Near Maximum-Likelihood Decoding of Generalized LDPC and Woven Graph Codes

Irina E. Bocharova, Boris D. Kudryashov, and Nikolay I. Makarov
Dept. of Information Systems
St.-Petersburg Univ. of Information Technologies, Mechanics and Optics
St.-Petersburg, Russia
Email: irina@eit.lth.se, boris@eit.lth.se, and nmakarov@rain.ifmo.ru

Rolf Johannesson
Dept. of Electrical and Information Technology
Lund University
Lund, Sweden
Email: rolf@eit.lth.se

***Abstract*—Relations between Generalized LDPC codes, nonbinary LDPC codes, and woven graph codes are considered. Focus is on rather short codes suitable, for example, for coding control signaling information in mobile communications. In particular, codes of lengths less than 200 bits are studied. Low-complexity near maximum-likelihood (ML) decoding for these classes of codes is introduced and analyzed. Frame error rate (FER) performance of the new decoding procedure is compared with the same performance of ML and belief propagation (BP) decoding. It is shown that unlike BP decoding whose performances are mainly governed by the girth of the Tanner graph the new decoding procedure has performances which significantly depend on the minimum distance and spectrum of the woven code. Short woven graph codes with large minimum distances are tabulated.**

I. Introduction

It is well-known that long low-density parity-check (LDPC) codes with BP decoding have good bit error rate (BER) performance close to the theoretical limit [1], [2]. On the other hand, LDPC codes have minimum distance which is less than that for the best known linear codes with the same parameters. This is not surprising since the minimum distance does not play an important role in iterative decoding. As a consequence, in numerous applications of LDPC codes with strict delay requirements such as, for example, wireless communications, BER and FER performances of short LDPC codes with BP decoding are rather far from maximum *a posteriori* probability (MAP) decoding and maximum-likelihood (ML) decoding performances, respectively.

A lot of attempts were done to improve the performance of LDPC codes by both improving code structures and improving decoding procedures. So-called *generalized* LDPC (GLDPC) codes were introduced in [3] and later studied in, for example, [4], [5], and [6]. The main idea behind this class of codes is to interpret an LDPC code as a concatenated code with single parity-check constituent codes and to improve the parameters of the code by replacing single parity-check codes by more powerful constituent codes. Simulated BER and FER performances of BP decoding with BCJR constituent decoders were presented in [5], [6]. Modifications of BP decoding with improved message exchange between symbol and check nodes were considered in [5]. Although the obtained results were promising, no GLDPC codes superior to their LDPC counterparts of short lengths were reported.

Relatively recently a significant step forward in obtaining better LDPC codes and their decoders was done by going to *nonbinary* LDPC codes. While keeping good asymptotic performance nonbinary LDPC codes of short and moderate lengths have improved both BER and FER performances but at the expense of increased decoding complexity ([7], [8], [9]). Codes of this class can be interpreted as GLDPC codes with nonbinary constituent codes. Nonbinary BCJR decoding, that is, computing bit likelihoods jointly, applied to constituent codes leads to better performance of generalized BP decoding compared to binary BP decoding. Actually, generalized BP decoding of nonbinary LDPC codes demonstrates so far the best FER performance for short and moderate lengths among iterative decoding procedures. However, these performances are still far from those of ML-decoding.

Woven graph codes were introduced in [10] and analyzed in [11]. In particular, in [11] it was shown that in the random ensemble of such codes there exist codes satisfying the Varshamov-Gilbert bound on the minimum distance. In other words, if ML decoding of woven graph codes would be feasible it would provide error probability performances close to the theoretical limits. Woven graph codes can be considered as a particular case of nonbinary LDPC codes where in each symbol node of the Tanner graph the same constituent code or its permutation determined by the underlying graph is used. Although none iterative decoding for woven codes was suggested in [10], [11] it is clear that generalized BP decoding can be applied to this class of codes.

Other approaches to reducing the gap between iterative and ML decoding were presented in [12], [13].

However, for code lengths of the order 50 – 200 none of the above mentioned decoding procedures can compete with ML decoding of tailbitten convolutional codes [14], [15]. Thus, the problem of constructing good short codes with near maximum-likelihood low-complexity decoding remains open.

In this paper, we shall present a new near ML low-complexity decoding procedure for woven graph codes of short and moderate lengths. We show that the FER performance of the proposed suboptimal decoding algorithm significantly depends on the code spectrum. Comparison with ML decoding of tailbitten convolutional codes of the same lengths as well as with generalized BP decoding of woven graph codes is performed in Section IV. Examples of rate $R = 1/3$ woven graph codes of lengths up to 126 with large minimum distances are tabulated. The paper is concluded with a summary in Section V.

II. GLDPC, Nonbinary LDPC, and Woven Graph Codes

A rate R_{G} GLDPC code $\mathcal{C}$ is determined by its parity-check matrix H_{G} obtained from the $(r_{\mathrm{g}} \times n_{\mathrm{g}})$ parity-check matrix H of the underlying LDPC code of rate R by replacing the nonzero elements of ith row by a parity-check matrix H_i^c of a rate $R^{\mathrm{c}} = k/n$ constituent code and the zero elements of each row by all-zero matrix of appropriate size. Notice that the matrix H can also be considered as an incidence matrix of a hypergraph [11]. One more generalization of an LDPC code determined by a parity-check matrix H can be obtained by replacing the nonzero elements of each row of H by the powers of a primitive element of $GF(q)$, where $q = 2^m$, $m > 1$ is an integer. The obtained parity-check matrix H_{NB} determines a nonbinary LDPC code of rate $R_{\mathrm{NB}} = R$.

As mentioned in Section I an s-partite hypergraph-based woven code of rate $R_{\mathrm{W}} = s(R^{\mathrm{c}}-1)+1$ with block constituent codes of rate R^{c} [11] can be interpreted as a particular case of a nonbinary LDPC code. It is determined by a parity-check matrix H_{W}

$$H_{\mathrm{W}} = \begin{pmatrix} H_1 \\ H_2 \\ \vdots \\ H_s \end{pmatrix} \tag{1}$$

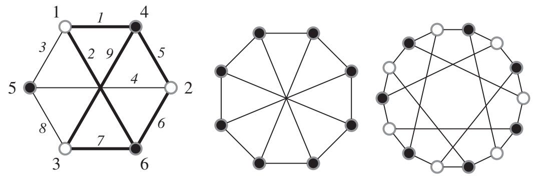
where the parity-check matrix H_1 is a block matrix

$$H_1 = \begin{pmatrix} H^{\mathrm{c}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & H^{\mathrm{c}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & H^{\mathrm{c}} \end{pmatrix} \tag{2}$$

of size $N \times Nl$, the parity-check matrix H^c of a rate $(l-1)/l$ constituent code has size $m \times ml$, H_i, $i = 2, 3, \ldots, s$ are permutations of the columns of H_1 determined by the graph (hypergraph), N denotes the number of constituent codes, and s denotes the number of ones in each column of H.

Notice that the representation (1) implies that H determines a regular s-partite graph [16]. In this paper, we mainly focus on such a representation but in general the underlying graphs, which are neither regular nor partite, can be used in woven constructions.

Next we give an example of a rate $R_{\mathrm{W}} = 1/3$ woven graph code based on the famous utility graph [16] shown in Fig. 1a.



a) Utility graph b) Wagner graph c) Heawood graph

Fig. 1. Famous Graphs

Example 1: The incidence matrix of the utility graph with $r_{\mathrm{g}} = 6$ vertices and $n_{\mathrm{g}} = 9$ edges is

$$H = \begin{array}{c|ccccccccc} & \mathit{1} & \mathit{2} & \mathit{3} & \mathit{4} & \mathit{5} & \mathit{6} & \mathit{7} & \mathit{8} & \mathit{9} \\ \hline \mathit{1} & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathit{2} & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ \mathit{3} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ \mathit{4} & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ \mathit{5} & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ \mathit{6} & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{array} \tag{3}$$

The corresponding matrix H_{W} has the form (1) – (2) with $s = 2$, $N = 3$ and

$$H_1 = \begin{pmatrix} H_1^{\mathrm{c}} & H_2^{\mathrm{c}} & H_3^{\mathrm{c}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & H_1^{\mathrm{c}} & H_2^{\mathrm{c}} & H_3^{\mathrm{c}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & H_1^{\mathrm{c}} & H_2^{\mathrm{c}} & H_3^{\mathrm{c}} \end{pmatrix} \tag{4}$$

$$H_2 = \begin{pmatrix} H_1^{\mathrm{c}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & H_2^{\mathrm{c}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & H_3^{\mathrm{c}} \\ \mathbf{0} & \mathbf{0} & H_1^{\mathrm{c}} & H_3^{\mathrm{c}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & H_2^{\mathrm{c}} & \mathbf{0} \\ \mathbf{0} & H_1^{\mathrm{c}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & H_3^{\mathrm{c}} & H_2^{\mathrm{c}} & \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{5}$$

The parity-check matrix of the constituent code consists of $l = 3$ submatrices of size $m \times m$, $m = 4$

$$H^{\mathrm{c}} = \begin{pmatrix} H_1^{\mathrm{c}} & H_2^{\mathrm{c}} & H_3^{\mathrm{c}} \end{pmatrix} \tag{6}$$

where

$$H_1^{\mathrm{c}} = \begin{pmatrix} 1\,0\,0\,0 \\ 0\,1\,0\,0 \\ 0\,0\,1\,0 \\ 0\,0\,0\,1 \end{pmatrix}, H_2^{\mathrm{c}} = \begin{pmatrix} 1\,1\,0\,1 \\ 1\,0\,1\,0 \\ 0\,1\,0\,1 \\ 1\,1\,1\,0 \end{pmatrix}, H_3^{\mathrm{c}} = \begin{pmatrix} 0\,1\,0\,1 \\ 1\,1\,1\,0 \\ 0\,1\,1\,1 \\ 1\,1\,1\,1 \end{pmatrix} \tag{7}$$

Notice that non-degenerate submatrices H_1^{c}, H_2^{c}, and H_3^{c} can be chosen in such a way that they would be powers of the companion matrix of a primitive polynomial of $GF(2^m)$. In this case the corresponding woven graph code is equivalent to a nonbinary LDPC code. In our example H_1^{c}, H_2^{c}, and H_3^{c} correspond to α^0, α, and α^4, where α is the primitive element of $GF(2^4)$ determined by the primitive polynomial $\alpha^4+\alpha+1$. The constituent codes corresponding to different rows of H_{W} are equivalent. Each codeword of one code can be obtained by block-wise permutation of a codeword of another code.

III. Decoding Algorithm

It is well-known that the main advantage of iterative decoding commonly used for decoding of GLDPC codes is relatively low computational complexity. We would like to keep this property in our new decoding procedure. Moreover, it follows from (4)–(5) that in woven constructions some subsets of q-ary code symbols form codewords of a constituent code (or their permutations). In fact, decoding of a woven graph code (GLDPC or nonbinary LDPC) is reduced to a few decodings of a constituent code.

It is easy to see that for any woven graph code there exists a subset of constituent codes such that if the corresponding constituent codewords are correctly decoded then the codewords of the remaining constituent codes can be reconstructed from the obtained decoding results. In other words, a codeword of the woven graph code can be easily reconstructed from the first set of constituent codewords. A trivial example: if the constituent codewords corresponding to H_1 in (4) are known then it is not necessary to reconstruct the constituent codewords corresponding to H_2.

The minimal set of such constituent codes or rows of $H_{\rm W}$ (each row uniquely determines a constituent code) we call *recovering set*. The recovering set is not unique. We consider the woven graph code from Example 1. For this code the minimal size of the recovering set is $n_{\rm R} = 2$. There exist in total $N_{\rm R} = 6$ different recovering sets of rows: {1,2}, {1,3}, {2,3}, {4,5}, {4,6} and {5,6}. It can be verified that for woven graph codes based on the Wagner graph (Fig. 1b) and on the Heawood graph (Fig. 1c) there are 32 recovering sets of size 3 and 70 recovering sets of size 4, respectively.

We call these sets "recovering sets" in order to distinguish them from information sets [17]. Notice that the code positions covered by each of the recovering sets contain an information set as a subset. In this sense recovering sets are redundant, and this redundancy plays an important role in the decoding algorithm.

Next we show how a codeword of the woven graph code from Example 1 can be reconstructed from a given recovering set. Assume that we chose the set {1,2}, that is, the constituent codewords corresponding to rows 1 and 2 are known. It means that the code symbols on positions 1,2,...,6 are known. In the 4th row, two among three q-ary symbols are known. It is enough to reconstruct the third q-ary symbol. It means that now the symbol on position 9 is also known. Similarly, from rows 5 and 6 we can reconstruct the symbols on positions 8 and 7, respectively.

Now we are prepared to explain our *Recovering sets based Soft-Decoding (RSB)* Algorithm.

At the first step of the algorithm ML list decoding with list size L is performed for each of $r_{\rm g}$ rows of the parity-check matrix H determining the underlying LDPC code or graph. The result of the first step is a set of $r_{\rm g}$ lists of size L each of which contains L estimates for the corresponding constituent codeword and their likelihood values.

Now we could make attempts of reconstructing a codeword using $N_{\rm R}$ different recovering sets of size $n_{\rm R}$. It is evident that the total number of candidates is equal to $N_{\rm R} L^{n_{\rm R}}$. Typically, this value is rather large and an exhaustive search over such a set of candidates has high computational complexity. We reduce the decoding complexity by

1 Exploiting the redundancy of the recovering sets.
2 Imposing restrictions on the sets of recovering sets.
3 Limiting the maximum number of decoding trials using some stopping rules.

Next we explain these steps using Example 1.

Using redundancy: Choose the recovering set {1,2}. It is clear that row 1 determines a constituent code with two information subblocks 1 and 2 and one check subblock 3. The same is true for row 2 (4 and 5 are the information subblocks and 6 is the check subblock). Indeed, rows 1 and 2 determine two constituent codes with three information subblocks (for example, 1, 2, 4) and three check subblocks (for example, 3, 5, 6) since the woven graph code has in total only three information subblocks. We can derive a parity-check matrix of size 1×4 determining a parity-check for subblocks 1, 2, 4, 5: $H_{1,2,4,5} = (\alpha^8 \ \alpha \ \alpha^5 \ 1)$. Parity-check matrices obtained for each of the recovering sets can be used as rejecting rules to exclude those combinations of constituent codewords which cannot form a codeword of the woven construction.

To further reduce the number of candidate codewords for each of the recovering sets we consider only those candidate constituent codewords from the lists of solutions sorted according to their likelihood values for which the sums of their indices in the list is less than or equal to a predetermined value. We also reduce the computational complexity by restricting the number of decoding trials. If the same solution is obtained more than a predetermined number of times we stop the decoding procedure and output the obtained solution. A formal description of the decoding procedure is given by Algorithm 1.

Input: Channel output $\boldsymbol{r}$
Output: Estimated codeword $\boldsymbol{v}$
STEP 1: `Loop over rows of the matrix` H
for $i = 1$ **to** $r_{\rm g}$ **do**
  Generate a list of L most probable codewords of the constituent code corresponding to the ith row of H
end
STEP 2: `Rejecting inconsistent candidate codewords`
for *All recovering sets* **do**
  1) Filter out combinations of constituent codewords which do not satisfy parity checks of the redundant recovering set.
  2) For survived candidate combinations $\boldsymbol{s}$ compute metrics $\mu(\boldsymbol{s})$ as the sum of likelihoods of their constituent codewords.
end
STEP 3: `Sorting`
Order lists of the survived combinations $\boldsymbol{s}$ from all recovering sets according to $\mu(\boldsymbol{s})$.
STEP 4: `Reconstructing codeword`
count=0; $\mu_0 = 0$; $\boldsymbol{v}_0 = \boldsymbol{0}$;
Starting from the most reliable recovering set,
for *All survived recovering sets* **do**
  1) Recover a codeword $\boldsymbol{v}$ and compute its likelihood function $\mu(\boldsymbol{v})$.
  2) **if** $\mu(\boldsymbol{v}) > \mu_0$ **then** Update solution: $\mu_0 = \mu(\boldsymbol{v})$, $\boldsymbol{v}_0 = \boldsymbol{v}$, count=1;
   else if $\mu(\boldsymbol{v}) = \mu_0$ *and* $\boldsymbol{v} = \boldsymbol{v}_0$ **then** count++
  3) **if** *count* $\geq T$ **then** , goto STEP 5
end
STEP 5: `Final solution`
Output decoding result: $\boldsymbol{v} = \boldsymbol{v}_0$.

Algorithm 1: RSB Algorithm

Notice that STEPS 2 and 4 of the decoding algorithm are most computationally consuming. In general, the computational complexity of STEP 2 is proportional to L^{n_R}. However, by applying a bidirectional search it can be reduced to $L^{n_\mathrm{R}/2}$. We accept as a complexity measure the number of iterations on STEP 4 as a function of signal-to-noise ratio per bit (SNR). It will be discussed in Section IV.

IV. Code Search and Simulation Results

In the following FER performance of the new decoding algorithm for a small selection of woven graph codes of rate $R_\mathrm{W} = 1/3$ with block length $n \approx 100$ shall be presented and compared with FER performance of generalized BP decoding and ML decoding for the same codes as well as with FER performance of ML decoding for the tailbiting block codes from the WINNER project [14]. The minimum distances $d_{\min}$ of the woven graph codes are given as well. Notice that instead of simulating the ML decoder for woven graph codes we used a lower bound on the FER performance of ML decoding. For each error event in the new suboptimal decoder we compared the likelihood of the obtained solution with the likelihood of the transmitted codeword, that is, we checked if the ML decoder would make this error as well. If the likelihood of the transmitted codeword was less than that of the solution we counted this error event as an ML decoding error.

In particular, the following block codes shall be compared: (the applied algorithms are given in parenthesis)

- $(75, 25)$ TB block code as defined in [14] (ML decoding with complexity 2^{18})
- $(84, 28)$ woven graph code with $d_{\min}$=15 obtained from the Heawood graph (ML decoding, near-ML decoding, BP decoding)
- $(126, 42)$ woven graph code with $d_{\min}$=19 obtained from the Heawood graph (ML decoding, near-ML decoding, BP decoding)

The corresponding FER performance for those codes is given in Fig. 2. Notice that 50 iterations of BP decoding were used. All algorithms were simulated until 50 error events have been occurred. It follows from the plots in Fig. 2 that woven graph codes with near-ML decoding significantly outperform the same codes with BP decoding. FER performance of woven graph codes with near-ML decoding is very close to the FER performance of their ML decoding and improves the FER performance of ML decoding for codes from the WINNER project. Moreover, increasing the minimum distance of woven graph codes plays a positive role in our decoding algorithm. However, the FER performance of BP decoding with increasing the minimum distance is improved only for rather high SNRs. For low SNRs the FER performance of the $(126, 42)$ woven graph code with $d_{\min}$=19 is the same as for the $(84, 28)$ woven graph code with $d_{\min}$=15.

A complexity estimate via the number of iterations on STEP 4 as a function of SNR is shown in Fig. 3. It easy to see that the computational complexity of the decoding procedure decreases with SNR rather fast and is quite reasonable at high SNRs. At the same plot we show BEAST [18] ML decoding complexity for the TB (75,25)-code from [15] measured as the average

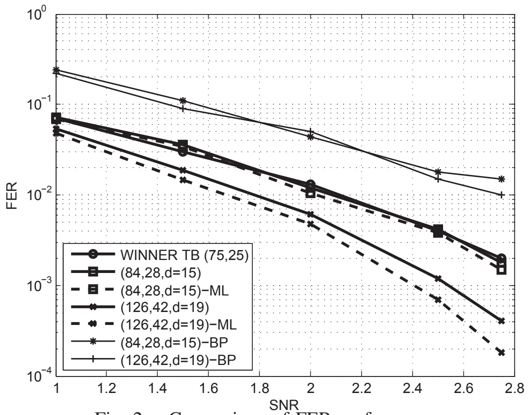


Fig. 2. Comparison of FER performances

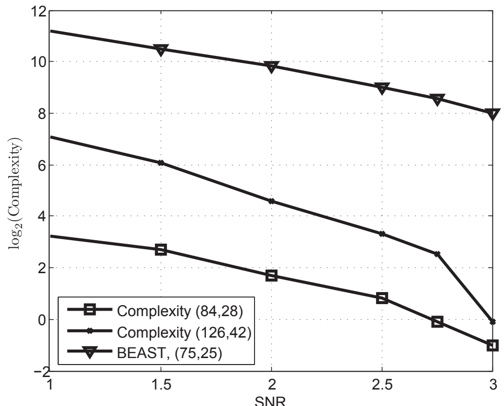


Fig. 3. Number of iterations on STEP 4 and BEAST ML decoding complexity as functions of SNR

number of visited tree nodes per decoded bit. The BEAST complexity is much lower than that for the corresponding code from [14], but it is still high and decreases with SNR slower than for the RSB decoder.

Since the minimum distance of the woven graph codes plays an important role in the new decoding, the following question arises: Is it possible to construct optimum or near-optimum short linear codes in the form of woven graph codes? In Table I the parameters of the new woven graph codes of short lengths are given. The minimum distances d_B for the best rate $R = 1/3$ linear block codes of the same lengths are given in the same table. It is easy to see that the minimum distances of the woven graph codes are a little bit less than those for the best linear codes and the gap between the minimum distances grows slowly with the code length. However, taking into account the structure of woven graph codes we can improve the upper bound on their minimum distance. First we explain the idea of our proof by an example and then formulate the corresponding theorem.

Consider the utility graph (Fig. 1a). Its incidence matrix can be interpreted as a parity-check matrix of the (9,3) graph code $\mathcal{C}$. We choose a (9,1) subcode $\mathcal{C}'$ of this code consisting of those codewords of $\mathcal{C}$ which have zeros on positions 3, 8, and 9. It is easy to verify that the parity-check matrix of $\mathcal{C}'$ determines the subgraph Γ'_U of the utility graph shown in bold in Fig. 1a. By labeling the edges of the utility graph by elements of $GF(2^4)$ or by equivalent binary submatrices of size (4×4) we obtain a (36,12) woven graph code with a (24,4) subcode corresponding to the labeled subgraph Γ'_{U}. The minimum distance of a (36,12) linear block code is upper-bounded by $d_{\mathrm{B}} = 12$ [19], the minimum distance of a (24,4) linear code is upper-bounded by $d_{\mathrm{U}} = 12$. It is clear that both d_{B} and d_{U} upper-bound the minimum distance of the woven graph code. In our example, $d_{\mathrm{B}} = d_{\mathrm{U}}$ but in general we can have $d_{\mathrm{U}} \leq d_{\mathrm{B}}$.

Theorem 1: Let Γ be a graph and $\Gamma' \in \Gamma$ be its subgraph such that all its vertices are of degree larger than 1. Denote by $d_{\mathrm{U}}(\Gamma, m)$ the upper bound on the minimum distance of a woven graph code based on Γ with $(m \times m)$ submatrices determining elements of the constituent code. Then

$$d_{\mathrm{U}}(\Gamma, m) \leq \min_{\Gamma' \in \Gamma} d_{\mathrm{U}}(\Gamma', m). \tag{8}$$

Proof: The codewords of a woven graph code correspond to cycles of the graph. Therefore, a linear code determined by the labeled subgraph $\Gamma' \in \Gamma$ is a subcode of the woven graph code determined by the labeled graph Γ. The statement of the theorem follows from the fact that the minimum distance of a linear code cannot be larger than that of its subcode. ■

The code description In Table I includes a primitive polynomial of $GF(2^m)$ in octal form, powers of the corresponding primitive elements and permutations determining H_2. Notice that the permutations for the woven graph codes based on the Wagner and Heawood graphs are omitted for shortness.

TABLE I
CODE PARAMETERS

m	$n, k, d_{\min}$	$d_{\mathrm{B}}(d_{\mathrm{U}})$	Code description	Spectrum
Utility graph				
3	27,9,8	10	0,1,4 (p=13) 231,213,312	4,16,34…
4	36,12,10	12	0,4,9 (p=23) 123,132,132	5,43,86…
5	45,15,12	14	0,6,17 (p=45) 231,123, 312	11, 99,177…
6	54,18,13	16	0,17,36 (p=103) 123,123,321	6,33,132…
6	54,18,14	16	0,7,39 (p=103) 231,123,312	30,144,330…
7	63,21,15	18	0,18,59 (p=211) 231,123,312	3,81,180…
8	72,24,16	20	0,38,46 (p=675) 231,123,312	9,45, 192…
9	81,27,17	22	0.30,112 (p=1021) 231,231,123	3,29,79…
10	90,30,19	24	0,171,449 (p=2011) 123,231,132	5, 33, 136…
Wagner graph				
7	84,28,16	22	0,18,59, (p=211)	1,8,40…
7	84,28,17	22	0,18,59, (p=211)	14,40,97…
Heawood graph				
3	63,21,12	18(15)	0,1,4 (p=13)	10,29,88…
4	84,28,15	22(18)	0,4,9 (p=23)	5,36,103…
5	105,35,17	26(22)	0,6,17 (p=45)	3,25,73…
6	126,42,19	31(26)	0,7,39 (p=103)	5,15,35…

V. Summary

A new near-ML decoding algorithm for generalized LDPC, nonbinary LDPC, and woven graph codes has been presented. Newly found woven graph codes of rate $R = 1/3$ with the new decoding have been compared with the corresponding codes defined in the WINNER project [14]. The woven graph codes yielded FER performances close to the FER performances of ML decoding of the same codes and they have won compared to the codes from the WINNER project. Moreover, it has been shown that the FER performance of the new decoding are improving when the minimum distance of the code grows. A table of new woven graph codes with improved minimum distances is given.

Acknowledgements

This research was supported in part by the Swedish Research Council under Grant 621-2010-5847.

References

[1] R. G. Gallager, *Low-density parity-check codes*. M.I.T. Press: Cambridge, MA, 1963.
[2] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, “Design of capacity-approaching irregular low-density parity-check codes,” *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.
[3] R. M. Tanner, “A recursive approach to low-complexity codes,” *IEEE Trans. Inf. Theory*, vol. 27, no. 5, pp. 533–546, 1981.
[4] N. Miladinovic and M. Fossorier, “Generalized LDPC codes and generalized stopping sets,” *IEEE Trans. Commun.*, vol. 56, no. 2, pp. 201–212, 2008.
[5] S. Abu-Surra, G. Liva, and W. E. Ryan, “On generalized LDPC codes and their decoders,” in *IEEE Commun. Theory Workshop.*, 2007.
[6] J. Boutros, O. Pothier, and G. Zemor, “Generalized low density (Tanner) codes,” in *Proc. ICC’99*, 1999, pp. 441–445.
[7] M. Davey and D. J. C. Mackay, “Low density parity check codes over GF(q),” *IEEE Commun Lett.*, vol. 2, no. 6, pp. 165–167, 1998.
[8] X.-Y. Hu and E. Eleftheriou, “Binary representation of cycle Tanner-graph GF(2^q) codes,” in *Proc. IEEE Int. Conf. Commun., Paris, France*, Jun. 2004, pp. 528–532.
[9] S. E. Hassani, M.-H. Hamon, and P. Penard, “A comparison study of binary and non-binary LDPC codes decoding,” in *Proc. SoftCOM*, Sep. 2010, pp. 355–359.
[10] I. E. Bocharova, R. Johannesson, B. D. Kudryashov, and V. V. Zyablov, “Asymptotic performances of woven graph codes,” in *Proc. ISIT*, Jul. 2008, pp. 1025–1029.
[11] ——, “Woven graph codes: Asymptotic performances and examples,” *IEEE Trans. Inf. Theory*, vol. 56, no. 1, pp. 121–167, 2010.
[12] H. Pishro-Nik and F. Fekri, “Results on punctured low-density parity-check codes and improved iterative decoding techniques,” *IEEE Trans. Inf. Theory*, vol. 53, no. 2, pp. 599–614, 2007.
[13] A. Valembois and M. Fossorier, “Box and match techniques applied to soft-decision decoding,” *IEEE Trans. Inf. Theory*, vol. 50, no. 5, pp. 796–810, 2004.
[14] WINNER Sixth Framework IST Project. [Online]. Available: http://www.ist-winner.org/
[15] I. Bocharova, R. Johannesson, B. Kudryashov, and P. Stahl, “Tailbiting codes: Bounds and search results,” *IEEE Trans. Inf. Theory*, vol. 48, no. 1, pp. 137–148, 2002.
[16] A. S. Asratian, T. M. J. Denley, and R. Haggkvist, *Bipartite graphs and their applications*. Cambridge University Press: Cambridge, U.K, 1998.
[17] E. Prange, “The use of information sets in decoding cyclic codes,” *IRE*, vol. 8, pp. S5–S8, 1962.
[18] I. Bocharova, M. Handlery, R. Johannesson, and B. Kudryashov, “A BEAST for prowling in trees,” *IEEE Trans. Inf. Theory*, vol. 50, no. 6, pp. 1295–1302, 2004.
[19] M. Grassl. Bounds on the minimum distance of linear codes and quantum codes. [Online]. Available: http://www.codetables.de