Multiple-Bases Belief-Propagation Decoding of High-Density Cyclic Codes

Thorsten Hehn[‡], Johannes B. Huber[‡], Olgica Milenkovic[†], Stefan Laendner[‡]
[‡] Institute for Information Transmission (LIT)
FAU Erlangen-Nuremberg, Germany
[†] Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign, USA

***Abstract*—We introduce a new method for decoding short and moderate-length linear block codes with dense parity-check matrix representations of cyclic form. This approach is termed multiple-bases belief-propagation. The proposed iterative scheme makes use of the fact that a code has many structurally diverse parity-check matrices, capable of detecting different error patterns. We show that this inherent code property leads to decoding algorithms with significantly better performance when compared to standard belief-propagation decoding. Furthermore, we describe how to choose sets of parity-check matrices of cyclic form amenable for multiple-bases decoding, based on analytical studies performed for the binary erasure channel. For several cyclic and extended cyclic codes, the multiple-bases belief-propagation decoding performance can be shown to closely follow that of the maximum-likelihood decoder.**

Index Terms: Algebraic Codes, Belief Propagation, Multiple-Bases Belief-Propagation Decoding, Stopping Sets.

I. Introduction

Classical algebraic codes of short block length have large minimum distance and efficient soft-decision decoding algorithms [2], [3]. Consequently, these codes represent a good choice for low-delay applications where high transmission reliability is required. Algebraic codes are also frequently used as components of product codes and parts of concatenated coding schemes. In modern storage and communication systems, low-density parity-check (LDPC) codes are most often used as the inner coding scheme. For this reason it is desirable to implement a soft-input soft-output decoder for algebraic codes as a belief-propagation (BP) algorithm. This is a reasonable choice as the decoder can handle both types of codes. Since algebraic block codes have dense parity-check matrices with a large number of short cycles [4], [5], Standard BP decoders offer poor error-correcting performance for such codes.

The use of redundant parity-check matrices for iterative decoding schemes when signaling over the binary erasure channel (BEC) has been excessively studied. Several authors proposed using a high number of redundant checks [6], [7], [8] to improve the decoding performance. This type of decoding has also drawn the attention of researchers who studied this concept in the context of the AWGN channel. Other authors proposed *adaptive BP algorithms* [9], [10], which iteratively optimize the matrix representation during the decoding process. Such schemes have large implementation complexity due to the required matrix reduction after each iteration. The *random redundant decoding* (RRD) [4] algorithm does not require this type of processing and obtains very promising results. This is accomplished by serially altering the applied matrix representation within the decoding process. Another closely related approach was described in [11], where a simple simulation-based study was performed using randomly chosen parity-check matrices of the $[24, 12, 8]$ extended Golay code. This approach offers poor performance when compared to the algorithms described in this work.

The approach followed in this paper draws upon the prior work of the authors on BEC decoding [12] and introduces a novel decoding method that operates in parallel and iterative fashion on a collection of parity-check matrices. Using this set of decoder representations, the algorithm performs joint output processing in order to estimate the transmitted codeword. This output processing can occur at various stages of decoding and it may have various degrees of complexity. We distinguish between techniques where the BP algorithms run separately and compare them to schemes where the decoders are allowed to exchange information. We investigate processing of the form of *metric and complexity selection*, *averaging of probabilities* [13], *information combining* [14], as well as certain reliability-based schemes.

As the different representations of the parity-check matrix form bases of the dual code, we refer to the proposed approaches as *Multiple-Bases Belief-Propagation* (MBBP) decoding. For the purpose of MBBP decoding, one needs to identify classes of parity-check matrices that *jointly* offer good decoding performance. In order to accomplish this task, we propose using parity-check matrix design techniques originally developed for the BEC but subsequently used for signaling over the AWGN channel. It was observed that this approach leads to good performance results [15]. Moreover, this method relies on the fact that the performance of a parity-check matrix for the BEC is completely characterized by combinatorial entities termed stopping sets [16]; and, that pseudocodewords for linear programming decoders of several classes of channels represent stopping sets for any channel in the Tanner

Parts of the results were presented at the IEEE International Symposium on Information Theory (ISIT), Nice, France, 2007. The work was supported in part by the NSF grant CCF 0644427 awarded to Olgica Milenkovic. Note: A full-size paper corresponding to this letter can be found online [1].

graph [17], [18]. Although we focus our attention on parity-check matrix construction techniques for cyclic codes, the described concepts can be generalized for other classes of codes as well.

The main differences between the existing RRD algorithm and the proposed MBBP scheme are that RRD operates in a *serial* fashion in terms of periodically permuting the received word, while MBBP works in a parallel manner. Further, the RRD algorithm uses message scaling processing between iterations that tends to increase the overall complexity of the scheme. Contrary, the MBBP algorithm deploys the standard update rules defined by message passing decoding. Finally, the RRD algorithm uses a greedy search over the Tanner graphs to find a representation which offers good performance. Further it deploys random techniques to select the permutations used to interchange the positions of the variable nodes. The MBBP algorithm relies on specially designed parity-check matrix families. Transformations from one element of a family to another one and between families, respectively, are done by means of algebraic rules.

The paper is organized as follows. Section II introduces relevant definitions and terminology, while Section III contains a description of the bases selection process. Section IV presents a set of different variations of MBBP decoding algorithms. Simulation results are presented in Section V.

II. Definitions and Terminology

We start by introducing the terminology related to stopping sets and the BEC. We also provide the terminology needed for describing the MBBP decoding approach.

Definition II.1. *Let $\boldsymbol{A}$ be an $m \times n$ matrix, and let the columns of $\boldsymbol{A}$ be indexed by the set of integers $\mathcal{J} = \{0, \ldots, n-1\}$. For a set $\mathcal{I} \subseteq \mathcal{J}$, we define the restriction of $\boldsymbol{A}$ to $\mathcal{I}$ as the $m \times |\mathcal{I}|$ array of elements composed of the columns of $\boldsymbol{H}$ indexed by $\mathcal{I}$.*

When transmitting over the BEC, stopping sets completely determine the failure modes of iterative decoders. For completeness we define these sets below [16].

Definition II.2. *For a given parity-check matrix $\boldsymbol{H}$ of an $[n, k, d]$ binary linear code $\mathcal{C}$, a stopping set $\mathcal{S}(\boldsymbol{H})$ of size σ is a set $\mathcal{I}$ of σ positions in the codeword for which the restriction of $\boldsymbol{H}$ to $\mathcal{I}$ does not contain rows of Hamming weight one.*

Clearly, a codeword is a stopping set and consequently the size of the smallest stopping set of any given parity-check matrix cannot exceed d.

In order to compare different parity-check matrix representations with respect to their decoding performance, we restrict our attention to a simple evaluation criteria: the number of stopping sets of size less than or equal to σ, for some predefined value $1 \leq \sigma \leq d$. Given a parity-check matrix $\boldsymbol{H}$, the number of its stopping sets of size σ will henceforth be denoted by $|\mathcal{S}_\sigma(\boldsymbol{H})|$. Although stopping sets are known to completely characterize the performance of iterative decoders *only* for the BEC, they also represent a partial performance indicator for transmission over the AWGN channel. This is due to the intimate connection between stopping sets and pseudocodewords [17], [19].

As we focus our attention on codes with parity-check matrices of cyclic form, a code category that includes cyclic codes, we also provide the following definitions.

Definition II.3. *Let $\mathcal{C}$ be a binary, linear code and $\mathcal{C}^\perp$ its dual. A parity-check matrix of $\mathcal{C}$ is said to be of cyclic form if it consists of $n - k \leq m \leq n$ consecutive cyclic shifts of one chosen codeword of $\mathcal{C}^\perp$. The qualifier "consecutive" implies that the $(i+1)$-th row of the parity-check matrix, $1 \leq i \leq m-1$, is the cyclic right shift of the i-th row by one position. A code $\mathcal{C}$ is called cyclic if any cyclic shift of a codeword $\boldsymbol{c} \in \mathcal{C}$ is also a codeword, and it necessarily has at least one parity-check matrix of cyclic form.*

For a code with at least one parity-check matrix of cyclic form, we introduce the notion of a partition of the set of codewords of $\mathcal{C}^\perp$ and the notion of a cyclic orbit generator (*cog*).

Definition II.4. *Let $\mathcal{C}$ be a binary, linear, cyclic code. Partition the set of codewords of $\mathcal{C}^\perp$ into disjoint orbits (subsets) consisting of cyclic shifts of one codeword. Let one designated codeword in the orbit be the representative of the subset. This codeword is referred to as the cyclic orbit generator (cog).*

Throughout the paper we focus our attention on cogs of minimum Hamming weight. Little technical modifications are required in the above definition to encompass parity-check matrices that are of cyclic form when restricted to a proper subset of columns, e.g. extended cyclic codes.

Let $\mathcal{G}$ be the set of cyclic orbit generators with Hamming weight equal to the minimum distance of the dual code, $d^\perp$. A cyclic orbit generator $\mathrm{cog}_\ell \in \mathcal{G}$, $\ell = 1, \ldots, |\mathcal{G}|$, is used to construct a parity-check matrix $\boldsymbol{H}_\ell$, $\ell = 1, \ldots, |\mathcal{G}|$, of size $m \times n$, $n - k \leq m \leq n$, such that the row-rank of the matrix is $n - k$[1]. This matrix consists of m consecutive right shifts of cog_ℓ. To avoid identical rows in $\boldsymbol{H}_\ell$, even if $m = n$ holds, only cogs with a period of n, i.e. cogs with a cyclic orbit that consists of n distinct shifts, are considered for the construction process.

Note that a redundant parity-check matrix of cyclic form can have at most n distinct rows. The larger the value of m, the larger the hardware implementation complexity of the BP decoder. Nevertheless, based on extensive computer simulations, it was observed that for decoding of algebraic codes signaled over the AWGN channel the best decoding performance is achieved for $m = n$. This finding holds for both the *bit error rate* (BER) and *frame error rate* (FER). The reason supporting this observation is intuitively clear. Consider a parity-check matrix of cyclic form for which $m = n - k$, as shown in (1) for $m = 3, n = 7$. Here, the symbol x serves as

[1] Here, and throughout the paper, we only consider cogs that generate parity-check matrices with row-rank $n - k$. As a consequence, we use the word *bases* to describe the underlying matrices, although the considered structures are actually *frames*. Frames are over-complete systems in which every element of a vector space can be represented in a not necessarily unique manner [20].

placeholder for the bits of the generating cog of the matrix.

$$\left(\begin{array}{cc|ccc|cc} 1 & x & x & x & 1 & 0 & 0 \\ 0 & 1 & x & x & x & 1 & 0 \\ 0 & 0 & 1 & x & x & x & 1 \end{array}\right) \tag{1}$$

As can be seen from (1), not all of the seven bits participate in the same number of parity-check equations - the column degrees of the parity-check matrix vary with the column. There exist at least two bits (including the first and last) that participate in only one parity-check, and therefore have very low probability of being correctly decoded. Depending on the particular choice of the cog, the set of symbols at the beginning and at the end of the codeword is strongly restricted with respect to the maximum number of parity-checks it can participate in. This problem can be solved by setting $m = n$: such a row-redundancy allows for achieving equal error protection for all code symbols [21]. Computer simulations show that for the codes considered in this work and in [1], this effect outbalances the performance degradation by additional cycles. Therefore, square parity-check matrices will be used throughout the remainder of this paper.

We conclude this section by introducing the notion of a cog *family* of a set of parity-check matrices.

Definition II.5. *Let $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_F$ be a partition of the set of indices $\{1, \ldots, |\mathcal{G}|\}$, so that the "stopping set performance" of $\boldsymbol{H}_\ell$ is fixed within each family $\mathcal{F}_f$, for all $\ell \in \mathcal{F}_f$, and for all $f \in \{1, \ldots, F\}$. The "stopping set performance" of a parity-check matrix, defined for both the BEC and AWGN channel, is the number of stopping sets of size up to and including d. We refer to the set $\{\mathrm{cog}_\ell\}$, $\ell \in \mathcal{F}_f$, as the f-th cog family.*

III. Bases Selection for MBBP decoding

Recall that a linear $[n, k, d]$ code $\mathcal{C}$ is uniquely defined by a parity-check matrix $\boldsymbol{H}$ of rank $n - k$ or a generator matrix $\boldsymbol{G}$ of rank k. There usually exists a large number of generator and parity-check matrices for the same code. For BP decoding over AWGN channels, one usually seeks a sparse parity-check matrix $\boldsymbol{H}$.

Adding redundant rows to parity-check matrices improves the performance of iterative decoding for the BEC, but usually has detrimental effects on BP decoders correcting data signaled over the AWGN channel. This can be attributed to the increase of the number of short cycles and the density of the matrix. But, as was shown by the authors in [22], adding judiciously chosen redundant rows may increase the minimum weight of pseudocodewords (and trapping sets) of the given parity-check matrix.

In order to exploit the benefits offered by redundant parity-check matrices with respect to pseudocodeword performance, while controlling the negative effects on the cycle lengths, the following approach can be used. Rather than decoding a received word in terms of only one parity-check matrix, one can use a collection of parity-check matrices, each with small row-redundancy, in parallel. The results of the decoders operating on different parity-check matrices can then be appropriately combined. This is the main idea behind MBBP decoding, and for this purpose, we propose to develop good heuristic techniques for identifying parity-check matrices which offer both good individual and joint decoding performance.

For cyclic algebraic codes, we proved in a companion paper [12] that parity-check matrices that consist of cyclic shifts of carefully chosen cogs offer excellent stopping set properties. In what follows, we focus on identifying families of cogs that obtain very good decoding performance for the AWGN channel. The iterative decoding performance of a fixed parity-check matrix used over the AWGN channel is, to a certain extent, strongly correlated with its BEC performance. Hence, we use the total number of stopping sets of size up to d as our cog family optimization criteria.

Since cogs from the same family define parity-check matrices of cyclic form with identical properties for the BEC and comparable properties for the AWGN channel, it is desirable to identify the family with the best performance and then exclusively use cogs from this family to build matrices for MBBP decoding. Identifying all families along with its members by counting stopping sets in the corresponding matrices is computationally expensive [23]. Also, storing all cogs used for decoding can be prohibitively costly. In order to avoid these problems, we propose to use a cog *mapping technique* that relies on the notion of the *automorphism group* of a code.

Definition III.1. *[24, Ch. 8] The permutations which send $\mathcal{C}$ into itself, i.e. codewords go into (possibly different) codewords, form the automorphism group of the code $\mathcal{C}$, denoted by $\mathrm{Aut}(\mathcal{C})$. If $\mathcal{C}$ is a linear code and $\mathcal{C}^\perp$ is its dual code, then $\mathrm{Aut}(\mathcal{C}) = \mathrm{Aut}(\mathcal{C}^\perp)$.*

It is straightforward to prove that there exists a set of permutations $\mathcal{P}$ in the automorphism group of a cyclic code which map one cog into another cog from the same family. Fixing one cog, and then applying a subset of permutations from $\mathcal{P}$ to it, represents a convenient way for generating redundant parity-check matrices with identical densities and comparable performance under MBBP decoding.

We provide next a partial characterization of the set $\mathcal{P}$ for cyclic codes. More precisely, we describe how to find a large set of permutations $\mathcal{P}$ for which $|\mathcal{S}_\sigma(\boldsymbol{H}_a)| = |\mathcal{S}_\sigma(\boldsymbol{H}_b)|$, for $\sigma \leq n$, where the generating cogs of $\boldsymbol{H}_a$ and $\boldsymbol{H}_b$ satisfy $\mathrm{cog}_b = \kappa(\mathrm{cog}_a)$, and where $\kappa(\cdot) \in \mathcal{P}$. Here, $\kappa(\mathrm{cog}_a)$ is used to denote the action of the permutation κ on the vector cog_a. Note that the same framework can be used when only stopping sets up to a size smaller than or equal to d are considered.

It is well known that the automorphism group of a cyclic code contains two classes of permutations [24]. The first class, referred to as P_1, contains cyclic permutations $\alpha^0, \alpha^1, \ldots, \alpha^{n-1}$, where $\alpha : i \rightarrow (i+1) \bmod n$, $i = 0, \ldots, n-1$, or, equivalently, $\alpha(\boldsymbol{c}) = (c_{n-1}, c_0, c_1, \ldots, c_{n-2})$ holds. The second class, denoted by P_2, is the class of permutations $\beta^0, \beta^1, \ldots, \beta^{h-1}$, where $\beta : i \rightarrow (2 \cdot i) \bmod n$, $i = 0, \ldots, n-1$. Alternatively, this can be specified by $\beta(\boldsymbol{c}) = (c_0\, c_{(n+1)/2}\, c_1\, \ldots c_{(n-1)/2})$. Here, h denotes the cardinality of the cyclotomic coset of the n-th roots of unity that contains one. Above, all subscripts are taken modulo n. For extended cyclic codes, the described permutations are only to be applied to the cyclic part of the codeword.

We find the following definition useful for our subsequent derivations.

Definition III.2. *Let κ be a permutation operating on a vector $\boldsymbol{v}$, resulting in a vector $\boldsymbol{t} = \kappa(\boldsymbol{v})$. We define the κ-permutation of an $m \times n$ matrix $\boldsymbol{V}$ as a matrix obtained by permuting each row of $\boldsymbol{V}$ according to κ. In this setting, $\boldsymbol{T} = \kappa(\boldsymbol{V})$ is used to denote $\boldsymbol{T}(\mu, :) = \kappa(\boldsymbol{V}(\mu, :))$, $\mu = 1, \ldots, m$, where $\boldsymbol{X}(\mu, :)$ represents the μ-th row of the matrix $\boldsymbol{X}$.*

The following straightforward results provide a partial characterization of the set of permutations $\mathcal{P}$ of cyclic codes. All proofs rely on the fact that $\alpha^j \, \theta = \theta \, \alpha^{j'}$, for any integer j and some integer j', and for $\theta \in \mathcal{A}(n)$, the affine group of order n.

Lemma III.3. *A necessary and sufficient condition for*

$$\theta(\alpha^j(\mathrm{cog}_a)) = \alpha^{j'}(\theta(\mathrm{cog}_a))$$

for all $j \in \{0, \ldots, n-1\}$ and some $j' \in \{0, \ldots, n-1\}$ to hold is that $\theta(\cdot)$ is an affine permutation. A permutation of that kind is defined as

$$\theta : i \rightarrow q \cdot i + \omega \mod n,$$

for some $q \in \{0, \ldots, n-1\}$, $\omega \in \{0, \ldots, n-1\}$, and $i = 0, \ldots, n-1$, such that $\gcd(q, n) = 1$. Here, $\gcd(q, n)$ denotes the greatest common divisor of q and n.

Proof: The claim of Lemma III.3 can be rewritten as

$$\theta(i + j) = \theta(i) + j' \mod n,$$

with $j \in \{0, \ldots, n-1\}$, $j' \in \{0, \ldots, n-1\}$, $i = 0, \ldots, n-1$, and where $\theta(i)$ denotes the action of the permutation θ on the coordinate i.

The former equality is true if and only if $\theta(i)$ is a linear function of the form $q \cdot i + \omega$ for which $\gcd(q, n) = 1$. Consequently, there exists a one-to-one correspondence between j and j'. ■

The lemma asserts that cyclic permutations commute (up to a cyclic shift) with all affine permutations in a symmetric group.

Claim III.4. *If $\mathrm{cog}_b = \alpha^j(\mathrm{cog}_a)$, for some $j \in \{0, \ldots, n-1\}$, then $|\mathcal{S}_\sigma(\boldsymbol{H}_a)| = |\mathcal{S}_\sigma(\boldsymbol{H}_b)|$, for all $\sigma \leq n$.*

Proof: Applying α^j to cog_a cyclically permutes the rows of $\boldsymbol{H}_a$. This cyclic permutation preserves all stopping sets, which proves the claimed result. ■

Claim III.5. *If $\mathrm{cog}_c = \beta^j(\mathrm{cog}_a)$, for some $j \in \{0, \ldots, h-1\}$, then $|\mathcal{S}_\sigma(\boldsymbol{H}_a)| = |\mathcal{S}_\sigma(\boldsymbol{H}_c)|$, for all $\sigma \leq n$.*

Proof: It is straightforward to see that

$$\begin{aligned} \boldsymbol{H}_c(\mu, :) &= \alpha^{\mu-1}(\boldsymbol{H}_c(1, :)) \\ &= \alpha^{\mu-1}(\beta^j(\boldsymbol{H}_a(1, :))) \\ &= \beta^j(\alpha^{\mu'-1}(\boldsymbol{H}_a(1, :))), \end{aligned} \tag{2}$$

since β^j is an affine permutation with $q = 2^j$ and $\omega = 0$. As a result, $\boldsymbol{H}_c$ can be transformed into $\boldsymbol{H}_a$ through row- and column-permutations. ■

We conclude that $\mathcal{A}(n) \cap \mathrm{Aut}(\mathcal{C}) \subseteq \mathcal{P}$: in other words, applying affine transforms from the automorphism group of the code to one chosen cog in the family produces cogs that generate parity-check matrices with identical stopping set distributions. Therefore, the MBBP decoder does not have to store all redundant bases, but rather a set of permutations, along with a low number of cog vectors that are known to have good stopping set properties, which is a desirable property for practical applications.

Let $\hat{f}$ denote the index of the optimal or near-optimal family. One vector $\mathrm{cog}_{\hat{l}}$, $\hat{l} \in \mathcal{F}_{\hat{f}}$, and a subset of permutations in $\mathrm{Aut}(\mathcal{C})$ suffice to generate a set of cogs from $\mathcal{F}_{\hat{f}}$. Depending on the set of available permutations, multiple cogs may be required to generate all cogs from the family $\mathcal{F}_{\hat{f}}$. Following, the matrices $\boldsymbol{H}_\ell$, $\ell \in \mathcal{F}_{\hat{f}}$ can be constructed by cyclically shifting the corresponding cogs, cf. Section II.

IV. MBBP DECODING

We describe next in more detail the operating principles of the MBBP decoders, and modifications thereof. The basic components of an MBBP decoder are collections of (possibly redundant) parity-check matrices, and logical units that combine and process outputs of BP decoders operating on the matrices of the collection. We distinguish two basic MBBP architectures: one, which allows information exchange between decoders (*MBBP-X*) and another, where the decoder outputs are obtained without exchange of information (*MBBP-NX*). The decoders in the former category have the feature that information on the reliability of the received symbols can be exchanged *during* the process of iterative BP decoding; decoders in the latter class can only combine their results *upon termination* of their individual decoding processes. Both types of decoders can be implemented by storing a set of parity-check matrices. However, if only matrices based on one family of cogs are used, one needs to store only a low number of cogs, along with a set of permutations from $\mathcal{P}$.

The simplest architecture of an MBBP decoder is depicted in Fig. 1, where the outputs of individual decoders are jointly processed only at the end of the decoding cycle. We refer to this technique as *standard* MBBP decoding.

A. MBBP-NX decoding

Standard MBBP decoding (MBBP-NX-S) and its variation *First-success MBBP decoding* (MBBP-NX-FS) generate a collection of decoded words and then perform an additional metric selection within this set of words. The result of this processing is passed on to the *information sink*, which represents the gateway for the final codeword estimate of the decoder.

The MBBP-NX-S decoder runs multiple BP decoders in parallel, each of them separately and on a different parity-check matrix representation of the code. Let the parity-check matrix representation used by the ℓ-th decoder be denoted by $\boldsymbol{H}_\ell$, $\ell = 1, \ldots, l$, and its decoded vector after at most N iterations by $\hat{\boldsymbol{c}}_\ell$, $\ell = 1, \ldots, l$. We let $\mathcal{V} \subseteq \{1, \ldots, l\}$ be the set of indices ℓ describing decoders that converged to a valid codeword. If none of the decoders converged to a valid codeword, we let $\mathcal{V} = \{1, \ldots, l\}$. The words estimated by the decoders, $\hat{\boldsymbol{c}}_v$, $v \in \mathcal{V}$ are passed on to a *least metric selector* (LMS) unit, which determines the “best” codeword estimate using the decision rule $\hat{\boldsymbol{c}} = \operatorname{argmax}_{v \in \mathcal{V}} \Pr\left\{\boldsymbol{Y} = \boldsymbol{y} \mid \boldsymbol{C} = \hat{\boldsymbol{c}}_v\right\}$. The estimated information vector $\hat{\boldsymbol{u}}$ is obtained from $\hat{\boldsymbol{c}}$ in the standard manner. Fig. 1 depicts the operation of the MBBP-NX-S decoder.

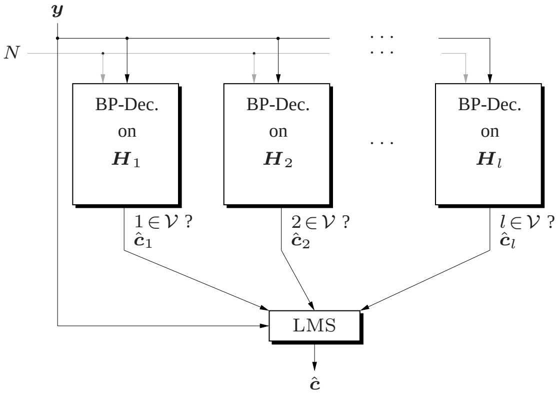

$\boldsymbol{y}$
N
. . .
. . .
BP-Dec. on $\boldsymbol{H}_1$
BP-Dec. on $\boldsymbol{H}_2$
. . .
BP-Dec. on $\boldsymbol{H}_l$
$1 \in \mathcal{V}$? $\hat{\boldsymbol{c}}_1$
$2 \in \mathcal{V}$? $\hat{\boldsymbol{c}}_2$
$l \in \mathcal{V}$? $\hat{\boldsymbol{c}}_l$
LMS
$\hat{\boldsymbol{c}}$

Fig. 1. MBBP-NX-S decoding.

The MBBP-NX-FS decoder follows the standard approach in so far that it runs multiple BP decoders separately, each on a different parity-check matrix of the code. Denote the number of iterations required by the ℓ-th decoder to converge by N_ℓ. As soon as the first decoder, indexed by ℓ^*, identifies a codeword, the decoding process terminates. The estimate obtained by the decoder indexed by ℓ^* is passed on to the information sink. If two or more decoders converge to a codeword after the same number of iterations, one of the outputs is randomly chosen from $\boldsymbol{c}_v$, where $v \in \mathcal{V}$ and where $\mathcal{V}$ is defined as for the MBBP-NX-S decoder. This approach requires only $N_{\ell^*} = \min_\ell N_\ell$ iterations to converge, and has therefore a lower time-complexity when compared to MBBP-NX-S. For the average number of iterations to decode one codeword, this effect shows in particular in the low signal-to-noise ratio (SNR) regime, cf. [1].

B. *MBBP-X decoding*

In this section we present MBBP approaches which allow for *periodic exchange* of information between decoders *during* iterative BP decoding. To this end, each decoder performs independently a given number of iterations, N_p, and afterwards exchanges information with other decoders only at iterations indexed by $\iota \cdot N_\mathrm{p}$, where $\iota \in \mathbb{N}$. The *intrinsic information* of a given variable node depends only on the channel output and is therefore equal for all decoders. For this reason, the decoders exchange only *extrinsic information* about the variable nodes.

To emphasize that the messages exchanged between decoders represent extrinsic information, they are denoted by $\Pr^{(\mathrm{e})}(\cdot)$. As part of their *cooperation scheme*, the decoders agree on the (extrinsic) probability values for each variable node. Afterwards, each decoder replaces its own information about a given variable node with the jointly derived estimate of the decoders, then calculates the *a-posteriori information*, and resumes decoding. The jointly derived estimate can be found by probabilistic averaging (probability-averaging MBBP, MBBP-X-PA), by selecting the most reliable output from all decoders (highest-reliability MBBP-X, MBBP-X-HR), and by information combining [14] (information-combining MBBP-X, MBBP-X-IC). The latter approach is the optimal decision method on the value of a random variable when multiple independent noisy observations are given. We point out that the combined observations, which originate from different BP decoders, are not independent. Nevertheless, we propose this type of algorithm.

For the purpose of computing the cooperative extrinsic probability, only a subset of active decoders $\mathcal{A}_\nu$, $\nu = 0, \dots, n-1$, is used. This subset is selected in terms of a *soft-metric* majority vote $\bar{v}_\nu$ which is calculated for a variable node ν according to

$$\bar{v}_\nu = \sum_{\ell=1}^{l} \log\left(\frac{\Pr^{(\mathrm{e})}(c_{\ell,\nu} = 0 \mid \boldsymbol{H}_\ell, \boldsymbol{y})}{\Pr^{(\mathrm{e})}(c_{\ell,\nu} = 1 \mid \boldsymbol{H}_\ell, \boldsymbol{y})}\right),\ \nu = 0, \dots, n-1.$$

The subset of decoders used in the described processing step is different for each variable node and defined as

$$\mathcal{A}_\nu = \left\{\ell \middle| \operatorname{sgn}(\bar{v}_\nu) = \operatorname{sgn}\left(\log\left(\frac{\Pr^{(\mathrm{e})}(c_{\ell,\nu} = 0 \mid \boldsymbol{H}_\ell, \boldsymbol{y})}{\Pr^{(\mathrm{e})}(c_{\ell,\nu} = 1 \mid \boldsymbol{H}_\ell, \boldsymbol{y})}\right)\right)\right\}, \tag{3}$$

where $\nu = 0, \dots, n-1$ holds, and $\operatorname{sgn}(\cdot)$ denotes the sign function.

If averaging is used, the updated probabilities are calculated by

$$\Pr^{(\mathrm{e})}(c_\nu = 0) = \frac{1}{|\mathcal{A}_\nu|} \sum_{\ell' \in \mathcal{A}_\nu} \Pr^{(\mathrm{e})}(c_{\ell',\nu} = 0 \mid \boldsymbol{H}_{\ell'}, \boldsymbol{y}),$$

where $\nu = 0, \dots, n-1$.

In the case of MBBP-X-HR, we calculate $\Pr^{(\mathrm{e})}(c_\nu = 0) = \Pr^{(\mathrm{e})}(c_{\ell^*,\nu} = 0 \mid \boldsymbol{H}_{\ell^*}, \boldsymbol{y})$, with

$$\ell^* = \operatorname{argmax}_{\ell' \in \mathcal{A}_\nu} \left|\Pr^{(\mathrm{e})}(c_{\ell',\nu} = 0 \mid \boldsymbol{H}_{\ell'}, \boldsymbol{y}) - 0.5\right|,$$

$\nu = 0, \dots, n-1$.

In the case of information-combining MBBP-X, the processing rule follows Equation (4).

Independent of the MBBP-X approach, we calculate the updated counter-probabilities by $\Pr^{(\mathrm{e})}(c_\nu = 1) = 1 - \Pr^{(\mathrm{e})}(c_\nu = 0)$, $\nu = 0, \dots, n-1$. We point out that the performance of these approaches strongly depends on the specific implementation of the proposed steps. An algorithmic summary of all decoding algorithms can be found in [1].

V. Results

As stated before, parity-check matrices that offer good performance when used for decoding signals transmitted over the BEC may also be good candidates for BP decoding over the AWGN channel [25]. The same characteristic of parity-check matrices was observed through extensive computer simulations for the variants of MBBP decoders, used over the AWGN channel. These results are presented in this section. Additionally, we provide a comparison of the error rates

$$\Pr^{(\mathrm{e})}(c_\nu = 0) = \frac{\prod\limits_{\ell' \in \mathcal{A}_\nu} \Pr^{(\mathrm{e})}(c_{\ell',\nu} = 0 \mid \boldsymbol{H}_{\ell'}, \boldsymbol{y})}{\prod\limits_{\ell' \in \mathcal{A}_\nu} \Pr^{(\mathrm{e})}(c_{\ell',\nu} = 0 \mid \boldsymbol{H}_{\ell'}, \boldsymbol{y}) + \prod\limits_{\ell' \in \mathcal{A}_\nu} \Pr^{(\mathrm{e})}(c_{\ell',\nu} = 1 \mid \boldsymbol{H}_{\ell'}, \boldsymbol{y})}, \ \nu = 0, \ldots, n-1. \tag{4}$$

of MBBP decoders with those of a full search algorithm (maximum-likelihood, ML, decoding). Furthermore, we compare our results to the *Gallager bound* (random coding bound) [26] and standard BP decoding. Due to space limitations, we illustrate our findings only on the $[24, 12, 8]$ extended Golay code[2].

The maximum number of iterations is set to $N = 100$ for all codes and decoding approaches considered. Furthermore, $N_\mathrm{p} = 10$ is set whenever MBBP-X approaches are simulated. We use l to denote the number of parallel BP decoders in the MBBP architecture.

A. *The* $[24, 12, 8]$ *Extended Golay Code*

For the purpose of MBBP decoding of the extended Golay code, we use the result of Sections II and III, and identify three different cog families denoted by $\mathcal{F}_1$, $\mathcal{F}_2$, and $\mathcal{F}_3$. For this code, one can generate cogs of a family by repeated application of permutations of type P_2. Let us describe the process of constructing the parity-check matrices for the extended Golay code in more detail.

The $[24, 12, 8]$ extended Golay code is self dual and contains 759 codewords of minimum weight 8. This set of codewords can be partitioned into 33 cyclic orbits. Again, these cyclic orbits can be partitioned into three families of equal size. In other words, each family $\mathcal{F}_f$, $f = 1, 2, 3$, contains 11 cogs.

Since the code is an extended cyclic code, we construct the parity-check matrices from each cog in terms of 23 shifts performed on positions 0 to 22 while keeping the last position fixed. For the 24-th row of the parity-check matrix, we use the all-one codeword: this codeword preserves the stopping set distribution of the 23×24 matrix, and is the only reasonable parity check of the $[24, 12, 8]$ extended Golay code invariant under all affine permutations. As a performance criterion for the cog families, we use the number of stopping sets up to size $d = 8$ in the parity-check matrices $\boldsymbol{H}_\ell$, $\ell \in \{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3\}$. We consider one representative cog for each family, termed $\mathrm{cog}_{\mathcal{F}_f}$, $f = 1, 2, 3$. These cogs read

$$\begin{aligned} \mathrm{cog}_{\mathcal{F}_1} &= 110101001100100000001000, \\ \mathrm{cog}_{\mathcal{F}_2} &= 111000001001100000100001, \\ \mathrm{cog}_{\mathcal{F}_3} &= 110100110000000101001000. \end{aligned}$$

The number of stopping sets in $\boldsymbol{H}_\ell$, $\ell \in \{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3\}$, is summarized in Table I. Note that the matrices of the considered families differ significantly from the near-optimal and highly redundant matrices used for decoding over the BEC, given in [6] and [7]. Due to the high redundancy, these matrices are not amenable for decoding over the AWGN channel, where short cycles may significantly degrade the performance of iterative decoders.

If matrices from the same family are used for signaling over the BEC, they provide the same performance under iterative decoding. Interestingly, the simulation results presented below show that the same is true for decoders used over the AWGN channel. Fig. 2 shows the BER performance of MBBP-NX-S decoding and MBBP-NX-FS decoding as well as MBBP-X-PA, MBBP-X-HR, and MBBP-X-IC of the extended Golay code. For all decoder types and all three families, $l = 11$ parallel BP decoders are used. FER performance results show similar characteristics, but these results are not plotted due to space limitations.

It can be observed for all approaches that MBBP decoders using matrices $\boldsymbol{H}_\ell$, $\ell \in \mathcal{F}_1$, have the best performance, followed by matrices $\boldsymbol{H}_\ell$, $\ell \in \mathcal{F}_3$. This finding is supported by the stopping set distribution of Table I, showing that the first family contains matrices that do not have stopping sets of size up to six and a low number of stopping sets of size seven. Matrices from the other two families have stopping sets of size six and exceed the number of stopping sets of size seven of the first family. The performance of MBBP decoders using parity-check matrices $\boldsymbol{H}_\ell$, $\ell \in \mathcal{F}_2$, is significantly worse than that of the two other classes - matrices in this family have 437 stopping sets of size six and over $10,000$ stopping sets of size seven.

The performance obtained by means of the MBBP-NX-S with $l = 11$ is approximately $0.75\,\mathrm{dB}$ better than standard BP and performs close to the ML decoding bound. When performing a direct comparison of MBBP-NX-S and MBBP-NX-FS decoders, cf. Fig. 2, one can observe that MBBP-NX-FS follows the performance of MBBP-NX-S very closely. This is a remarkable result, as the MBBP-NX-FS decoder requires a significantly lower number of iterations on the average.

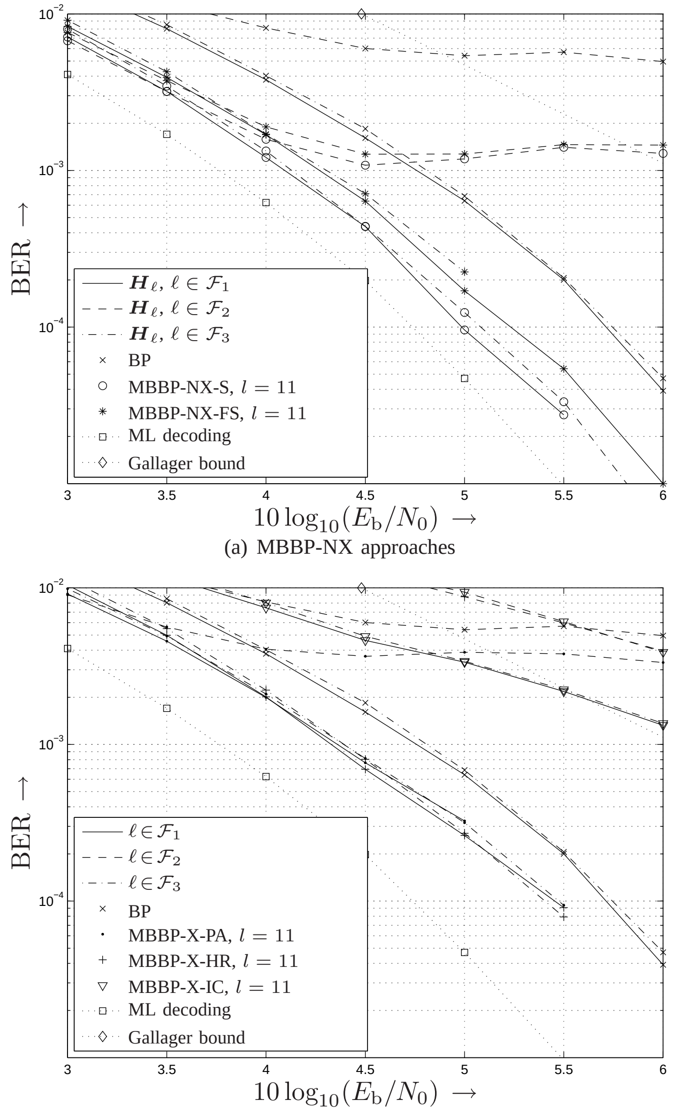
It is worth pointing out that the MBBP-X-PA and MBBP-X-HR algorithms achieve similar results when the cog family is fixed. Also, if the cogs are chosen from $\mathcal{F}_1$ or $\mathcal{F}_3$, the MBBP-X approaches outperform standard BP decoding, but do not attain the performance of MBBP-NX decoders. If the decoders operate on parity-check matrices constructed from cogs in $\mathcal{F}_2$, very poor performance results and error floors are observed in most of the cases. MBBP-X-IC decoders perform very poorly, regardless of the family considered. A probable cause for this phenomena is the strong correlation of the data, which makes information combining techniques highly suboptimal.

Let us now assess the results in a more general way. In Fig. 2 we observe that the results for both standard BP and MBBP outperform the Gallager bound. In further simulations we observed for codes of longer length that BP decoding does not outperform this bound while MBBP decoding does. This is exemplary shown in [1] using a $[47, 24, 11]$ QR code. For codes of significantly longer lengths, performance

[2] Note that results for the $[31, 16, 7]$ BCH code, the $[47, 24, 11]$ QR code, and the $[127, 64, 21]$ BCH code are available in [1].

	$\lvert\mathcal{S}_\sigma(\boldsymbol{H}_\ell)\rvert$, $\ell \in \mathcal{F}_1$	$\lvert\mathcal{S}_\sigma(\boldsymbol{H}_\ell)\rvert$, $\ell \in \mathcal{F}_2$	$\lvert\mathcal{S}_\sigma(\boldsymbol{H}_\ell)\rvert$, $\ell \in \mathcal{F}_3$
$\sigma \leq 5$	0	0	0
$\sigma = 6$	0	437	46
$\sigma = 7$	1357	10143	1495
$\sigma = 8$	25783	73209	20631

TABLE I
Number of stopping sets for the $[24, 12, 8]$ extended Golay code, $\boldsymbol{H}_\ell$, $\mathrm{cog}_\ell \in \mathcal{F}_f$, $f = 1, \ldots, 3$.



(a) MBBP-NX approaches

(b) MBBP-X approaches

Fig. 2. Performance comparison for the $[24, 12, 8]$ extended Golay code using $\boldsymbol{H}_\ell$, $\ell \in \mathcal{F}_1$, $\mathcal{F}_2$, $\mathcal{F}_3$.

improvements exist when using MBBP decoding instead of BP decoding, but do not suffice to underrun the Gallager bound.

Similar observations were made for Chase Type-2 decoding [21]. While MBBP-NX-S outperforms Chase Type-2 decoders for codes of short length [27], we show in [1] that Chase Type-2 is superior to MBBP decoding for the $[127, 64, 21]$ BCH code. This is strongly related to the number of test patterns in the Chase decoding setup, which grows exponentially with the minimum distance. In the discussed example, it reads $2^{\lfloor d/2 \rfloor} =$ 1024 and is hence significantly higher than the diversity order in any reasonable MBBP-NX-S setup.

It is also of interest to relate the presented results to the performance obtained by a BP decoder operating on the union of the parity checks from the matrices used in the MBBP setup. We have shown in [27] that the performance of such an approach is even worse than the performance of a standard BP algorithm which operates on any of these matrices.

VI. Conclusions

We introduced a class of decoding algorithms that operate in parallel on a judiciously chosen family of parity-check matrices. We considered two variants of this class of techniques: one, in which the decoders are not allowed to exchange information during individual runs of the BP algorithm, and another, in which periodic information exchange is allowed. Algorithms in the first class were shown to offer significant performance improvements when compared to the standard BP technique. The approaches in the second class often compare favorably to standard BP, but do not match the performance of algorithms that do not make use of periodic information exchange. Possible reasons for this behavior include the fact that “cycles” are created during the process of information exchange. These cycles “in-between” the graphs of the representations negatively affect the performance of each decoder.

The presented approaches were shown to work for classical high-density codes and are applicable to cyclic and extended cyclic codes. It is possible to generalize the introduced methods to codes like the progressive edge-growth (PEG) [28] family. For this class of codes, significant gains in performance can be obtained. Results in this direction were presented in [29].

References

[1] T. Hehn, J. Huber, O. Milenkovic, and S. Laendner. (2009, May) Multiple-bases belief-propagation decoding of high-density cyclic codes. Website. [Online]. Available: http://arxiv.org/abs/0905.0079

[2] M. Fossorier and S. Lin, “Soft-decision decoding of linear block codes based on ordered statistics,” *IEEE Transactions on Information Theory*, vol. 41, pp. 1379–1396, September 1995.

[3] R. Lucas, M. Bossert, and M. Breitbach, “On iterative soft-decision decoding of linear binary block codes and product codes,” *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 276–296, February 1998.

[4] T. Halford and K. Chugg, “Random redundant soft-in soft-out decoding of linear block codes,” in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, Seattle, Washington, USA, July 2006, pp. 2230–2234.

[5] T. Halford, A. Grant, and K. Chugg, “Which codes have 4-cycle-free Tanner graphs?” *IEEE Transactions on Information Theory*, vol. 52, no. 9, pp. 4219–4223, September 2006.

[6] M. Schwartz and A. Vardy, “On the stopping distance and stopping redundancy of codes,” *IEEE Transactions on Information Theory*, vol. 52, no. 3, pp. 922–932, March 2006.

[7] J. Han and P. Siegel, “Improved upper bounds on stopping redundancy,” *IEEE Transactions on Information Theory*, vol. 53, no. 1, pp. 90–104, January 2007.

[8] H. Hollmann and L. Tolhuizen, “On parity-check collections for iterative erasure decoding that correct all correctable erasure patterns of a given size,” *IEEE Transactions on Information Theory*, vol. 53, no. 2, pp. 823–828, February 2007.

[9] 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.

[10] J. Jiang and K. Narayanan, “Iterative soft-input soft-output decoding of Reed-Solomon codes by adapting the parity-check matrix,” *IEEE Transactions on Information Theory*, vol. 52, no. 8, pp. 3746–3756, August 2006.

[11] K. Andrews, S. Dolinar, and F. Pollara, “LDPC decoding using multiple representations,” in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, Lausanne, Switzerland, June 2002, p. 456.

[12] T. Hehn, O. Milenkovic, S. Laendner, and J. Huber, “Permutation decoding and the stopping redundancy hierarchy of cyclic and extended cyclic codes,” *IEEE Transactions on Information Theory*, vol. 54, no. 12, pp. 5308–5331, December 2008.

[13] S. Laendner and O. Milenkovic, “Algorithmic and combinatorial analysis of trapping sets in structured LDPC codes,” in *Proceedings of the International Conference on Wireless Networks, Communications, and Mobile Computing (WirelessComm)*, Maui, Hawaii, June 2005, pp. 630–635.

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

[15] L. Zeng, L. Lan, Y. Y. Tai, S. Lin, and K. Abdel-Ghaffar, “Construction of LDPC codes for AWGN and binary erasure channels based on finite fields,” in *Proceedings of IEEE Information Theory Workshop (ITW)*, Rotorua, New Zealand, August 2005, pp. 273–276.

[16] C. Di, D. Proietti, I. Telatar, T. Richardson, and R. Urbanke, “Finite-length analysis of low-density parity-check codes on the binary erasure channel,” *IEEE Transactions on Information Theory*, vol. 48, no. 6, pp. 1570–1579, June 2002.

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

[18] C. Kelley and D. Sridhara, “Pseudocodewords of Tanner graphs,” *IEEE Transactions on Information Theory*, vol. 53, no. 11, pp. 4013–4038, November 2007.

[19] R. Koetter and P. Vontobel, “Graph covers and iterative decoding of finite-length codes,” in *Proceedings of the 3rd International Symposium on Turbo Codes & Related Topics*, Brest, France, September 2003, pp. 75–82.

[20] J. Kovacevic and A. Chebira, “Life beyond bases: The advent of frames (Part I),” *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 86 – 104, July 2007.

[21] S. Lin and D. Costello, *Error Control Coding*, 2nd ed. Pearson Education, Inc., 2004.

[22] S. Laendner, T. Hehn, O. Milenkovic, and J. Huber, “When does one redundant parity-check equation matter?” in *Proceedings of the 49th annual IEEE Global Telecommunications Conference (GlobeCom)*, San Francisco, USA, November 2006.

[23] A. McGregor and O. Milenkovic, “On the hardness of approximating stopping and trapping sets in LDPC codes,” in *Proceedings of the IEEE Information Theory Workshop (ITW)*, Lake Tahoe, California, USA, September 2007, pp. 248–253.

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

[25] T. Richardson, M. Shokrollahi, and R. Urbanke, “Design of capacity-approaching irregular low-density parity-check codes,” *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 619–637, February 2001.

[26] R. G. Gallager, *Information Theory and Reliable Communication*. John Wiley and Sons, 1968.

[27] T. Hehn, J. Huber, S. Laendner, and O. Milenkovic, “Multiple-bases belief-propagation decoding for short block-codes,” in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, Nice, France, June 2007, pp. 311–315.

[28] 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.

[29] T. Hehn, J. Huber, P. He, and S. Laendner, “Multiple-bases belief-propagation with leaking for decoding of moderate-length block codes,” in *Proceedings of the International ITG Conference on Source and Channel Coding (SCC)*, Ulm, Germany, January 2008.