , when communicating over an AWGN channel. By comparison, in uncorrelated Rayleigh fading channels an  $E_b/N_0$  improvement of 2.7 dB was achieved in comparison to the Hamming-code based benchmark scheme.

## V. CONCLUSIONS

In this paper, an efficient symbol-based hard decision aided decoding algorithm designed for nonbinary GLDPC codes was proposed. The symbol-flipping decoding algorithm evolving from the WBFV algorithm of [11] can be successfully used for decoding GLDPC codes constructed from nonbinary constituent codes. The simulation results demonstrated that GLDPC codes defined over  $GF(q)$  have the potential of outperforming similar-rate binary constituent codes. Improvements of 1 dB and 2.7 dB were achieved for a code rate around 0.47, when using  $GF(32)$  for communicating over AWGN and uncorrelated Rayleigh fading channels, respectively. Furthermore, the nonbinary GLDPC codes using the proposed symbol-flipping algorithm have exhibited no error floor. In our future research, we will consider the employment of the proposed codes in sophisticated system studies using unequal-protection multilevel codes and Fountain codes.

## REFERENCES

- [1] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA: MIT Press, 1963.
- [2] M. Sipser and D. A. Spielman, "Expander codes," *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1710 – 1722, 1996.
- [3] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Transactions on Information Theory*, vol. 45, no. 2, pp. 1710–1722, Mar. 1999.
- [4] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, pp. 533 – 547, Sept. 1981.
- [5] J. Boutros, O. Pothier, and G. Zemor, "Generalized low density (Tanner) codes," in *IEEE International Conference on Communications (ICC 1999)*, vol. 1, June 1999, pp. 441 – 445.
- [6] M. Lentmaier and K. S. Zigangirov, "On generalized low-density parity-check codes based on Hamming component codes," *IEEE Communication Letters*, vol. 3, no. 8, pp. 248 – 250, Aug. 1999.
- [7] O. Pothier, L. Brunel, and J. Boutros, "A low complexity FEC scheme based on the intersection of interleaved block codes," in *Vehicular Technology Conference*, 1999, pp. 274–278.
- [8] O. Pothier, "Compound codes based on graphs and their iterative decoding," Ph.D. thesis, Ecole Nationale Supérieure des Telecommunications, Jan. 2000, <http://www.comelec.enst.fr/boutros/coding/>.
- [9] T. Zhang and K. Parhi, "A class of efficient encoding generalized low-density parity-check codes," in *2001 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '01)*, vol. 4, May 2001, pp. 2477 – 2480.
- [10] —, "High-performance, low-complexity decoding of generalized low-density parity-check codes," in *IEEE Global Telecommunications Conference*, vol. 1, Nov. 2001, pp. 181 – 185.
- [11] S. Hirst and B. Honary, "Decoding of generalised low-density parity-check codes using weighted bit-flip voting," *IEE Proceedings Communications*, vol. 149, no. 1, pp. 1 – 5, Feb. 2002.
- [12] —, "Application of efficient Chase algorithm in decoding of generalized low-density parity-check codes," *IEEE Communications Letters*, vol. 6, no. 9, pp. 385 – 387, Sept. 2002.
- [13] T. M. N. Ngatched and F. Takawira, "Efficient decoding of generalized low-density parity-check codes based on long component codes," in *2003 IEEE Wireless Communications and Networking*, vol. 1, March 2003, pp. 705 – 710.
- [14] I. Djordjevic, O. Milenkovic, and B. Vasic, "Generalized low-density parity-check codes for optical communication systems," *Journal of Lightwave Technology*, vol. 23, no. 5, pp. 1939 – 1946, May 2005.
- [15] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity-check codes based on finite geometries," *IEEE Transactions on Information Theory*, vol. 47, pp. 2711–2736, Nov. 2001.
- [16] T. Zhang, "Efficient VLSI Architectures for Error-Correction Coding," Ph.D. thesis, University of Minnesota, July 2002, <http://www.ecse.rpi.edu/homepages/tzhang/RVSAL/research.html>.
- [17] L. Hanzo, T. Liew, and B. L. Yeap, *Turbo Coding, Turbo Equalisation and Space-Time Coding*. John Wiley & Sons Ltd., 2002.