Multiple-Bases Belief-Propagation with Leaking for Decoding of Moderate-Length Block Codes

Thorsten Hehn, Johannes B. Huber, Pei He, Stefan Laendner
Institute for Information Transmission, Cauerstr. 7, 91058 Erlangen, Germany, {hehn, huber, hepei, laendner}@LNT.de

Abstract

Short algebraic codes promise low-delay data transmission and good performance results when transmitted over the additive white Gaussian noise (AWGN) channel and decoded by maximum-likelihood (ML) soft-decision decoding. One reason for this is the large minimum distance of these codes. For belief-propagation (BP) decoding, short algebraic codes show suboptimal results due to their high-density parity-check matrices. Using a collection of parity-check matrices for a given code, Multiple-Bases Belief-Propagation (MBBP) allows for decoding of many dense linear block codes with near ML performance. One convenient way to generate these matrices is using the automorphism group of a code. We point out limits of this approach and show two novel improvements, a decoder modification and a construction algorithm for parity-check matrices, which emphasize that MBBP is a more general approach and independent of the automorphism group. We use these methods to extend the field of application for MBBP to codes of moderate length. This includes codes with parity-check matrices tailored for BP decoding, in particular Progressive Edge-Growth (PEG) codes.

1 Introduction

Iterative decoding of low-density parity-check (LDPC) codes by the BP algorithm was shown to approach the Shannon limit for asymptotically long codes and high iteration numbers [1]. For almost all applications, however, restrictions on computational complexity and tolerable time delay limit the practical performance of BP decoders. This has evoked interest in low-delay BP decoding, for which especially block codes of short to moderate length have been considered as they promise both low delay and good decoding performance. Since performance results for the standard BP decoder are poor, several authors have proposed decoder modifications tailored to obtain improved performance for short-length codes. In [2] the parity-check matrix is adapted after each iteration. There, columns corresponding to symbols with low reliability become very sparse which lowers the probability that less reliable information participates in a loop. The authors of [3] combine the approach from [2] with their concept of ordered statistics decoding [4]. Although improving decoding performance, all these methods require one or more steps involving a matrix reduction. In [5], multiple parity-check matrices are used in a sequential and random-like manner and a damping factor is applied. This scheme leads to good results for short codes but actually extends the decoding delay as it operates in serial fashion.

The recently introduced Multiple-Bases Belief-Propagation (MBBP) algorithm [6] performs parallel decoding operations using multiple BP decoders in order to obtain better decoding performance. At the same time it allows for a trade-off between decoding delay and parallel decoders. One practical way is to use parity-check matrices which are equivalent to each other under the permutations of the *automorphism group* of the code (cf. Section 3). In this case, MBBP approaches ML decoding performance for many dense codes of short and moderate length where we say that a code is dense if the weight of its parity-check matrix grows linearly with the square of the codelength. There, the weight of a matrix is the lowest number of ones within a selection of $n - k$ linearly independent rows. MBBP can obtain ML performance with less than 20 iterations per decoder for many dense codes [6]. However, the sole use of the automorphism group can confine the performance and field of application of MBBP decoding. The size of the automorphism group of a code is limited and, depending on the structure of the parity-check matrices, only a subset of the valid permutations can be used to identify non-equivalent matrices. In this work, we focus on alternative ways to construct parity-check matrix representations and, using these results, apply MBBP decoding to moderate-length LDPC codes with unknown automorphism group such as PEG codes. These are LDPC codes [7] defined by a parity-check matrix which is constructed by an algorithm that uses random numbers. For PEG codes, the automorphism group is in general unknown. However, MBBP decoding does not depend on the knowledge of this group.

The contributions presented in this work both aim at providing a set of decoder representations for a given code. First, a novel BP decoding method, termed *Leak BP decoding* ("Leaking"), is introduced. The Leak BP decoder considers at first only decoder input messages of high reliability for a given number of iterations and then successively inserts more and more unreliable information into the decoding process. A Leak BP decoder qualifies as an additional representation for MBBP when running on a parity-check matrix that is also processed by a decoder using standard BP. Second, a new method for constructing several parity-check matrices of a given code is introduced. This approach does not depend on the presence of an automorphism group and uses an existing parity-check matrix together with knowledge of the local girth to find non-equivalent parity-check matrices of comparable density. Note that both methods can be used for any linear code.

This paper is organized as follows. Section 2 introduces useful definitions and a consistent terminology. Section 3 details the standard MBBP decoder and certain limitations. Leaking is defined in Section 4, while Section 5 gives a detailed description of the construction algorithm for matrix collections. Simulation results are presented in Section 6.

2 Definitions and Terminology

Let us first give some definitions and introduce a consistent terminology. The considered system model includes transmission of binary data over the AWGN channel and estimation by a BP decoder. Source bits $u_\nu \in \{0,1\}$ are encoded by a linear binary block code and mapped using binary phase shift keying (BPSK) modulation. The equally likely channel symbols $x_\nu \in \{+1,-1\}$ are transmitted over the AWGN channel. At the channel output, the symbols y_ν are modeled as $y_\nu = x_\nu + n_\nu$, where the noise samples $n_\nu \sim \mathcal{N}(0,\sigma^2)$ follow a Gaussian distribution with zero mean and noise standard deviation σ.

Let $\mathcal{C}$ denote a linear $[n,k,d]$ block code. One way to generate the representations for the MBBP decoder is to apply the permutations of the automorphism group of the code. The definition in the sense of [8] is given in the following.

Definition 2.1: [8, Ch. 8] The permutations which send $\mathcal{C}$ into itself, i.e. codewords go into (possibly different) codewords, form the *automorphism group* of $\mathcal{C}$, denoted by $\mathrm{Aut}(\mathcal{C})$. If $\mathcal{C}^\perp$ denotes the code dual to $\mathcal{C}$, then $\mathrm{Aut}(\mathcal{C}) = \mathrm{Aut}(\mathcal{C}^\perp)$.

A special class of permutations is the set of cyclic shifts, as parity-check matrices of cyclic form compare favorably for dense algebraic codes [6].

Definition 2.2: A parity-check matrix of a code $\mathcal{C}$ is said to be in *cyclic form* if it consists of m cyclic shifts of one codeword of $\mathcal{C}^\perp$. The code $\mathcal{C}$ is called *cyclic* if any cyclic shift of a codeword $\boldsymbol{c} \in \mathcal{C}$ is also a codeword.

It can easily be deduced from Definition 2.1 that the dual code $\mathcal{C}^\perp$ of any cyclic code $\mathcal{C}$ is also cyclic. This guarantees that there exist parity-check matrices of cyclic form for any cyclic code.

For a thorough description of parity-check matrices of cyclic form, we find the following definition of cyclic orbit generators useful.

Definition 2.3: Let $\mathcal{C}$ be a binary linear cyclic code. Divide the set of codewords of $\mathcal{C}^\perp$ into sets (orbits) consisting only of cyclic shifts of one codeword. Choose one codeword to be the representative of each set, henceforth referred to as the *cyclic orbit generator* (cog).

Little technical modifications are required for non-cyclic codes with parity-check matrices of cyclic form.

3 MBBP Decoding

MBBP decoding [6] is an approach used for decoding dense algebraic codes, such as BCH codes or Golay codes, by means of the BP algorithm. MBBP decoding uses the fact that the parity-check matrix of a linear code is not unique. The dual code $\mathcal{C}^\perp$ is an $n-k$ dimensional subspace of the vector space of dimension n over the binary field. A multitude of bases exists for this space and is used to generate a collection of judiciously chosen, non-equivalent parity-check matrices for $\mathcal{C}$. Using the automorphism group of the code is one practical way to create this collection of matrices [9]. In this case, the MBBP algorithm can be understood as applying permutations to the received word to obtain different results with a fixed parity-check matrix. These permutations correspond to those elements in the automorphism group that allow to transform parity-check matrices into each other [9]. However, MBBP does not depend on the permutations given by the automorphism group. Any way to create a set of parity-check matrix representations which yield a desired decoding performance if valid for MBBP decoding. As BP decoders assume sparse matrices, only representations of (near) minimum weight are of interest. For decoding, each BP decoder performs at most I iterations on its parity-check matrix what allows for correcting different error patterns. We use this fact by combining different representations to reduce the probability of a decoding failure significantly. Several ways to combine the information are available [9]. In this correspondence we focus on the decoding scheme presented in [6]. It is our objective to extend the field of application of the MBBP algorithm to PEG codes of length up to $n = 200$. We will first outline limitations of the MBBP approach for dense cyclic codes such as those considered in [6] and then discuss limitations for PEG codes.

3.1 Limitations of MBBP decoding for dense cyclic codes

MBBP decoding requires the presence of a collection of non-equivalent parity-check matrices with good decoding properties. To decode a dense cyclic $[n,k,d]$ code $\mathcal{C}$, only parity-check matrices of cyclic form with checks corresponding to codewords of minimum weight from $\mathcal{C}^\perp$ should be considered. These representations have proven favorable for the BEC channel [6]. Many entities related to the error performance of the BEC can be transferred to the AWGN channel [10] and simulation results show the appropriateness of cyclic representations for the AWGN channel [9]. In the case of cyclic codes, each constructed parity-check matrix $\boldsymbol{H}_\ell$, $\ell \in \{1,\ldots,l\}$, of smallest possible density requires the presence of one cog of minimum possible weight. However, there are codes with a low number of cogs of this kind. One straightforward idea to overcome this would be to employ cogs of higher weight and use them to enlarge the number of matrix representations. Section 6 will show that this is not advisable for dense codes. Another approach is to modify the BP decoder and use already present matrices as additional representations for the MBBP approach. This type of decoding will be referred to as "Leak MBBP" and will be introduced in Section 4.

3.2 Limitations of MBBP decoding for PEG codes

Unlike for the codes considered in [6], the automorphism group of a PEG code is not readily available or may contain only the identity permutation. It is thus required to construct a collection of non-equivalent parity-check matrices by linear combination of codewords from the dual code. The decoding performance of the MBBP decoder depends to a great extend on the appropriateness of these matrices. We propose a construction method to generate a collection of suitable matrix representations for PEG codes. This approach is based on the knowledge of cycles in the corresponding Tanner graph and is given in Section 5.

4 Leaking

We introduce a novel variation of the BP decoder, which can be used in connection with MBBP decoding to replace

an additional matrix representation. This algorithm, called Leaking, is a general algorithm that can be used for any linear code.

4.1 Motivation

The Leaking algorithm aims at reducing the influence of short cycles. Especially dense codes contain many cycles of length $c = 4$. Leaking weakens their impact in two ways. First, in early iterations only reliable information is included in the decoding process, unreliable channel information is not passed to the variable nodes. Instead, these bits are declared as erasures, i.e. their intrinsic probabilities are set to 0.5. This can be understood as shifting the channel model from AWGN towards the BEC. The girth problem is mitigated as erasures prohibit that information is relayed at check nodes. The second impact exhibits in later iterations when high a-posteriori probabilities are available at the variable nodes. This breaks cycles as weak information is not forwarded through variable nodes where reliable information attains. With the influence of unreliable symbols and short cycles being weakened, the Leaking decoder may find the correct solution while a standard BP decoder converges to the wrong codeword and vice versa. This behavior qualifies Leaking to support an existing BP decoder and thus as an additional representation in an MBBP decoder.

4.2 The Leaking Algorithm

A standard BP decoder assigns the channel output probabilities $\Pr_\nu^{(\mathrm{C})} = \Pr(X = 0 \mid y_\nu)$, $\nu = 1, \ldots, n$ to the intrinsic probabilities $\Pr_\nu^{(\mathrm{I})}$, for the code symbols prior to starting the decoding process[1]. There, this information is used to calculate the a-posteriori probability $\Pr_\nu^{(\mathrm{APP})}$ and the extrinsic probabilities $\Pr_{\nu,\mu}^{(\mathrm{E})}$, $\mu = 1, \ldots, M$, that are exchanged between check nodes and variable nodes during the decoding process, with M denoting the number of parity-check equations.

In contrary, the proposed Leak BP decoding (or short: *Leaking*) algorithm holds back less reliable parts of this information and lets it leak gradually into the variable nodes during the decoding process. In iteration i of the decoding process, the proposed approach provides only a subset $\mathcal{L}_i \subseteq \{1, \ldots, n\}$ of variable nodes with the corresponding information $\Pr_{\nu'}^{(\mathrm{C})}$, $\nu' \in \mathcal{L}_i$ with $\mathcal{L}_i \subseteq \mathcal{L}_j$ for $i \leq j$. Channel information is leaked to the decoder by setting $\Pr_\nu^{(\mathrm{I})} := \Pr_\nu^{(\mathrm{C})}$, $\nu \in \mathcal{L}_i$. Note that this does not overwrite existing information as $\Pr_\nu^{(\mathrm{I})}$ is only employed in operations which are based on information combining [11].

We introduce two strategies to determine the sets of instructed variables $\mathcal{L}_i$ for a given iteration i. First, for the *threshold-based* approach an initial probability-based threshold needs to be set. Before performing the first iteration, the decoder determines $\mathcal{L}_1^{(\mathrm{t})}$ by checking which channel probabilities exceed a given threshold $\delta_1^{(\mathrm{t})}$, $\delta_1^{(\mathrm{t})} \geq 0.5$, or underrun $1 - \delta_1^{(\mathrm{t})}$, and in case include them in $\mathcal{L}_1^{(\mathrm{t})}$. The sets $\mathcal{L}_i^{(\mathrm{t})}$, $i > 1$, are based on $\delta_i^{(\mathrm{t})}$, which will be defined shortly. Second, the *ratio-based* approach realizes the idea to start the decoding process with a given ratio $\rho = \frac{\left|\mathcal{L}_1^{(\mathrm{r})}\right|}{n}$ of variable nodes that are instructed on their intrinsic information. From this ratio, a threshold in the probability domain, $\delta_1^{(\mathrm{r})}$, $\delta_1^{(\mathrm{r})} \geq 0.5$, is calculated such that the channel information is leaked to the fraction ρ of variable nodes which have the "strongest" information, i.e. all variable nodes in $\mathcal{L}_1^{(\mathrm{r})}$ with $\mathcal{L}_1^{(\mathrm{r})} = \{\nu \mid \max\left(\Pr(X_\nu = 0 \mid y_\nu), \Pr(X_\nu = 1 \mid y_\nu)\right) > \delta_1^{(\mathrm{r})}\}$ and $\nu = 1, \ldots, n$ are informed about their channel probabilities. To determine $\delta_1^{(\mathrm{r})}$, a straightforward approach would be to collect statistical data of the received values prior to starting the decoding process. As the statistical parameters of the AWGN channel are known and we focus on meeting the constraint $\rho \stackrel{!}{=} \frac{\left|\mathcal{L}_1^{(\mathrm{r})}\right|}{n}$ on the average, this is not necessary. It is possible to determine ρ as a function of $\delta_1^{(\mathrm{r})}$. We first define that for a given probability-domain threshold $\delta_1^{(\mathrm{r})}$, all received values y_ν with

$$|y_\nu| \geq \delta^{(\mathrm{y})} = \frac{\sigma^2}{2} \ln\left(\frac{\delta_1^{(\mathrm{r})}}{1 - \delta_1^{(\mathrm{r})}}\right) \tag{1}$$

pass their information to the decoding process, for all others $y_\nu = 0$ is assumed, thus an erasure is declared.

Using the parameters of the Gaussian distribution, it is easy to calculate the ratio of variable nodes to which intrinsic information has been leaked if a threshold $\delta^{(\mathrm{y})}$ is given.

$$\begin{aligned} \rho &= \Pr(Y \leq -\delta^{(\mathrm{y})}) + \Pr(Y \geq \delta^{(\mathrm{y})}) \\ &= Q\left(\frac{\delta^{(\mathrm{y})} - 1}{\sigma}\right) + Q\left(\frac{\delta^{(\mathrm{y})} + 1}{\sigma}\right) \end{aligned} \tag{2}$$

with $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty \mathrm{e}^{-t^2/2}\,\mathrm{d}t$, $x \in \mathbb{R}$

Using Equation (1) and Equation (2), one obtains

$$\begin{aligned} \rho &= Q\left(\frac{\sigma}{2} \ln\left(\frac{\delta_1^{(\mathrm{r})}}{1 - \delta_1^{(\mathrm{r})}}\right) - \frac{1}{\sigma}\right) \\ &\quad + Q\left(\frac{\sigma}{2} \ln\left(\frac{\delta_1^{(\mathrm{r})}}{1 - \delta_1^{(\mathrm{r})}}\right) + \frac{1}{\sigma}\right). \end{aligned} \tag{3}$$

With knowledge of the transmission parameters, Equation (3) can be solved for $\delta_1^{(\mathrm{r})}$ prior to decoding.

Once $\delta_1^{(\mathrm{t})}$ is set or $\delta_1^{(\mathrm{r})}$ is determined from ρ, the threshold is decreased linearly with i by $\delta_i^{(\mathrm{t/r})} = \max\left(\delta_1^{(\mathrm{t/r})}\left(1 - \frac{i-1}{I'}\right), 0.5\right)$, $1 < i \leq I$, where I denotes the total number of iterations and I' is chosen such that (almost) all channel information is included at the end of the decoding process. A detailed description of the Leak BP decoding algorithm is given in Algorithm 1.

Let us state that Leaking is a suboptimal decoding approach as information is retained from the decoder for a given number of iterations. But as BP is suboptimal itself, this method may help to lower the girth problem. It was observed that numerous bits are erroneously decided when a frame is decoded incorrectly or the decoder does not converge [12].

[1] W.l.o.g. all probabilities are given w.r.t. $X = 0$ if not stated elsewise.

Algorithm 1 Leak BP decoding

Input: $\boldsymbol{y}$, $\boldsymbol{H}$, I, I', $\delta_1^{(\mathrm{t})}$ or ρ
Output: $\hat{\boldsymbol{c}}$

// **Initialization Phase**
Calculate vector $[\delta_i^{(\mathrm{t/r})}]$ from I' and $\delta_1^{(\mathrm{t})}$ or ρ,$i = 1, \ldots, I$
// **Processing Phase**
for $i = 1, \ldots, I$ **do**
  for $\nu = 1, \ldots, n$ **do**
    $\Pr_\nu^{(\mathrm{C})} = \Pr(X_\nu = 0 \mid y_\nu)$
    if $\Pr_\nu^{(\mathrm{C})} \geq \delta_i^{(\mathrm{t/r})} || \Pr_\nu^{(\mathrm{C})} \leq 1 - \delta_i^{(\mathrm{t/r})}$ **then**
      $\Pr_\nu^{(\mathrm{I})} = \Pr_\nu^{(\mathrm{C})}$
    else
      $\Pr_\nu^{(\mathrm{I})} = 0.5$
    end if
  end for
  $\{[\Pr_\nu^{(\mathrm{APP})}], [[\Pr_{\nu,\mu}^{(\mathrm{E})}]]\} = \mathrm{BPiter}([\Pr_\nu^{(\mathrm{I})}], [[\Pr_{\nu,\mu}^{(\mathrm{E})}]])$, $\nu = 1, \ldots, N$, $\mu = 1, \ldots, M$
  if $\mathrm{round}([\Pr_\nu^{(\mathrm{APP})}]) \cdot \boldsymbol{H}^{\mathrm{T}} == \boldsymbol{0}$ **then**
    $\hat{\boldsymbol{c}} = \lfloor [\Pr_\nu^{(\mathrm{APP})}] \rceil$, **break**; $\nu = 1, \ldots, N$
  end if
end for

But, Leaking has proven to be a valuable addendum to MBBP decoding what will be confirmed by simulation results.

The following section specifies a method which allows to create a collection of parity-check codes which are required to extend MBBP to PEG-optimized codes.

5 Matrix collections for PEG codes

Good linear block codes of moderate length $n \leq 200$ and rates close to $1/2$ can be obtained by the PEG algorithm. PEG is a greedy algorithm which optimizes the cycle distribution [13] of a code according to a given degree distribution [1]. Good codes are usually obtained by using optimized degree distributions. These distributions are obtained by *density evolution* [1] and can be found at [14]. The performance of such a distribution is classified by its threshold, i.e. the required signal-to-noise ratio to under-run any (frame) error-rate for unlimited code length.

As the actual performance of moderate length codes greatly differs from the threshold, a set of degree distributions with different maximum degrees was used to construct PEG codes up to length 200. The required signal-to-noise ratios to obtain predefined error rates (bit error rate $\mathrm{BER} = 10^{-5}$ and frame error rate $\mathrm{FER} = 10^{-3}$) were used as a criterion to compare the performance. Extensive computer simulations showed that optimized degree distributions with moderate maximum degree (cf. Section 6) lead to the best performance for PEG codes of length $n \leq 200$.

Random number generators are used in the PEG algorithm, thus it is straightforward to see that the automorphism group of such a code is in general unknown. The convenient approach to generate a set of appropriate representations through permutations fails in this case. A first step towards a collection of parity-check matrices is to construct a single matrix, denoted by $\boldsymbol{H}_{\mathrm{PEG}}^{(1)}$, that does not coincide with the original PEG matrix, $\boldsymbol{H}_{\mathrm{PEG}}^{(0)}$. A straightforward approach is to replace a set $\mathcal{R} \subseteq \{1, \ldots, n-k\}$ of parity-check equations with different parity-check equations of the same weight, which requires information on additional codewords of the dual code. For short PEG codes ($n \leq 50$) of rate $1/2$, most of these codewords are already present in $\boldsymbol{H}_{\mathrm{PEG}}^{(0)}$, for longer codes the required information on the weight distribution [8] can not be obtained. A second method is to generate linear combinations of parity-check equations randomly and use them to replace existing parity-check equations while ensuring the full rank of the generated matrix. This usually leads to high-weight parity-check equations and to poor performance results for BP decoding. Using these matrices besides $\boldsymbol{H}_{\mathrm{PEG}}^{(0)}$ for MBBP decoding would not involve performance improvements.

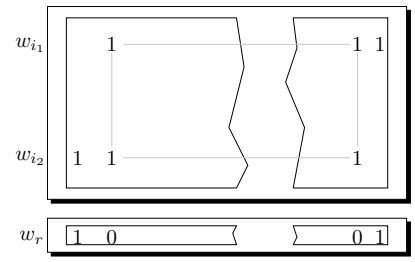

Fig. 1. Construction of a redundant check equation for $c = 4$

A promising construction applicable to codes of moderate length is given in the following. Let the indices of parity-checks which are connected to each other by at least one cycle of length c form the sets $\mathcal{G}_{c,\tau}$, $\tau = 1, \ldots, T$. Linear combination of $c/2$ checks indexed by the elements in $\mathcal{G}_{c,\tau}$ leads to a redundant check of weight $w_{\mathrm{r}} = \sum_{i \in \mathcal{G}_{c,\tau}} w_i - c$, where w_i is the weight of the i-th parity-check equation of $\boldsymbol{H}_{\mathrm{PEG}}^{(0)}$. This check can replace a check indexed by one element of $\mathcal{G}_{c,\tau}$ in $\boldsymbol{H}_{\mathrm{PEG}}^{(0)}$ while the matrix keeps full rank. Repeated application of these steps for different values of τ allows a whole set of checks to be replaced and to generate a further matrix representation. Figure 1 shows the generation of a redundant check for $c = 4$.

The PEG codes considered in this paper usually deploy codewords of the dual code of weight $w_i = 6$ to $w_i = 8$ $i = 1, \ldots, n-k$. Consequently, the proposed strategy leads to a set of equations with $w_{\mathrm{r}} \approx w_i$, $i = 1, \ldots, n-k$ for $c = 4$ and $c = 6$.

6 Results

This section presents simulation results for the $[63, 39, 9]$-BCH code as well as PEG codes of rate $1/2$. To assess the presented results more generally, they will be compared to the *Gallager* bound [15]. It is well known that this bounding technique leads to desirable results which allow us to value the excellence of the proposed scheme.

6.1 $[63, 39, 9]$-BCH code $\mathcal{B}$

Let $\mathcal{B}$ denote the $[63, 39, 9]$-BCH code. Its dual code $\mathcal{B}^\perp$ has 450 codewords of minimum weight 14 which lead to only 6 cyclic orbit generators that can be used to construct $n \times n$ parity-check matrices of cyclic form and full rank. It is a straightforward idea to search for codewords of higher weight

within $\mathcal{B}^{\perp}$ to increase the number of parallel representations. In this case it turns out that the codewords of weight 18 are appropriate to construct matrices of full rank.

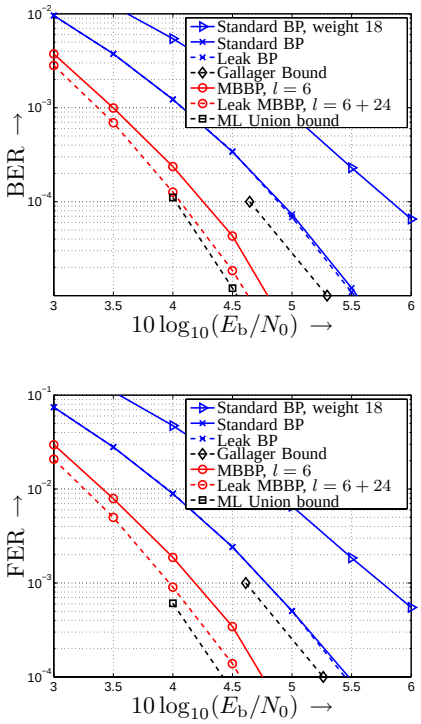

Fig. 2. BER and FER performance, $\mathcal{B}$ ($[63, 39, 9]$-BCH)

Figure 2 shows simulation results for $\mathcal{B}$, where BER and FER are considered as performance measures, respectively. Observe first that the representations of row-weight 18 obtain poor frame error rates. This leads to a high number of average iterations for the BP decoder and disqualifies these representations for MBBP, especially because the maximum number of iterations required by the decoders is of interest for MBBP decoding. The results for Leak BP decoding show that this approach can obtain a performance comparable to standard BP. Next, observe that usual MBBP decoding with $l = 6$ parity-check matrix representations achieves a desirable performance but still leaves a gap of 0.25 dB to the ML union bound. Leak MBBP, i.e. considering the decoders performing leaking as additional representations of the MBBP approach, allows to approach the ML union bound very closely. To yield the results presented in Figure 2, another feature of Leak MBBP was used. More than 6 additional decoding units were employed by using different values for ρ. In total, $l = 30$ representations with $I = 100$, $I' = 300$ were used for the presented results.

6.2 PEG-optimized codes

We present simulation results for PEG codes of rate $1/2$ and length $n = 50$, termed $\mathcal{P}_1$, and a PEG code $\mathcal{P}_2$ of length $n = 200$. Also, a PEG code of length $n = 100$ is considered, but not discussed in detail. All codes follow the degree distribution [14] given from node perspective for variable nodes in Eq. (4).

$$\begin{aligned} L(x) \;=\; & 0.5043865558 \cdot x^2 + 0.2955760529 \cdot x^3 + \\ & 0.0572634080 \cdot x^5 + 0.0362602194 \cdot x^6 + \\ & 0.0049622081 \cdot x^7 + 0.0292344776 \cdot x^9 + \\ & 0.0650312477 \cdot x^{11} + 0.0072858305 \cdot x^{12} \end{aligned} \quad (4)$$

These short codes contain cycles of length 4 and 6 what allows the use of the construction method specified in Section 5 for both $c = 4$ and $c = 6$.

6.2.1 $[50, 25]$-PEG code $\mathcal{P}_1$

The code $\mathcal{P}_1$, constructed with the PEG algorithm, has minimum distance $d = 5$ and its dual has minimum distance $d^{\perp} = 6$. The presented construction scheme promises redundant parity-check equations of weight $w_r = 8$ to $w_r = 10$ for $c = 4$ and weight $w_r = 12$ to $w_r = 15$ for $c = 6$. A collection of 19 $n - k \times n$ parity-check matrices is created by the construction scheme presented in Section 5. For each matrix we specify how many parity-checks of $\boldsymbol{H}_{\text{PEG}}^{(0)}$ were replaced to obtain the current representation as well as the length of the cycles c that were considered to obtain the redundant checks. The matrix collection consists of $\boldsymbol{H}_{\text{PEG}}^{(0)}$ and

- 7 matrices, each with $q = 10$ rows replaced, $c = 4$,
- 1 matrix with $q = 20$ rows replaced, $c = 4$,
- 10 matrices, each with $q = 5$ rows replaced, $c = 6$.

We will compare the results for standard BP decoding to those obtained by MBBP decoding and MBBP decoding in connection with Leaking, where $\rho = 0.9$ is chosen as a start parameter and $I' = 200$, $I = 100$ are set.

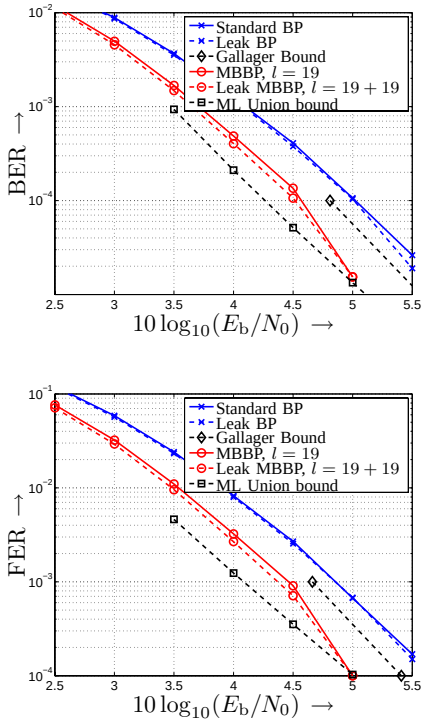

Fig. 3. BER and FER performance, $\mathcal{P}_1$ ($[50, 25]$-PEG)

The obtained gain of MBBP over BP decoding is smaller than for the code $\mathcal{B}$, but it is still significant, cf. Figure 3.

This can be attributed to the fact that PEG codes are already optimized for BP decoding. Again, Leak MBBP allows to approach the ML bound.

6.2.2 $[200, 100]$-PEG code $\mathcal{P}_2$

We determine an upper bound on the minimum distance of $\mathcal{P}_2$ and its dual and state that $d \leq 14$, $d^{\perp} \leq 6$. For this code, 26 matrices of dimension $n - k \times n$ are found in total. Due to a low number of cycles of length 4, the collection includes besides $\boldsymbol{H}_{\mathrm{PEG}}^{(0)}$

- 4 matrices, each with $q = 2$ rows replaced, $c = 4$,
- 1 matrix with $q = 5$ rows replaced, $c = 4$,
- 20 matrices, each with $q = 10$ rows replaced, $c = 6$.

Again, a significant performance improvement by MBBP is observed, with an additional gain when using Leak MBBP, where $\rho = 0.9$ and $I' = 300$, $I = 200$ is set. Figure 4 shows the performance results in terms of BER and FER, respectively.

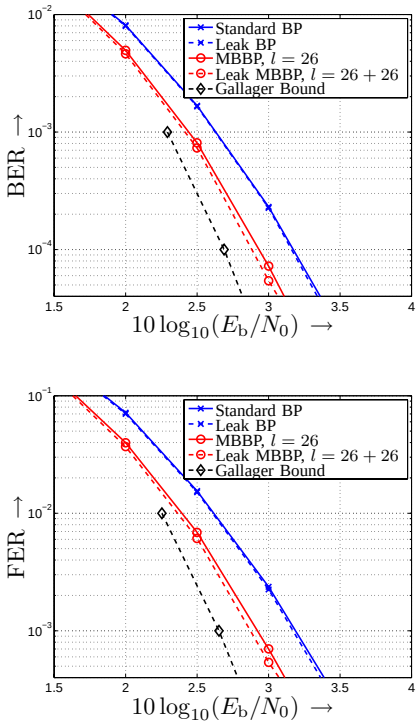

Fig. 4. BER and FER performance, $\mathcal{P}_2$ ($[200, 100]$-PEG)

6.2.3 PEG codes - power efficiency vs. code length

We compare $\mathcal{P}_1$, $\mathcal{P}_2$, as well as a PEG code of similar properties and length $n = 100$ for their power efficiency, measured as the required signal-to-noise ratio to obtain a given frame error rate of FER $= 10^{-3}$. Figure 5 compares standard BP decoding, MBBP decoding, and Leak MBBP. The simulations use all representations found and the ratio-based approach for Leaking with $\rho = 0.9$. It is to observe that the Gallager bound is outperformed for very short code lengths ($n \approx 50$). For codes of moderate length, it can be approached quite closely.

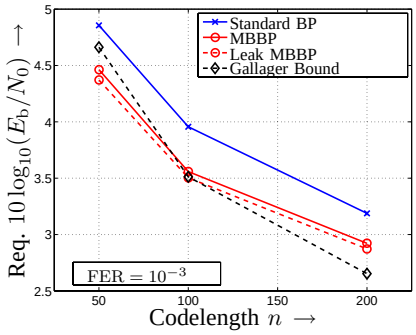

Fig. 5. Required signal-to-noise ratio to obtain FER $= 10^{-3}$

7 Conclusions

MBBP decoding can achieve performance improvements for dense algebraic codes and PEG codes of moderate length. The automorphism group of the code can be used but is not required. This is emphasized by two novel methods, applicable to any linear code, which allow to create a set of matrices without knowledge of this group. Besides its application for PEG codes, Leaking can be used as an addendum for dense codes where the number of minimum-weight codewords limits the performance of MBBP.

References

[1] S.-Y. Chung, G. Forney, T. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Communications Letters*, vol. 5, no. 2, pp. 58–60, February 2001.

[2] J. Jiang and K. Narayanan, "Iterative soft decision decoding of Reed Solomon codes based on adaptive parity check matrices," in *Proc. IEEE Int. Symp. on Inform. Theory*, 2004, p. 261.

[3] A. Kothiyal, O. Y. Takeshita, W. Jin, and M. Fossorier, "Iterative reliability-based decoding of linear block codes with adaptive belief propagation," *IEEE Communications Letters*, vol. 9, no. 12, pp. 1067–1069, December 2005.

[4] M. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. Inform. Theory*, vol. 41, pp. 1379–1396, September 1995.

[5] T. Halford and K. Chugg, "Random redundant soft-in soft-out decoding of linear block codes," in *Proceedings of Int. Symp. on Inform. Theory (ISIT)*, Seattle, WA, July 2006, pp. 2230–2234.

[6] T. Hehn, J. Huber, S. Laendner, and O. Milenkovic, "Multiple-bases belief-propagation decoding for short block-codes," in *Proceedings of Int. Symp. on Inform. Theory (ISIT)*, Nice, France, June 2007, pp. 311–315.

[7] X.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth Tanner graphs," *IEEE Transactions on Information Theory*, vol. 51, pp. 386–398, January 2005.

[8] F. MacWilliams and N. Sloane, *The Theory of Error-Correcting Codes*. North-Holland Publishing Company, 1977.

[9] T. Hehn, J. Huber, O. Milenkovic, and S. Laendner, "Multiple-bases belief-propagation for decoding of dense cyclic codes," *Preprint*, 2007.

[10] J. Feldman, "Decoding error-correcting codes via linear programming," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, 2003. [Online]. Available: http://www.columbia.edu/∼jf2189/pubs.html

[11] I. Land and J. Huber, "Information combining," *Foundations and Trends in Communications and Information Theory*, vol. 3, no. 3, pp. 227–330, November 2006.

[12] P. He, "Modified belief-propagation decoding," Master's thesis, University of Erlangen-Nuremberg, Erlangen, Germany, August 2007.

[13] Y. Mao and A. Banihashemi, "A heuristic search for good low-density parity-check codes at short block lengths," in *Proc. IEEE Int. Conf. on Communications (ICC)*, vol. 1, Helsinki, Finland, June 2001, pp. 41–44.

[14] R. Urbanke, "LdpcOpt - a fast and accurate degree distribution optimizer for LDPC ensembles. http://lthcwww.epfl.ch/research/ldpcopt/index.php."

[15] G. C. Clark and J. B. Cain, *Error-Correction Coding for Digital Communications*. Plenum Press, New York, 1981.