Construction of Structured Nonbinary Low-Density Parity-Check Codes

Wojciech Sułek, Marcin Kucharczyk, and Grzegorz Dziwoki

***Abstract*—Low Density Parity Check (LDPC) codes over nonbinary Galois Fields GF(*q*) are a generalization of the industrial standard binary LDPC codes for forward error correction in communication and information systems. The nonbinary codes can achieve significantly better performance for short and moderate block lengths. A lot of works concerning "good" LDPC codes parity check matrix construction has been published so far. However, it is well known that efficient partially parallel hardware decoder architectures are allowed only for codes with blockwise partitioned structure of the parity check matrix, called structured codes. In this paper we present a versatile algorithm for construction of codes that are both nonbinary and structured. The proposed algorithm aims at optimizing the code graph (Tanner graph) by reducing the existence of small cycles with low external connectivity, while at the same time selecting appropriate nonzero coefficients from the Galois Field under interest. The algorithm can be used for code construction of any field order, block length and code rate.**

***Index Terms*—LDPC codes, nonbinary codes, structured codes, tanner graph.**

I. Introduction

Low-density parity-check (LDPC) codes, after their "rediscovery" in late 90's [1], have attracted great research attention due to their excellent error-correcting performance and highly parallel iterative decoding scheme. They have become the industry standard for error correction coding, adopted for instance in the ETSI Digital Video Broadcasting (DVB) and the IEEE WiMAX.

In the case of small to moderate codeword length or in the case of higher order modulation, the nonbinary LDPC codes over Galois Fields GF (q) [2] can outperform their binary counterparts with comparable bit-length and rate. At the same time, a so-called structured LDPC codes offer the advantage of reduced implementation complexity and resolved memory access contention in the semi-parallel hardware decoder implementation [3]-[5]. Therefore it is of great interest to develop algorithmic design methods for codes construction that are both nonbinary and structured. The class of structured codes is also known as Architecture-Aware LDPC (AA-LDPC) or Implementation Oriented codes [3], [6].

A lot of works concerning parity check matrix construction for LDPC codes has been published so far, e.g. [7]-[9]. However, the literature concerning construction methods for structured LDPC codes is still quite poor, especially for the nonbinary codes. In this paper we present a flexible algorithm based on computer search for "good" structured nonbinary LDPC codes. We use the known fact of the relationship between performance of the code and existence of some harmful subgraphs [10], [11] in the code graph (Tanner graph [12]). Moreover we take into account the performance dependence on the coefficients selection for the nonzero parity check matrix entries [2], [13]. Our code construction algorithm combines the reduction of harmful subgraphs in the structured code graph with the specific coefficients selection for the nonbinary entries. The algorithm can be used for code construction of any block length and code rate.

The paper is organized as follows. The next section recalls the definition of nonbinary LDPC codes and their structured subclass. Definitions of the concepts connected with code graph properties are presented in Section III. Then in Section IV we present the developed algorithm. Finally in Sections V and VI we present results of the algorithm experimental verification and the conclusions.

II. Structured LDPC Codes over Galois Fields

Low-density parity-check codes are a class of a linear block error correcting codes. Encoding process for a linear code (N, K) adds M=N−K redundant elements to the information vector $\boldsymbol{u}=\{u_1, u_2, \ldots, u_K\}$ to form the code vector $\boldsymbol{x}=\{x_1, x_2, \ldots, x_N\}$. The information and code vectors are defined over the Galois field GF(q) with restriction to fields of the size being power of two ($q=2^p$). In the case of the well known binary codes the field size is 2 (thus p=1), whereas for the nonbinary codes p>1.

The (N,K) LDPC code is defined by a low density parity check matrix $\boldsymbol{H}_{M\times N}$ with GF(2^p) entries, where M=N−K is the number of the parity check equations. Remark that since the information vectors are over GF(2^p), the source block comprises $K\cdot p$ bits and the code block comprises $N\cdot p$ bits. We denote the entries of the parity check matrix as $h_{m,n}$.

In the decoder, a row vector $\boldsymbol{c}$ (in GF(2^p)) of length N is recognized as a correct codeword if and only if it satisfies the parity check equation $\boldsymbol{Hc}^T = \boldsymbol{0}_{M\times N}$, where the operations ("+" and "·") are performed in the Galois field arithmetic. This equation can be partitioned into M checks associated with M rows of $\boldsymbol{H}$. When the parity check equation is not satisfied, then the error correction decoding is applied by means of the iterative message passing algorithm [1].

As is well known (see e.g. [3], [6], [14]), efficient partially

Manuscript received March 28, 2013; revised July 12, 2013. This work was supported by the Polish National Science Centre under Grant number 4698/B/T02/2011/40.

W. Sulek, M. Kucharczyk, and G. Dziwoki are with Faculty of Automatic Control, Electronics and Computer Science, Silesian University of Technology, Akademicka 16, 44-100 Gliwice, Poland (e-mail: Wojciech.Sulek@polsl.pl, Marcin.Kucharczyk@polsl.pl, Grzegorz.Dziwoki@polsl.pl).

DOI: 10.7763/IJEEEE.2013.V3.267

parallel decoder implementation is possible for parity-check matrices with special constraints on their form. The main building blocks of partially-parallel decoder are message memories and a number of computation units. In order to suitably organize message memories accesses and eliminate contentions, the parity check matrix should be in a structured form, partitioned into square submatrices. In the literature this problem is treated mainly for binary LDPC codes, however it is similar for codes over higher order Galois fields.

The structured GF (q) LDPC code is defined by the parity check matrix $\boldsymbol{H}$ being a composite of a square submatrices:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{P}_{1,1} & \boldsymbol{P}_{1,2} & \cdots & \boldsymbol{P}_{1,L} \\ \boldsymbol{P}_{2,1} & \boldsymbol{P}_{2,2} & \cdots & \boldsymbol{P}_{2,L} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{P}_{D,1} & \boldsymbol{P}_{D,2} & \cdots & \boldsymbol{P}_{D,L} \end{bmatrix} \quad (1)$$

where each submatrix $\boldsymbol{P}_{d,l}$ of size $P \times P$ is either an all-zero matrix or a matrix with exactly one nonzero element in every row and every column. In our construction of the GF(q) LDPC structured codes we use submatrices, which are cyclical shifts of columns of an identity matrix multiplied by a constant coefficient $a \in \mathrm{GF}(q)$, e.g.:

$$\boldsymbol{P}_{d,l} = \begin{bmatrix} 0 & a & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & a \\ a & 0 & 0 & 0 \end{bmatrix} \quad (2)$$

Therefore the "good" code construction algorithm encompass specifying positions of nonzero submatrices as well as cyclical shift values and coefficients a. The proposed algorithm use code graph optimization to determine positions and cyclical shifts as well as "good" row coefficient sets [13] for coefficients selection.

III. Code Graph Properties

LDPC codes can also be defined by an alternative to parity check matrix representation: a bipartite Tanner graph with so called symbol nodes representing data symbols and check nodes representing parity checks associated with rows of $\boldsymbol{H}$. In his section we review the essential code graph properties affecting the code error correcting capabilities.

The parity check matrix and its Tanner graph have direct correspondence, e.g. as shown in Fig. 1. We define Tanner graph $\mathcal{G} = (\mathcal{V}_c \cup \mathcal{V}_b, \mathcal{E})$, where $\mathcal{V}_c = \{c_1, c_2, \ldots, c_M\}$ is the set of the check nodes, $\mathcal{V}_b = \{b_1, b_2, \ldots, b_N\}$ is the set of symbol nodes and $\mathcal{E} \subseteq \mathcal{V}_b \times \mathcal{V}_c$ is the set of edges. An edge $e_i = (b_n, c_m)$ belongs to $\mathcal{E}$ if and only if $h_{m,n} \neq 0$. Because of the mentioned direct correspondence between $\boldsymbol{H}$ and $\mathcal{G}$, we will use them interchangeably for a code definition.

Using a concentration theorem, the authors of [15] have shown that for an ensemble of sufficiently long LDPC codes with given degree distribution of the graph nodes, the performance is concentrated at the average performance of the ensemble. However, at short to medium block lengths, performance of the randomly selected codes significantly deviate from the theoretical ensemble average performance. It is caused by short cycles in the Tanner graph as well as some more complex subgraphs that are shortly discussed below.

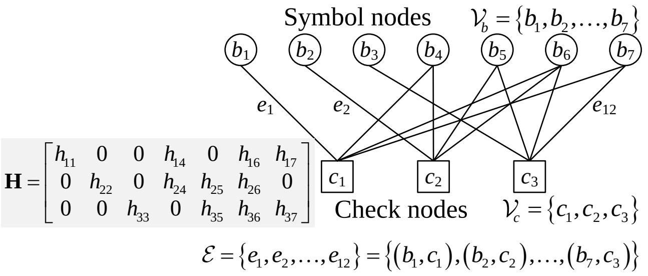

Symbol nodes
$\mathcal{V}_b = \{b_1, b_2, \ldots, b_7\}$
b_1
b_2
b_3
b_4
b_5
b_6
b_7
e_1
e_2
e_{12}
$\mathbf{H} = \begin{bmatrix} h_{11} & 0 & 0 & h_{14} & 0 & h_{16} & h_{17} \\ 0 & h_{22} & 0 & h_{24} & h_{25} & h_{26} & 0 \\ 0 & 0 & h_{33} & 0 & h_{35} & h_{36} & h_{37} \end{bmatrix}$
c_1
c_2
c_3
Check nodes
$\mathcal{V}_c = \{c_1, c_2, c_3\}$
$\mathcal{E} = \{e_1, e_2, \ldots, e_{12}\} = \{(b_1, c_1), (b_2, c_2), \ldots, (b_7, c_3)\}$

Fig. 1. Parity check matrix and Tanner graph.

A **cycle** in the graph is usually defined as a sequence of adjacent edges which starts and ends at the same vertex, and satisfies the condition that no vertex appears more than once. In this paper we also use a term of **closed walk**, which is a sequence of adjacent edges without restriction of their single appearance [16]. Cycles and closed walks are denoted as $e_1 \sim e_2 \sim \cdots \sim e_k \sim$, where $e_1, e_2, \ldots, e_k$ are consecutive edges in the cycle.

A **Stopping Set** $\mathcal{SS}_a$ is a subset of a symbol nodes, for which the subgraph of $\mathcal{G}$ induced by $\mathcal{SS}_a$ and its neighboring check nodes contains no check nodes of degree one [11].

A **Trapping Set** $\mathcal{TS}_{a,b}$ is a subset of a symbol nodes, for which the subgraph of $\mathcal{G}$ induced by $\mathcal{TS}_{a,b}$ and its neighbors contains b odd-degree check nodes [11], for example see Fig. 2.

As was proved [10], [11], the performance of an iteratively decoded LDPC code is limited by the existence of small stopping sets and trapping sets in the graph of the code. Since there are very few check nodes capable of correcting errors within the trapping set, this incorrect information remains "trapped" until the termination of the decoding process.

Finally, we recall the parameter that indicates "harmfulness" of the cycles in the Tanner graph. An extrinsic check node of a symbol node set is a check node that is singly connected to this set. The **Extrinsic Message Degree** (**EMD**) of a symbol node set (e.g. a set forming a cycle) is the number of extrinsic check nodes of this symbol node set [7]. A set of symbol nodes with large EMD will require concatenation of an additional nodes to become a stopping set.

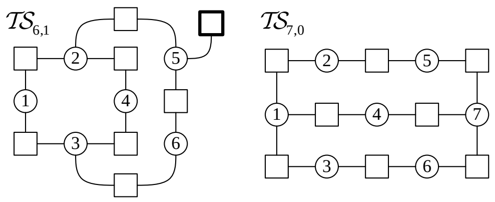


Fig. 2. Subgraphs induced by Trapping Sets $\mathcal{TS}_{6,1}$ and $\mathcal{TS}_{a,b}$.

In a bipartite graph free of degree-1 bit nodes, every stopping set contains cycles [7]. Similarly trapping sets are composed of interconnected cycles (Fig. 2). Hence, if short cycles are eliminated from the graph, also small stopping sets and trapping sets are eliminated. However, not only the length, but also the connectivity of cycles plays importantroles, because cycles having low EMD are more prone to induce small trapping sets [7]. Minimization of short cycles with low EMD is the basis of many published code construction methods, e.g. [7], [8], [17]. In this paper the idea of low-EMD short cycles elimination is adopted for structured GF(q) LDPC codes construction.

IV. Structured Codes Construction Algorithm

The parity check matrix $\boldsymbol{H}$ of size $DP\times LP$ that is suitable for efficient decoder architecture, as discussed earlier, is an $D\times L$ matrix of $P\times P$ submatrices. Let us define the $D\times L$ matrix $\mathbf{W}$ with entries $w_{d,l}$ =0 indicating the all-zero submatrices $\boldsymbol{P}_{d,l}$ in $\boldsymbol{H}$ and nonzero entries $w_{d,l} = a\neq 0$ indicating the nonzero submatrices positions in $\boldsymbol{H}$ and its coefficients $a \in \mathrm{GF}(2^p)$. We call $\boldsymbol{W}$ the seed matrix and the respective graph we call the seed graph. Thus the two main steps in the structured LDPC code parity check matrix construction are:

- the seed matrix $\boldsymbol{W}$ (seed graph) construction,
- expansion of the seed matrix (seed graph), which means replacing its scalar entries with appropriate $P\times P$ submatrices.

A. Seed Graph Construction

The seed matrix (seed graph) can be constructed in any known method for constructing binary LDPC parity check matrices without structured form. Especially suitable is the PEG (Progressive Edge Growth) algorithm [8], [9]. Then the nonzero entries of obtained binary matrix are substituted with GF(q) entries (coefficients). The selection of coefficients should be made carefully, because it affects the code performance [13]. This issue is treated later in this section.

Regardless of the method used for construction, the relatively small seed graph contains a lot of short cycles, which can be eliminated in the graph expansion. Thus, the crucial step for achieving desirable graph properties is the expansion of the seed graph.

B. Seed Graph Expansion

The seed graph expansion is defined by cyclical shift values selected for all nonzero submatrices (2). With every edge e_i in the seed graph, a cyclical shift value s_{ei} should be associated, where s_{ei} is an integer, $0 \leq s_{ei} < P$. Then the submatrix of the structured parity check matrix, associated with e_i, is a s_{ei} -shift of columns of the identity $P\times P$ matrix, multiplied by a coefficient taken from the seed matrix.

The proper selection of cyclical shift values is crucial for achieving a code graph with good properties. Cycles in the graph after expansion are related to closed walks in the seed graph, however they can be "removed" (strictly speaking: their length can be increased) in the expansion process by proper selection of the cyclical shift values. An example is presented in Fig. 3.

The crucial theorem characterizing the relationship between cycles in the seed graph and cycles in the expanded graph was presented in [18]. It can be summarized as follows. If the seed graph contains a closed walk $e_1 \sim e_2 \sim \cdots \sim e_k \sim$ of length k, then the graph after expansion contains respective cycles of length k if the following condition (3) is satisfied:

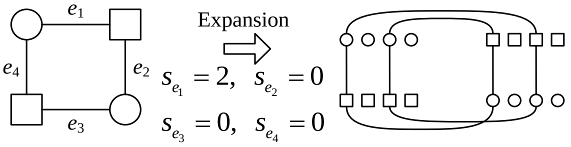


Fig. 3. An expansion of a subgraph induced by a 4-cycle.

$$s_{e1} - s_{e2} + s_{e3} - s_{e4} + \ldots + s_{ek-1} - s_{ek} \equiv 0 \mod P \tag{3}$$

where s_{e1}, s_{e2}, …, s_{ek} are the selected cyclical shift values associated with edges e_1, e_2, …, e_k.

The importance of taking into consideration all closed walks (not only cycles) in the seed graph is illustrated in Fig. 3, where the closed walk $e_1 \sim e_2 \sim e_3 \sim e_4 \sim$ is not a source of a 4-cycle (condition (3) not satisfied), but the closed walk $e_1 \sim e_2 \sim e_3 \sim e_4 \sim e_1 \sim e_2 \sim e_3 \sim e_4 \sim$ is a source of an 8-cycle. The condition (3) in this case is: $s_{e1} - s_{e2} + s_{e3} - s_{e4} + s_{e1} - s_{e2} + s_{e3} - s_{e4} = 2 - 0 + 0 - 0 + 2 - 0 + 0 - 0 \equiv 0 \pmod 4$.

The condition for non-existence of a cycle in the covering graph on the contrary to (3) is:

$$s_{e1} - s_{e2} + s_{e3} - s_{e4} + \ldots + s_{ek-1} - s_{ek} \not\equiv 0 \mod P \tag{4}$$

C. Nonzero Coefficients Selection

The nonzero coefficients of $\boldsymbol{W}$ choice affect performance of the code, as was proved in [13]. For every row of the parity check matrix, an independent choice of a set of nonzero coefficients can be made. The coefficient set for every row can be chosen from a precomputed collection of sets, with unrestricted sequence inside the set. In our structured code construction algorithm, we incorporate the coefficient collections pre-computing method presented in [13].

Moreover we incorporate the cycles conditioning method [13]: for every cycle that has not been removed in the expansion process, the proposed code construction algorithm tries to modify the coefficient set choices in a way that enables fulfilling the so-called FRC (full rank condition), which can be given as:

$$\prod_{i=1}^{k/2} \alpha_{2i-1} \neq \prod_{i=1}^{k/2} \alpha_{2i} \tag{5}$$

where a_1, a_2, … , a_k are the coefficients (elements of $\boldsymbol{W}$) associated with edges of a cycle in the structured code graph $e_1 \sim e_2 \sim \cdots \sim e_k \sim$, which has not been removed in the expansion process. The details about the theorem behind the FRC can be found in [13].

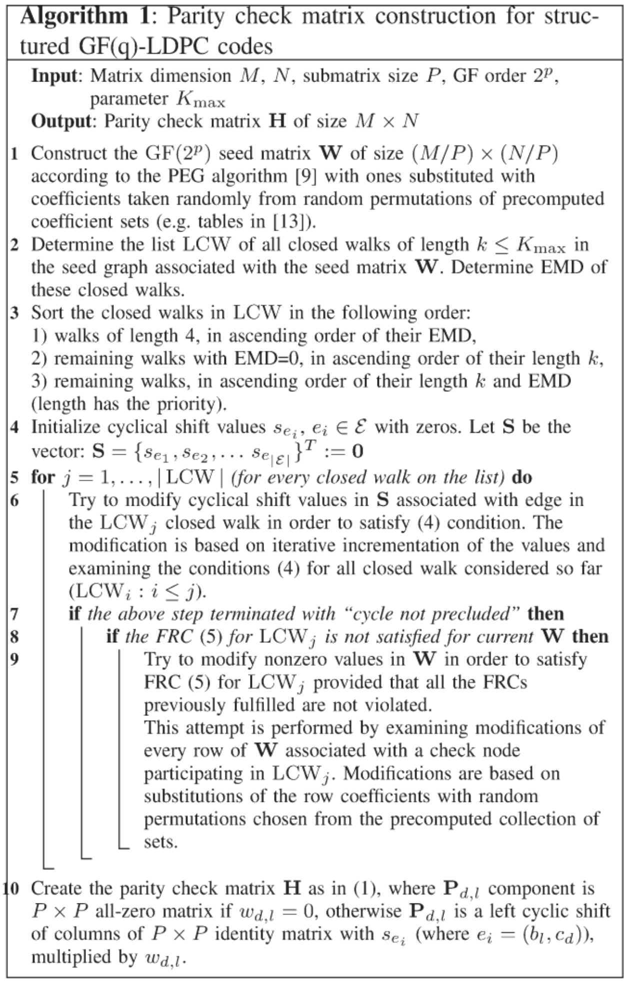
D. Code Construction Algorithm

Based on considerations presented in the previous sections, a proper direction to generate a graph for a structured GF(q) LDPC code is to construct a "good" seed graph and then select cyclical shift values s_{ei} in a way to remove the existing cycles (especially cycles with low EMD) in the expansion process. The complex algorithm for the parity check matrix construction that has been developed is summarized below as Algorithm 1.

The essential part of the algorithm is the seed graph expansion and careful selection of coefficients in order to satisfy FRC (5) for cycles that cannot be removed.

All closed walks in the seed graph of length lower than some predetermined $K_{\max}$ are placed on the priority list LCW (List of Closed Walks). Walks of length 4 begin the list, because of their well known destructive influence on the iterative decoding algorithm, then walks with EMD=0, because their bit nodes constitute the most harmful trapping sets $\mathcal{TS}_{a,0}$. Finally all remaining walks in order of of their length *k* and EMD are placed on the list.

Consecutively for every closed walk on the created LCW list, the cyclical shift values ***S*** are adjusted by a computer search technique that has been developed previously for binary codes (see [19]) by the authors of this paper. The technique is simply based on multiple trials of incrementing the values s_{ei} associated with edges e_i belonging to the closed walk.



Algorithm 1: Parity check matrix construction for structured GF(q)-LDPC codes

```
Input: Matrix dimension $M$, $N$, submatrix size $P$, GF order $2^p$,
       parameter $K_{\max}$
Output: Parity check matrix $\mathbf{H}$ of size $M \times N$
Construct the GF($2^p$) seed matrix $\mathbf{W}$ of size $(M/P) \times (N/P)$
according to the PEG algorithm [9] with ones substituted with
coefficients taken randomly from random permutations of precomputed
coefficient sets (e.g. tables in [13]).
Determine the list LCW of all closed walks of length $k \le K_{\max}$ in
the seed graph associated with the seed matrix $\mathbf{W}$. Determine EMD of
these closed walks.
Sort the closed walks in LCW in the following order:
1) walks of length 4, in ascending order of their EMD,
2) remaining walks with EMD=0, in ascending order of their length $k$,
3) remaining walks, in ascending order of their length $k$ and EMD
(length has the priority).
Initialize cyclical shift values $s_{e_i}$, $e_i \in \mathcal{E}$ with zeros. Let $\mathbf{S}$ be the
vector: $\mathbf{S} = \{s_{e_1}, s_{e_2}, \dots s_{e_{|\mathcal{E}|}}\}^T := \mathbf{0}$
for $j = 1, \dots, |\mathrm{LCW}|$ (for every closed walk on the list) do
    Try to modify cyclical shift values in $\mathbf{S}$ associated with edge in
    the LCW$_j$ closed walk in order to satisfy (4) condition. The
    modification is based on iterative incrementation of the values and
    examining the conditions (4) for all closed walk considered so far
    (LCW$_i : i \le j$).
    if the above step terminated with "cycle not precluded" then
        if the FRC (5) for LCW$_j$ is not satisfied for current $\mathbf{W}$ then
            Try to modify nonzero values in $\mathbf{W}$ in order to satisfy
            FRC (5) for LCW$_j$ provided that all the FRCs
            previously fulfilled are not violated.
            This attempt is performed by examining modifications of
            every row of $\mathbf{W}$ associated with a check node
            participating in LCW$_j$. Modifications are based on
            substitutions of the row coefficients with random
            permutations chosen from the precomputed collection of
            sets.
Create the parity check matrix $\mathbf{H}$ as in (1), where $\mathbf{P}_{d,l}$ component is
$P \times P$ all-zero matrix if $w_{d,l} = 0$, otherwise $\mathbf{P}_{d,l}$ is a left cyclic shift
of columns of $P \times P$ identity matrix with $s_{e_i}$ (where $e_i = (b_l, c_d)$),
multiplied by $w_{d,l}$.
```

After some number of stages, as the values of ***S*** are increasing, conditions for some cycles cannot be satisfied without violating conditions for previously considered ones. In such a case, the result of the search is "cycle not precluded". Then the coefficients of ***W*** associated with this cycle are adjusted in order to satisfy the FRC condition.

In summary, the novelty of the algorithm is the application of the specific combination of the cyclic shift values search and coefficient selection for structured GF(*q*) LDPC parity check matrix construction.

V. Results

A number of simulation experiments have been made in order to examine performance of the codes that can be obtained with the proposed code construction algorithm. We have constructed structured LDPC codes over Galois Fields of various orders, with various block lengths *N* and submatrix sizes *P*. Some of the results are presented in this section.

As a performance reference to the proposed algorithm we use two other approaches:

- Structured codes with the same seed matrix, but cyclical shifts as well as non-zero coefficients selected randomly. We call those codes "Random" in figures.
- Non-structured codes (submatrix size *P*=1) constructed with binary PEG algorithm, in which ones are substituted by random coefficients from the given Galois field. We call those codes "Rand. Coeffs" in figures.

Comparison with the former method shows significance of the proper cyclic shifts selection and comparison with the latter method shows significance of the coefficients choice.

Simulation results for rate-1/2 GF(2^4) and GF(2^6) LDPC codes are presented in Fig. 4- Fig. 5 respectively. We provide Frame Error Rate (FER) as well as Bit Error Rate (BER) curves in the function of Signal to Noise Ratio (E_b/N_0). The block size is 2400 bits in both cases, which gives *N*=600 symbols in the GF(2^4) case and *N*=400 symbols in the GF(2^6) case. We used the Binary Input AWGN (Additive White Gaussian Noise) channel model and the belief propagation decoding algorithm [2] for these simulations.

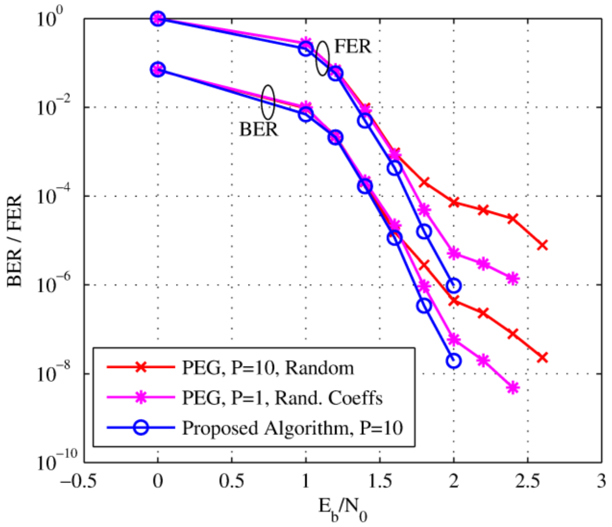


Fig. 4. GF(2^4) codes performance over AWGN channel.

Performance of the codes constructed with the proposed algorithm is significantly better than the reference codes of the same parity check matrix size. This observation confirms that both cyclic shifts selection and coefficients selection is important in "good" parity check matrix construction of structured GF(q) LDPC codes. Similar conclusion can be drawn based on results for other block sizes, submatrix sizes as well as other communication system and channel models.

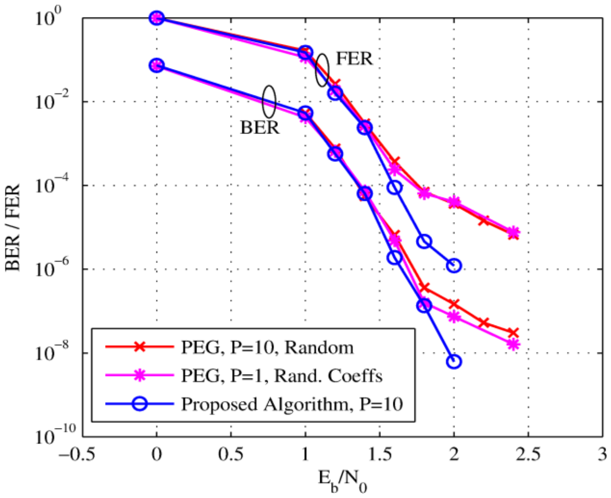


Fig. 5. GF(2^6) codes performance over AWGN channel.

VI. Conclusion

The complex algorithm for the structured nonbinary LDPC codes construction has been presented in the paper. Two main stages of the algorithm are the seed matrix construction and matrix expansion. The novelty of the algorithm is the application of the specific combination of the cyclic shift values search and coefficient selection in the stage of the seed matrix expansion.

The algorithm aims to reduce the number of low-length cycles with low external connectivity (EMD) in the Tanner graph by a proper cyclic shift value selection for the parity check matrix submatrices. This heuristically reduces the number of small Stopping Sets and Trapping Sets that are harmful to the code performance. At the same time the proper non-zero coefficient selection method is applied with the so-called FRC condition for the cycles that cannot be removed.

A number of structured GF(q) LDPC codes has been constructed making use of the proposed algorithm. Performance of the codes constructed with the algorithm is generally better than performance of the codes obtained with two reference algorithms for structured and non-structured codes. This confirms the effectiveness of the proposed algorithm. The nonbinary codes outperform their binary counterparts and the structured form of the parity check matrix enables efficient hardware decoder implementation.

References

[1] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, pp. 399-431, March 1999.

[2] M. C. Davey and D. MacKay, "Low-density parity check codes over gf(q)," *IEEE Commun. Lett.*, vol. 2, pp. 165-167, June 1998.

[3] H. Zhong and T. Zhang, "Block-LDPC: A practical LDPC coding system design approach," *IEEE Trans. Circuits Syst. I*, vol. 52, pp. 766-775, April 2005.

[4] Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam, and M. J. Winwright, "Design of LDPC decoders for improved low error rate performance: Quantization and algorithm choices," *IEEE Trans. Wireless Commun.*, vol. 8, pp. 3258-3268, November 2009.

[5] W. Sulek, M. Kucharczyk, and G. Dziwoki, "GF(q) LDPC decoder design for FPGA implementation," in *Proc. (IEEE) 10th Annual Consumer Communications & Networking Conference (CCNC)*, Las Vegas, USA, January 2013, pp. 445–450.

[6] J. Sha, Z. Wang, M. Gao, and L. Li, "Multi-Gb/s LDPC Code Design and Implementation," *IEEE Trans. VLSI Syst.*, vol. 17, pp. 262-268, February 2009.

[7] T. Tian, C. Jones, J. D. Villasenor, and R. D. Wesel, "Selective avoidance of cycles in irregular LDPC code construction," *IEEE Trans. Commun.*, vol. 52, pp. 1242-1247, August 2004.

[8] H. Xiao and A. H. Banihashemi, "Improved progressive-edge-growth (PEG) construction of irregular LDPC codes," *IEEE Commun. Lett.*, vol. 8, pp. 715-717, December 2004.

[9] X. Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Trans. Inf. Theory*, vol. 51, pp. 386-398, January 2005.

[10] C. Di, D. Proietti, I. E. Telatar, T. J. Richardson, and R. L. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *IEEE Trans. Inf. Theory*, vol. 48, pp. 1570-1579, June 2002.

[11] M. Karimi and A. H. Banihashemi, "Efficient algorithm for finding dominant trapping sets of LDPC codes," *IEEE Trans. Inf. Theory*, vol. 58, pp. 6942-6958, November 2012.

[12] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vol. 27, pp. 533-547, September 1981.

[13] C. Poulliat, M. Fossorier, and D. Declercq, "Design of regular (2, dc)-LDPC codes over GF(q) using their binary images," *IEEE Trans. Commun.*, vol. 56, pp. 1626-1635, October 2008.

[14] K. Zhang, X. Huang, and Z. Wang, "A high-throughput LDPC decoder architecture with rate compatibility," *IEEE Trans. Circuits Syst. I*, vol. 58, pp. 839-847, April 2011.

[15] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, pp. 599-618, February 2001.

[16] R. Diestel, *Graph Theory*, Berlin: Springer-Verlag, 2006.

[17] S.-H. Kim, D.-S. Kim, and H.-Y. Song, "Modification on the IPEG algorithm for constructing LDPC codes with low error floor," in *Proc. (IEEE) Vehicular Technology Conference*, Dublin, Ireland, April 2007, pp. 2215-2217.

[18] M. E. O'Sullivan, "Algebraic construction of sparse matrices with large girth," *IEEE Trans. Inf. Theory*, vol. 52, pp. 718-727, February 2006.

[19] W. Sulek, "Seed graph expansion for construction of structured LDPC codes," in *Proc. (IEEE) International Symposium on Wireless Communication Systems (ISWCS)*, Siena, Italy, September 2009, pp. 216-220.

Wojciech Sułek received the PhD in the discipline of Electronics from Silesian University of Technology, Gliwice, Poland, in 2009. He is an assistant professor in the Institute of Electronics at this University. His PhD thesis regarded Architecture Aware LDPC codes construction and decoder implementation. Modern coding theory and coding systems hardware design are under his research interests up till today.

Marcin Kucharczyk received MSc and PhD degrees in Electronics from Silesian University of Technology, Gliwice, Poland, in 2001 and 2006 respectively. Since 2006, he is an assistant professor in the Institute of Electronics at this University. His scientific interests are related to digital communication systems and signal processing, and are focused mainly on OFDM systems and forward error correction with LDPC codes.

Grzegorz Dziwoki received MSc and PhD degrees from Silesian University of Technology, Gliwice, Poland in 1998 and 2004 respectively. Since 2004, he has been an assistant professor in the Institute of Electronics at this University. His scientific interests concern digital signal processing in physical layer of communication systems.