Iterative decoding threshold analysis for LDPC convolutional codes

Lentmaier, Michael; Sridharan, Arvind; Costello Jr., Daniel J.; Zigangirov, Kamil

Published in:
IEEE Transactions on Information Theory

DOI:
10.1109/TIT.2010.2059490

2010

Iterative Decoding Threshold Analysis for LDPC Convolutional Codes

Michael Lentmaier, *Member, IEEE,* Arvind Sridharan, *Member, IEEE,*
Daniel J. Costello, Jr., *Life Fellow, IEEE,* and Kamil Sh. Zigangirov, *Fellow, IEEE*

***Abstract*—An iterative decoding threshold analysis for terminated regular LDPC convolutional (LDPCC) codes is presented. Using density evolution techniques, the convergence behavior of an iterative belief propagation decoder is analyzed for the binary erasure channel and the AWGN channel with binary inputs. It is shown that for a terminated LDPCC code ensemble, the thresholds are better than for corresponding regular and irregular LDPC block codes.**

***Index Terms*—Low-density parity-check (LDPC) codes, LDPC convolutional codes, iterative decoding, message passing, belief propagation, threshold analysis, density evolution**

I. Introduction

In classical coding theory, the efficiency of block and convolutional coding schemes is characterized by two criteria: decoding complexity and decoding reliability. The decoding complexity is usually measured by the number of operations per symbol or per frame. The decoding reliability of block codes is typically measured by the bit error probability or the block error probability. In convolutional coding, when the length of the sequences becomes much larger than the constraint length, the burst or first event error probability is considered instead of the block error probability. Since an analysis of particular codes is often difficult, the asymptotic performance of code ensembles, when block or constraint length tend to infinity, is frequently considered.

For maximum likelihood (ML) decoding, the decoding complexity is an exponential function of block or constraint length. At the same time, for most codes the bit or block/burst error probabilities decrease exponentially with block or constraint length if the code rate is less than the channel capacity. As a consequence, both decoding complexity and reliability are characterized by exponential functions of block or constraint length. By definition, we call convergence to zero of the block/burst error probability, when block or constraint length tends to infinity, the *reliable communication* condition.

The reliability of codes of a fixed rate is characterized by the *Shannon limit*, the boundary value of the channel parameter for which the channel capacity exceeds the code rate. For the additive white Gaussian noise (AWGN) channel, it is the smallest signal-to-noise ratio (SNR), for the binary symmetric channel (BSC) or the binary erasure channel (BEC), the largest crossover or erasure probability, respectively, for which reliable communication is possible.

Modern coding theory deals mostly with low-density parity-check (LDPC) codes and iterative decoding methods. The decoding complexity of these codes is usually characterized by the number of operations per bit and per iteration. In most cases this number is a constant, i.e., it does not depend on the block or constraint length. At the same time, for LDPC codes with symbol node degrees larger than two, the bit error probability decreases at least doubly exponentially with the number of iterations [3] as the block or constraint length goes to infinity, given that the SNR is larger than some constant (for the AWGN channel) or the crossover/erasure probability is smaller than some constant (for the BSC/BEC). This makes iterative decoding of codes with large block or constraint lengths feasible.

We call the boundary value of the channel parameter for which reliable communication is possible with iterative decoding the *iterative decoding limit*. In some sense, the iterative decoding limit is the counterpart of the Shannon limit, but it is associated with particular ensembles of codes and particular iterative decoding methods.

While an exact calculation of the iterative decoding limit is difficult, it is possible to find bounds. The Shannon limit is an obvious lower bound on the iterative limit for the AWGN channel and an upper bound for the BSC or BEC. Bounds in the other direction are commonly called *convergence thresholds*. These thresholds are often assumed to coincide with the actual iterative decoding limit, although this is difficult to prove.

In this paper, we study the iterative decoding thresholds of terminated (J, K) *regular* LDPC convolutional codes (LDPCC codes), with $J > 2$. LDPCC codes, the convolutional counterpart of Gallager's LDPC block codes [4], have been described[1] in [6]–[9]. Both LDPC block and LDPC convolu-

This work was supported in part by NSF Grants CCR-02-05310 and CCF-0514801 and NASA Grants NNG05GH73G and NNX07AK53G. Some of the material in this paper previously appeared in the Proceedings of the 2004 Allerton Conference [1] and the 2005 IEEE International Symposium on Information Theory [2].

M. Lentmaier was with the Department of Electrical Engineering, University of Notre Dame. He is now with the Vodafone Chair Mobile Communications Systems, Dresden University of Technology (TU Dresden), D-01062 Dresden, Germany (e-mail: Michael.Lentmaier@ifn.et.tu-dresden.de).

A. Sridharan was with the Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN 46556. He is now with Seagate Technologies, 389 Disc Drive, Longmont, CO 80503, USA (e-mail: arvind.sridharan@seagate.com).

K. Sh. Zigangirov is with the University of Notre Dame, Notre Dame, IN 46556 USA, Lund University, Lund, Sweden, and the Institute for Problems of Information Transmission, Moscow, Russia (e-mail: Kamil.Zigangirov@eit.lth.se).

D. J. Costello, Jr. is with the Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN 46556, USA. (e-mail: Daniel.J.Costello.2@nd.edu)

[1]Some suggestions for the construction and decoding of LDPCC codes were earlier given in a patent application by Tanner [5].

tional codes are defined by sparse parity-check matrices and can be decoded iteratively using *message passing* algorithms with complexity per bit per iteration independent of the block or constraint length. An example of a message passing decoding algorithm for LDPC codes is belief propagation, which will be considered throughout this paper. The ensemble of LDPCC codes we analyze is defined by parity-check matrices composed of permutation matrices and is described in Section II. This ensemble was previously used to lower bound the free distance of LDPCC codes [10]. The block counterparts of these codes were introduced in the appendix of Gallager's book [4] and also studied in [11] and [12]. In particular, [12] proved the existence of codes in this ensemble with a minimum distance satisfying Gallager's lower bound [4].

The calculation of convergence thresholds for LDPC codes is based on estimating the bit error probability when the block or constraint length and the number of iterations go to infinity. The bit error probability, in turn, can be found by calculating the probability density function of the decision statistics used in the message passing algorithm in different iterations. A commonly used recursive numerical algorithm for the calculation of this density function is known as *density evolution* [13]. Since numerical methods alone are not sufficient to prove convergence of the error probability to zero, density evolution can be combined with analytical methods, e.g., bounding the evolution of the Bhattacharyya parameter [3], [14]–[18]. Several papers have been devoted to analyzing convergence thresholds for regular and *irregular* LDPC block codes (e.g., [19] and [20]), with the result that thresholds of irregular codes are generally better than those of regular codes.

The application of density evolution requires that all messages exchanged during the iterations are independent, i.e., that the decoder operates in cycle-free regions of the graph. In Section III it is shown that the standard parallel updating schedule can be used for the threshold analysis of terminated LDPCC codes, provided that the constraint length can be chosen sufficiently large. In the BEC case, density evolution reduces to observing a single parameter, the erasure probability, and can be described by simple recursive equations. Unfortunately, for the AWGN channel, performing density evolution for the standard parallel updating schedule becomes increasingly difficult in the case of large frame lengths.

In Section IV, a sliding window updating schedule for threshold analysis is described. This approach can be used to calculate thresholds even when the termination length of the convolutional codes is large. Furthermore, we conjecture that this approach can also be applied to analyze the thresholds of unterminated LDPCC codes.

A numerical analysis of the calculated thresholds, with results for the BEC and the AWGN channel, is given in Section V, and we see that the thresholds of some regular LDPCC codes closely approach the ML decoding thresholds of the corresponding regular LDPC block codes [21], [22], which in turn approach the Shannon limit exponentially with increasing symbol node degrees. Further, it is surprising to note that increasing the density of the parity-check matrices leads to improved belief propagation thresholds, unlike the behavior of regular LDPC block codes. Simulation results for the BEC confirm the calculated thresholds.

II. LDPC Convolutional Code Ensemble

A rate $R = b/c$ time-varying binary convolutional code can be defined as the set of sequences

$$\boldsymbol{v} = (\ldots, \boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_t, \ldots) \ , \quad \boldsymbol{v}_t \in \mathbb{F}_2^c \ , \tag{1}$$

satisfying the equality $\boldsymbol{v}\boldsymbol{H}^{\mathrm{T}} = \boldsymbol{0}$, where the infinite syndrome former matrix $\boldsymbol{H}^{\mathrm{T}}$ is given by

$$\boldsymbol{H}^{\mathrm{T}} = \begin{bmatrix} \ddots & & \ddots & & \\ \boldsymbol{H}_0^{\mathrm{T}}(1) & \ldots & \boldsymbol{H}_{m_{\mathrm{s}}}^{\mathrm{T}}(1+m_{\mathrm{s}}) & & \\ & \ddots & & \ddots & \\ & & \boldsymbol{H}_0^{\mathrm{T}}(t) & \ldots & \boldsymbol{H}_{m_{\mathrm{s}}}^{\mathrm{T}}(t+m_{\mathrm{s}}) \\ & & \ddots & & \ddots \end{bmatrix} , \tag{2}$$

and each $\boldsymbol{H}_i^{\mathrm{T}}(t+i)$ is a $c \times (c-b)$ binary matrix. If $\boldsymbol{H}^{\mathrm{T}}$ defines a rate $R = b/c$ convolutional code, the matrix $\boldsymbol{H}_0^{\mathrm{T}}(t)$ must have full rank for all time instants t. In this case, by suitable row permutations, we can ensure that the last $(c-b)$ rows are linearly independent. Then the first b symbols at each time instant are information symbols and the last $(c-b)$ symbols are the corresponding parity symbols. The largest i such that $\boldsymbol{H}_i^{\mathrm{T}}(t+i)$ is a nonzero matrix for some t is called the *syndrome former memory* m_s [23]. LDPCC codes have sparse syndrome former matrices. A (J, K) regular LDPCC code is defined by a syndrome former that contains exactly J ones in each row and K ones in each column.

We now describe the ensemble of LDPCC codes introduced in [10]. Although the ensemble can be defined more generally, in this paper we focus on the case $K = 2J$, $J > 2$. We construct LDPCC codes defined by syndrome formers $\boldsymbol{H}^{\mathrm{T}}$ with syndrome former memory $m_{\mathrm{s}} = J-1$. For $i = 0, 1, \ldots, J-1$, the sub-matrices of the syndrome former are equal to

$$\boldsymbol{H}_i^{\mathrm{T}}(t+i) = \begin{bmatrix} \boldsymbol{P}_i^{(0)}(t+i) & \boldsymbol{P}_i^{(1)}(t+i) \end{bmatrix}^{\mathrm{T}} \ , \tag{3}$$

where the matrices $\boldsymbol{P}_i^{(h)}(t+i),\ h = 0, 1$, are $M \times M$ permutation matrices. All other entries of the syndrome former are zero matrices. Equivalently, each $\boldsymbol{H}_i^{\mathrm{T}}(t+i)$, $i = 0, 1, \ldots, J-1$, is a $c \times (c-b)$ binary matrix. In the case $K = 2J$, we have $c = 2M$ and $b = M$. By construction it follows that each row of the syndrome former $\boldsymbol{H}^{\mathrm{T}}$ has J ones and each column has K ones. Suppose the permutation matrices $\boldsymbol{P}_i^{(h)}(t+i),\ t = 1, 2, \ldots,\ i = 0, 1, \ldots, J-1,\ h = 0, 1$, are chosen independently and all $M!$ values are equally likely. Then the corresponding ensemble of syndrome formers $\boldsymbol{H}^{\mathrm{T}}$ defines an ensemble $\mathcal{C}_P(J, 2J, M)$ of $(J, 2J)$ regular time-varying LDPC convolutional codes. Figure 1 shows the syndrome former matrix of a (3,6) regular LDPC convolutional code in $\mathcal{C}_P(3, 6, M)$.

Since $\boldsymbol{H}_0^{\mathrm{T}}(t)$ consists of two non-overlapping permutation matrices, it has full rank. Hence $\boldsymbol{H}^{\mathrm{T}}$ defines a rate $R = \frac{M}{2M}$ code. Further, the constraint imposed by the syndrome former, i.e.,

$$\boldsymbol{v}_t\boldsymbol{H}_0^{\mathrm{T}}(t) + \boldsymbol{v}_{t-1}\boldsymbol{H}_1^{\mathrm{T}}(t) + \cdots + \boldsymbol{v}_{t-m_s}\boldsymbol{H}_{m_s}^{\mathrm{T}}(t) = \boldsymbol{0} \ , \tag{4}$$

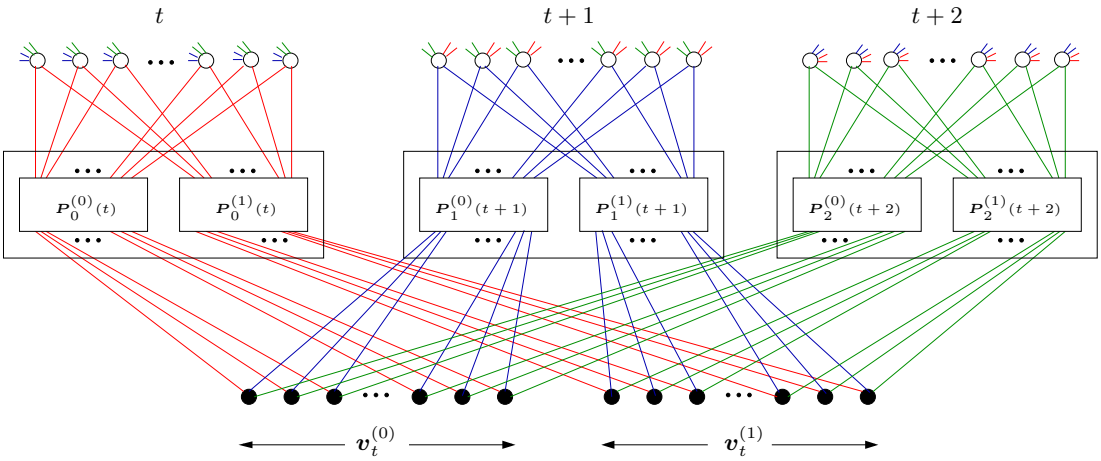

Fig. 2. Tanner graph connections of the $2M$ symbol nodes at time t for $J = 3$. Symbol nodes and constraint nodes are shown as black and white circles, respectively.

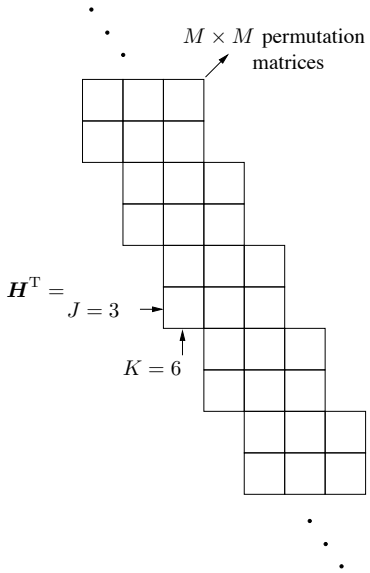

Fig. 1. Syndrome former matrix for a (3,6) regular LDPCC code in $\mathcal{C}_P(3, 6, M)$.

where $t \in \mathbb{Z}$, $\boldsymbol{v}_t = (\boldsymbol{v}_t^{(0)}, \boldsymbol{v}_t^{(1)})$, $\boldsymbol{v}_t^{(0)} = (v_{t,1}^{(0)}, v_{t,2}^{(0)}, ..., v_{t,M}^{(0)})$, $\boldsymbol{v}_t^{(1)} = (v_{t,1}^{(1)}, v_{t,2}^{(1)}, ..., v_{t,M}^{(1)})$, can be used to implement systematic encoding [6], [9]. The overall constraint length of the codes in $\mathcal{C}_P(J, 2J, M)$ is defined as $\nu = (m_s + 1) \cdot c = J \cdot 2M = KM$. Thus, the overall constraint length of codes in the ensemble $\mathcal{C}_P(3, 6, M)$ is $6M$.

The syndrome formers of codes in the ensemble $\mathcal{C}_P(J, 2J, M)$ have memory $m_{\mathrm{s}} = J - 1$ independent of M, while b and c depend on M. This is different from the LDPC convolutional codes considered in [6]–[8], where the codes have varying syndrome former memories m_{s}, while b and c are fixed. However, by a permutation of rows, the syndrome former of each code in the ensemble can be converted into the syndrome former of a conventional rate $R = 1/2$ LDPCC code with $b = 1$, $c = 2$, and $m_{\mathrm{s}} \leq 3M - 1$ [10]. This code differs from the original one only in the order of the symbols, i.e., it is an *equivalent* code.

The Tanner graph for a code in $\mathcal{C}_P(J, 2J, M)$ can be obtained from its syndrome former matrix. The graph consists of symbol nodes and constraint nodes, each symbol node corresponding to a particular row and each constraint node to a particular column of the syndrome former matrix. There is an edge between a symbol node and a constraint node if the corresponding symbol takes part in the respective parity-check equation. For the Tanner graph of a convolutional code we can associate a notion of time. At each time instant t, the sub-matrices $\boldsymbol{H}_i^{\mathrm{T}}(t)$ of the syndrome former $\boldsymbol{H}^{\mathrm{T}}$ (see (2)) lead to $c - b = M$ constraint nodes in the Tanner graph. Similarly, for each time instant t, there are $c = 2M$ symbol nodes in the Tanner graph. Observe that $\boldsymbol{H}_i^{\mathrm{T}}(t)$ is non-zero only for $i = 0, 1, \ldots, m_s$, and hence nodes in the Tanner graph can be connected at most m_s time units away. For $J = 3$, Fig. 2 illustrates how the information symbols $\boldsymbol{v}_t^{(0)} = (v_{t,1}^{(0)}, v_{t,2}^{(0)}, ..., v_{t,M}^{(0)})$ and the parity symbols $\boldsymbol{v}_t^{(1)} = (v_{t,1}^{(1)}, v_{t,2}^{(1)}, ..., v_{t,M}^{(1)})$ at time t are connected through different permutation matrices to parity-check equations at time t, $t + 1$, and $t + 2$.

For practical applications, a convolutional encoder starts to operate from a known state (usually the all-zero state) and after the block of data symbols to be transmitted has been encoded, the encoder is terminated back to the all-zero state using tail bits. Trellis based decoding algorithms for convolutional codes can exploit this known starting and ending state of the encoder to simplify decoding. As we shall see, a message passing decoder can also exploit the fact that the encoder starts and ends in the all-zero state. For LDPCC codes in the ensemble $\mathcal{C}_P(J, 2J, M)$, the syndrome former matrix can be used to determine the tail bits needed for termination. A termination procedure is presented in the appendix, together with a sketch of the proof of the following theorem (a complete proof can be found in [24]):

Theorem 1: Almost all codes in $\mathcal{C}_P(J, 2J, M)$ can be terminated with a tail of length $\tau = m_s+1$ blocks, i.e., $2M(m_s+1)$ bits. □

Termination of LDPCC codes is not only a practical method of transmitting finite blocks of data symbols but also a convenient instrument for theoretical performance assessment.

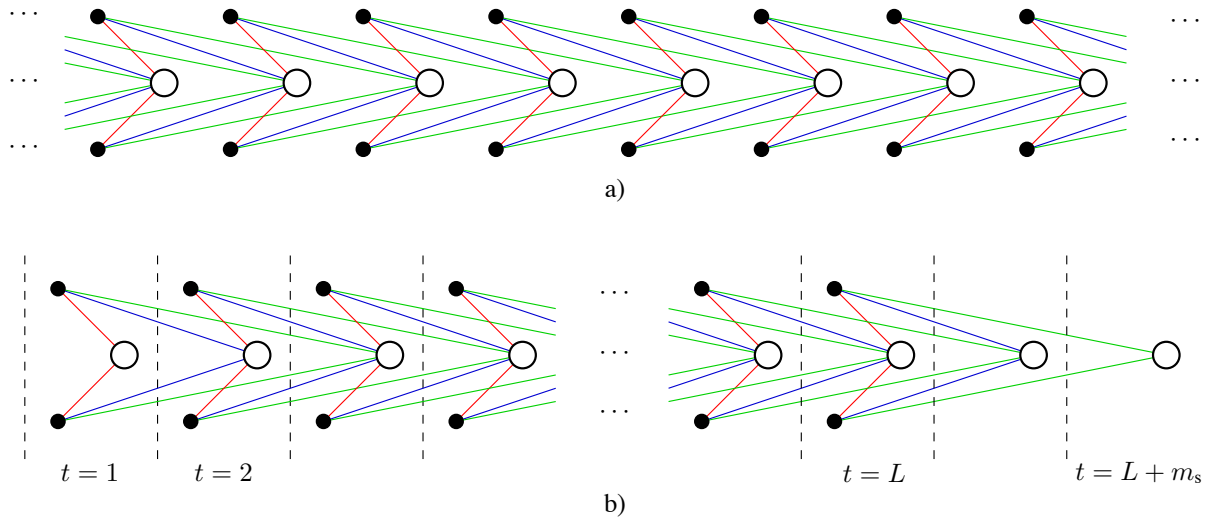

Fig. 3. Tanner graph for the case $M=1$ of a) the unique LDPCC code $\mathcal{C}_P(3,6,1)$ and b) the corresponding terminated code.

By terminating LDPCC codes we reduce the investigation of their thresholds to the estimation of thresholds of LDPC block codes and we can apply analysis methods developed for the latter.

In this paper we consider transmitting blocks of $N=2LM$ code bits using a code from $\mathcal{C}_P(J,2J,M)$. We assume that the encoder starts at time $t=1$ in the zero state, encodes M information bits into $2M$ code symbols at times $t=1,\ldots,L-\tau$, and then brings the encoder back to the zero state by adding a corresponding tail at times $t=L-\tau+1,\ldots,L$. It follows that the resulting terminated code has rate

$$R=\frac{(L-\tau)}{2L}\geq\frac{L-J}{2L}=\frac{1}{2}\left(1-\frac{J}{L}\right)\ , \qquad (5)$$

where we assume, motivated by Theorem 1, that the chosen codes satisfy $\tau\leq m_s+1=J$. Note that the rate loss is negligible for $L>>J$.

Fig. 3 shows the Tanner graph of an LDPCC code in the ensemble $\mathcal{C}_P(3,6,M)$ and of the corresponding terminated code. The depicted graphs correspond to the simplest case, $M=1$. They can be interpreted as *protographs* [25] for the ensembles with larger M, whose Tanner graphs can be obtained from the graphs in Fig. 3 by a copy-and-permute procedure. In the terminated graph, two symbol nodes (black) are followed by one constraint node (white) at each time instant t, $t=1,\ldots,L$. As for regular LDPC block codes, all symbol nodes and constraint nodes in the infinite Tanner graph of the original convolutional code have fixed degree J and $K=2J$, respectively (compare Fig. 2). In the case of termination, however, all symbol nodes at times $t<1$ and $t>L$ are known to be zero. But known symbols do not affect their associated parity-check equations and, consequently, their nodes can be removed from the Tanner graph along with all connected edges. As a result, while all symbol nodes have fixed degree $J=3$, the constraint nodes at times $t=1,\ldots,J-1$ and $t=L+1,\ldots,L+J-1$ will have a reduced degree. In particular, in Fig. 3, the constraint nodes at $t=1$ and $t=L+2$ have degree two and the constraint nodes at $t=2$ and $t=L+1$ have degree four. At all other time instants the constraint nodes have their full degree of $K=6$. It follows that, even though the convolutional code is regular, knowing the bits at time instants $t<1$ and $t>L$ leads to a slight irregularity in the Tanner graph of the convolutional code. As a consequence, the constraint nodes of lower degree provide better protection for the symbols at the beginning and end of a block.

III. Iterative Decoding Analysis

A. Decoding Trees

While the implementation of a message passing decoder can be described by a Tanner graph, for the analysis of iterative decoding of a particular code symbol $v_{t,m}^{(h)}$, $t=1,2,...$, $m=1,2,...M$, $h=0,1$, it is more convenient to consider a tree-shaped graph (or a computation tree [26]) for symbol $v_{t,m}^{(h)}$, showing how different code symbols contribute to the decoding of this symbol as the iterations proceed.

Such a tree-shaped graph for an arbitrary code symbol $v_{t,m}^{(h)}$ of an LDPC code can be obtained directly from the Tanner graph. In this case symbol $v_{t,m}^{(h)}$ is called the *clan head*. Each node at the 2ℓth level, $\ell=0,1,...$, of the tree represents one of the code symbols and each node at the $(2\ell+1)$th level, $\ell=0,1,...,$, represents one of the parity-check equations. We call the set of code symbols corresponding to nodes at all the even levels of the tree a *clan*. Symbol $v_{t,m}^{(h)}$ located at the 2ℓth level of the tree is designated as $v_{t,m}^{(h)}(\ell)$. The clan head $v_{t,m}^{(h)}$ is located at level zero (the root of the tree) and is designated as $v_{t,m}^{(h)}(0)$. The set of code symbols at the 2ℓth level, where ℓ is a non-negative integer, is called the ℓth *generation* of the clan. The code symbols in the clan, excluding $v_{t,m}^{(h)}(0)$, are called *descendants* of the clan head $v_{t,m}^{(h)}(0)$. The configuration of the tree, as we will see later, depends on the time instant t corresponding to the position of the clan head $v_{t,m}^{(h)}(0)$ in the Tanner graph.

The edges leaving the root node, together with the nodes at the first level of the tree, correspond to the J parity-check

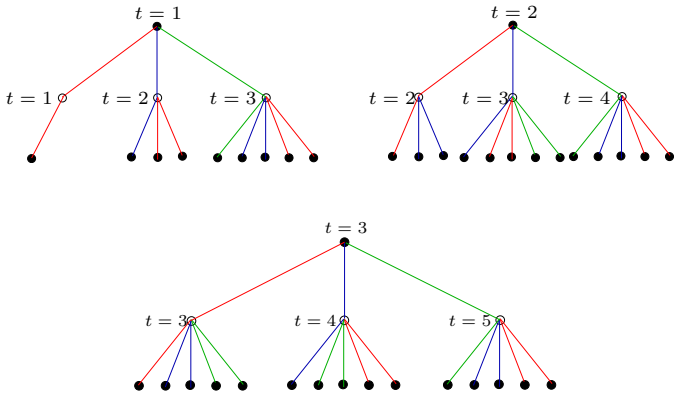

Fig. 4. The first levels of the computation trees for $t = 1$, $t = 2$, $t = 3$ with $J = 3$.

equations that include the clan head. The nodes at this first level are called *families* of the clan head. Each of these nodes, in turn, is connected by edges to nodes at the second level. For LDPCC codes in the ensemble $\mathcal{C}_P(J, 2J, M)$, the number of these edges/nodes can be 1,3,..., or $2J-1$, depending on the time instant of the constraint node. These second level nodes correspond to the code symbols that are included in the same parity-check equations as the clan head. The second level symbols are called *direct descendants* or *children* of the clan head. Hence, the clan head has J families and in each family there can be 1,3,..., or $2J-1$ children. For the trees of the clan heads $v_{t,m}^{(h)}(0)$ at time instants $t = 1, t = 2, t = 3$, the nodes at levels zero, one, and two are illustrated in Figure 4 for the case $J = 3$. Note that for terminated LDPCC codes, the tree of the clan head $v_{t=L,m}^{(h)}(0)$ is analogous to that of $v_{t=1,m}^{(h)}(0)$, and the tree of $v_{t=L-1,m}^{(h)}(0)$ is analogous to that of $v_{t=2,m}^{(h)}(0)$.

Each of the direct descendants of the clan head has $J-1$ families, i.e., $J-1$ edges leave each second level node leading to $J-1$ nodes in the third level of the tree. These $J-1$ third level nodes correspond to the $J-1$ other parity-check equations that include a direct descendant of the clan head. Each of the third level families can include a maximum of $2J-1$ children, i.e., the edges leaving each third level node lead to at most $2J-1$ nodes at the fourth level, and so on. This expansion of levels can be continued indefinitely, and eventually different nodes in the graph will represent the same code symbols, i.e., there is a cycle in the graph.

Using this tree representation, we will be able to analyze the iterative decoding behavior of LDPCC codes.

Definition 1: The $v_{t,m}^{(h)}(0)$-clan is called ℓ_0-nondegenerate if all nodes of the clan up to the ℓ_0th generation correspond to different code symbols, but in the set of nodes of the clan up to the (ℓ_0+1)th generation there is at least one pair of nodes that corresponds to the same symbol. □

Definition 2: A code is called ℓ_0-nondegenerate if the clans of all code symbols are ℓ-nondegenerate, where $\ell \geq \ell_0$, and there exists at least one ℓ_0-nondegenerate clan. □

Note that the girth g of the Tanner graph of an ℓ_0-nondegenerate code satisfies either $g = 4\ell_0+2$ or $g = 4\ell_0+4$, i.e., $\ell_0 = \lceil (g-4)/4 \rceil$. The parameter ℓ_0 was introduced by Gallager [4]. For message passing decoders with standard parallel updating schedule this value defines the maximal number of decoding iterations that can be performed before the independence of the decision statistics becomes invalid. The following theorem affirms the existence of a code in $\mathcal{C}_P(J, 2J, M)$ with ℓ_0 proportional to $\log M$.

Theorem 2: There exists a code in $\mathcal{C}_P(J, 2J, M)$ for which the number of independent decoding iterations, l_0, satisfies

$$l_0 > \frac{\log M}{2\log(2J-1)(J-1)} - c_1 \ , \tag{6}$$

where the constant c_1 does not depend on M. □

Theorem 2 is the convolutional code counterpart of Gallager's theorem [4] which states existence of an LDPC block code with ℓ_0 proportional to the logarithm of the block length, and an analog of Theorem 2 for conventional LDPCC codes was proved in [15]. Theorem 2 can be proved either analogously to [15] or using a random ensemble approach [3]. Note that termination of an LDPCC code can only increase ℓ_0, since termination only reduces the degrees of some symbol nodes.

B. Binary Erasure Channel

In our threshold analysis of LDPCC codes, we will follow the principles developed in [3] for analyzing the error probability of iteratively decodable block codes. Since terminated LDPCC codes are, in fact, a special case of LDPC block codes, we can directly use the methods described in that paper.

Consider first the binary erasure channel, as shown in Fig. 5. The probability of an erasure is ε and with probability $1-\varepsilon$ the transmitted symbol is received correctly. In each iteration,

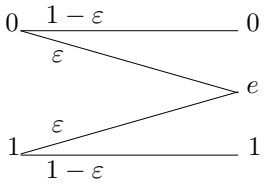

Fig. 5. Binary Erasure Channel.

a message passing decoder exchanges messages between the symbol nodes and the constraint nodes. A symbol node can be recovered correctly if its channel value or at least one message from the constraint nodes to which it connects is not an erasure.

Thus convergence of message passing decoding on the erasure channel can be analyzed by tracking the erasure probability of the messages. This is straightforward as long as the messages exchanged during the iterations are independent. Theorem 2 guarantees the existence of a code in $\mathcal{C}_P(J, 2J, M)$ such that the number of independent iterations is lower-bounded by the right hand side of (6).

Consider the ith iteration of the decoding procedure, where $1 \leq i \leq \ell_0$. Each symbol node is characterized by the time instant t, i.e., the position in the Tanner graph, and by the level 2ℓ of the symbol in the computation tree. Analogously, a constraint node is characterized by the time instant t and by the level $2\ell+1$.

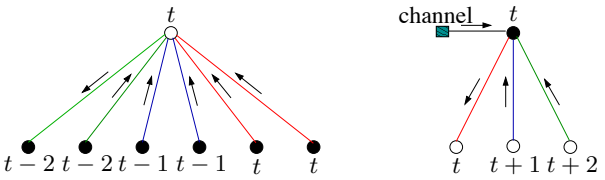

Fig. 6. Illustration of (a) a constraint node update and (b) a symbol node update for the case $J = 3$.

The operations of the message passing decoder can be divided into two stages, as illustrated in Figure 6. In both stages, nodes receive messages from adjacent connected nodes and combine them using a computation rule to form new messages, which are then sent back to the same nodes. The stage where messages received from symbol nodes are combined at a constraint node, we call a constraint node update. The other stage, where messages received from constraint nodes are combined at a symbol node, is called a symbol node update. With the standard parallel updating schedule, first all constraint nodes and then all symbol nodes are updated at each decoding iteration. In this case Theorem 2 guarantees that the number of independent iterations can be made arbitrarily large if M goes to infinity. Note that the nodes involved in an update can belong to different time instants. For an iteration i we study the probability distribution of messages sent by symbol nodes to constraint nodes and the probability distribution of messages sent by constraint nodes to symbol nodes. These probabilities depend on the time instants of the nodes that send and receive the messages, but not on the position of those nodes within a time instant t of the Tanner graph.

The message computed at a constraint node update to send to a connected symbol node (see Figure 6 (a)) is an erasure if at least one of the other $K-1$ neighboring symbol nodes has been erased. It follows that the probability $q^{(i)}_{t+j,t}$ that the message from a constraint node at time instant $(t+j)$ to a symbol node at time instant t, $j \in \{0, J-1\}$, is an erasure is equal to

$$q^{(i)}_{t+j,t} = 1 - \left[\left(1 - p^{(i-1)}_{t,t+j}\right) \prod_{j' \neq j} \left(1 - p^{(i-1)}_{t+j-j',t+j}\right)^2 \right] \quad , \quad (7)$$

for $j, j' \in \{0, J-1\}$, where $p^{(i-1)}_{t,t+j}$ denotes the probability that the message sent in the previous, i.e., the $(i-1)$th, iteration from a symbol node at time instant t to a constraint node at time instant $(t+j)$ corresponds to an erasure. For $i = 0$, these probabilities are initialized as $p^{(0)}_{t,t'} = 1$ for all t and t'. For terminated convolutional codes, we also have the boundary conditions $p^{(i)}_{t,t'} = 0$ if $t < 1$ or $t > L$. This condition takes into account the lower constraint node degrees at the beginning and the end of the Tanner graph.

The message computed at a symbol node update for a constraint node (see Figure 6 (b)) is an erasure if all incoming messages from the neighboring constraint nodes and from the channel are erasures. Thus we have

$$p^{(i)}_{t,t+j} = \varepsilon \prod_{j' \neq j} q^{(i)}_{t+j',t} \quad , \quad j, j' \in \{0, J-1\} \quad . \quad (8)$$

For regular LDPC block codes, the distribution of the messages exchanged in the ith iteration is the same for all nodes regardless of their position within the Tanner graph. Likewise, for the random irregular code ensembles considered in [19], the message distributions are averaged over all codes, and only a single density has to be considered for all constraint and symbol nodes. In the LDPCC code case, while nodes at the same time instant behave identically, the messages from nodes at different time instants can behave differently and must be tracked separately.

Thus the density evolution procedure in this case involves calculating the erasure probabilities $p^{(i)}_{t,t+j}$ in the ith iteration, $1 \leq i \leq \ell_0$, for all time instants t. It follows from [3] that, if $J > 2$, it is sufficient to observe density evolution up to iteration I, when the maximum of $p^{(I)}_{t,t+j}$ over all t and j becomes less than the breakout value

$$B_{\text{br}} = \varepsilon^{-1/(J-2)} (2J-1)^{-(J-1)/(J-2)} \quad .$$

Then, if the number of independent iterations ℓ_0 tends to infinity, the probability that one or more symbols in a block are erased goes to zero at least doubly exponentially with the number of iterations. This defines a strategy for threshold calculation: we perform density evolution up to iteration I, at which point all $p^{(I)}_{t,t+j}$ have crossed the breakout value.

C. Binary-Input Output-Symmetric Memoryless Channels

The threshold analysis for LDPC block codes on an AWGN channel with binary inputs in [3] required observing not only the evolution of the probability density function (PDF) but also the Bhattacharyya parameter. (Note that, for the BEC, the erasure probability $p^{(i)}_{t,t+k}$ is itself the Bhattacharyya parameter.) If $J > 2$, then, as in the BEC case, the observations can be stopped if the Bhattacharyya parameter became less than some breakout value. In this case, the decoding error probability goes to zero at least doubly exponentially with the number of iterations as ℓ_0 and the number of iterations I tend to infinity.

In the case of density evolution for terminated LDPCC codes, the PDFs of the log-likelihood ratios (LLRs) for nodes at different time instants are, generally speaking, different. Thus we must observe the PDFs and the Bhattacharyya parameter separately for each time instant, and the observations can be stopped only when the maximal Bhattacharyya parameter over all time instants becomes less than the breakout value.

In each decoding iteration, messages are exchanged between the symbol nodes and the constraint nodes. At a constraint node update, extrinsic LLRs are computed by decoding the associated single parity-check component code. The message received by the kth symbol node from its jth neighboring constraint node, $j = 1, \ldots, J$, in the ith iteration can be written as

$$\beta^{(i)}_{j,k} = 2\text{arctanh}\left(\prod_{k' \neq k} \tanh(z^{(i-1)}_{k',j}/2) \right) , \quad k, k' \in \{1, K\} \quad , \quad (9)$$

where $z^{(i-1)}_{k',j}$ is the message that the jth constraint node has received from its k'th neighboring symbol node in the $(i-1)$th

iteration. At a symbol node update, the incoming extrinsic LLRs are combined with the intrinsic channel LLR α_k to give the LLRs

$$z_{k,j}^{(i)} = \alpha_k + \sum_{j' \neq j} \beta_{j',k}^{(i)} \ , \quad j, j' \in \{1, J\} \ , \tag{10}$$

which form the messages to be sent by the kth symbol node to the neighboring jth constraint nodes in the ith iteration. Initially, before the first decoding iteration, the LLRs are set to $z_{k,j}^{(0)} = 0$ for the symbols at time instant t, $t = 1, \ldots, L$. For $t < 1$ and $t > L$, the code symbols are defined to be zero, which implies that the LLR messages sent by these nodes are $z_{k,j}^{(i)} = \infty$ for all iterations. This boundary condition automatically takes into account the lower constraint node degrees at the beginning and end of the Tanner graph.

The standard parallel updating schedule assumes that, at each decoding iteration, first all constraint nodes and then all symbol nodes are updated according to (9) and (10), respectively. The messages computed in this way are true LLRs as long as they are produced from independent observations. Theorem 2 guarantees that the number of independent iterations can be made arbitrarily large if M goes to infinity.

Given that all messages are formed from independent observations, it is possible to calculate the evolution of their exact PDFs during the iterations [20]. Since, in general, density evolution must be performed numerically, we follow the approach in [3] and estimate the asymptotic convergence rate by observing not only the PDFs of the LLRs $z_{k,j}^{(i)}$ but also their Bhattacharyya parameter.

Consider again the flow of messages shown in Fig. 6. In Fig. 6(a), the messages $z_{k',j}^{(i-1)}$ in (9) come from nodes, generally speaking, belonging to different time instants. The same holds for the messages $\beta_{j',k}^{(i)}$ that are combined in (10) (see Figure 6(b)). Then, as in the BEC case, the messages from nodes at different time instants must be tracked separately. To take this into account, at each time instant different PDFs must be computed for both $\beta_{j,k}^{(i)}$ and $z_{k,j}^{(i)}$, $k = 1, \ldots, K$, $j = 1, \ldots, J$.

Consider now the ith decoding iteration, where $1 \leq i \leq \ell_0$. Let $\varphi_{t,t+j}^{(i)}(z|0)$ and $\varphi_{t,t+j}^{(i)}(z|1)$ be the PDFs of the messages sent from the symbol node $v_{t,m}^{(h)}$, $m = 1, \ldots, M$, $h = 0, 1$, associated with the tth time instant in the Tanner graph, to one of its neighboring constraint nodes associated with the $(t + j)$th time instant, conditioned on $v_{t,m}^{(h)} = 0$ and $v_{t,m}^{(h)} = 1$, respectively. (Note that the PDF of the messages sent by the symbol $v_{t,m}^{(h)}$ to one of its neighboring constraint nodes at time instant $(t + j)$ depends only on the symbol, its position (time instant) t in the Tanner graph, and the time instant increment j, i.e., they do not depend on h and m.) It follows from symmetry arguments that we also have $\varphi_{t,t+j}^{(i)}(z|0) = \varphi_{t,t+j}^{(i)}(-z|1)$. The Bhattacharyya parameter $B_{t,t+j}^{(i)}$ of these messages, $j = 0, \ldots, J-1$, is equal to

$$\begin{aligned} B_{t,t+j}^{(i)} &= \int_{-\infty}^{\infty} \sqrt{\varphi_{t,t+j}^{(i)}(z|0)\varphi_{t,t+j}^{(i)}(z|1)}dz \\ &= \int_{-\infty}^{\infty} \sqrt{\varphi_{t,t+j}^{(i)}(z|0)\varphi_{t,t+j}^{(i)}(-z|0)}dz \ . \end{aligned} \tag{11}$$

Note that the Bhattacharyya parameter is an upper bound on the bit error probability and hence can serve to estimate its value. The Bhattacharyya parameter A of the intrinsic channel LLRs α is equal to $\exp(-RE_{\rm b}/N_0)$, where $E_{\rm b}/N_0$ is the SNR per bit [3]. The following theorem connects the Bhattacharyya parameters corresponding to the LLRs of two consecutive decoding iterations.

Theorem 3: The Bhattacharyya parameter $B_{t,t+j}^{(i)}$, defined by (11) for $i = 2, \ldots, \ell_0$ and $j = 0, \ldots, J-1$, satisfies the following inequality

$$B_{t,t+j}^{(i)} < A \prod_{j' \neq j} \left(B_{t,t+j'}^{(i-1)} + \sum_{l \neq j'} (B_{t+j'-l,t+j'}^{(i-1)})^2 \right) \ , \tag{12}$$

where $j', l \in \{0, \ldots, J-1\}$ and $B_{t,t+j}^{(1)} = A$. □

The proof of Theorem 3 is analogous to the proof of Lemma 1 in [3]. Inequality (12) also follows from [18, Lemma 4.1 and equation (4.6)].

Now define $B_{\max}^{(i)}$ as the largest value of $B_{t,t+j}^{(i)}$ over all edges in the graph, i.e.,

$$B_{\max}^{(i)} = \max_{t,j} B_{t,t+j}^{(i)} \ , \quad j \in \{0, \ldots, J-1\} \ . \tag{13}$$

Then it follows from (12) that

$$B_{t,t+j}^{(i)} \leq B_{\max}^{(i)} < A \left((2J-1) B_{\max}^{(i-1)} \right)^{J-1} \ . \tag{14}$$

Suppose now that, after some iteration $i, i < \ell_0$, all Bhattacharyya parameters $B_{t,t+j}^{(i)}$ are smaller than the *breakout value*

$$B_{\rm br} = A^{-1/(J-1)} (2J-1)^{-(J-1)/(J-2)} \ . \tag{15}$$

Then, as shown in [3], if the number of independent iterations ℓ_0 goes to infinity, the bit error probability of all symbols converges to zero at least doubly exponentially with the number of iterations.

Instead of observing the evolution of the Bhattacharyya parameter $B_{t,t+j}^{(i)}$, we can also observe the evolution of the *probability of hard decision error* $P_{t,t+j}^{(i)}$ defined[2] as

$$P_{t,t+j}^{(i)} = \int_0^{\infty} \varphi_{t,t+j}^{(i)}(z|1)dz = \int_{-\infty}^{0} \varphi_{t,t+j}^{(i)}(z|0)dz \ . \tag{16}$$

The integrals are the probabilities of error of the hard decisions made with respect to a symbol, given that the symbol is 1 and 0, respectively, after i iterations. Then it follows from (11) that

$$\begin{aligned} B_{t,t+j}^{(i)} &= 2\int_0^{\infty} \sqrt{\varphi_{t,t+j}^{(i)}(z|0)\varphi_{t,t+j}^{(i)}(-z|0)dz} \\ &\leq 2\sqrt{\int_0^{\infty} \varphi_{t,t+j}^{(i)}(z|0)dz \int_0^{\infty} \varphi_{t,t+j}^{(i)}(-z|0)dz} \\ &= 2\sqrt{P_{t,t+j}^{(i)}(1 - P_{t,t+j}^{(i)})} \ , \end{aligned} \tag{17}$$

[2]Strictly speaking, $P_{t,t+j}^{(i)}$ is the probability of hard decision error made on the basis of a message (10) sent by a symbol node to one of its neighboring constraint nodes, i.e., it does not include information sent by this constraint node to the symbol node, but $P_{t,t+j}^{(i)}$ can be easily calculated in the process of density evolution.

were we have used Cauchy's inequality. From (17), it follows that it is sufficient to observe the evolution of

$$P_{\max}^{(i)} = \max_{t,j} P_{t,t+j}^{(i)} \ , \quad j \in \{0, \ldots, J-1\} \ , \qquad (18)$$

until it became less than

$$P_{\text{br}} = B_{\text{br}}^2/4 \ . \qquad (19)$$

This means that we can perform density evolution until the hard decision error probabilities $P_{t,t+j}^{(i)}$ reach a sufficiently low level defined by (19).

To determine the convergence thresholds of terminated codes from the ensemble $\mathcal{C}_P(J, 2J, M)$ it is possible, in principle, to use the standard parallel updating schedule. In this case we must evaluate numerically, iteration by iteration, the different PDFs $\varphi_{t,t+j}^{(i)}(z|0) = \varphi_{t,t+j}^{(i)}(-z|1)$ for all time instants t, $1 \le t \le L$, and for all j, $0 \le j \le J-1$. Note that, in addition to the node degrees J and K, the termination length L is another parameter that influences the result. While increasing L reduces the rate loss of the terminated code, the computational burden of performing density evolution becomes increasingly difficult for larger L. Both the number of different PDFs to be tracked and the number of iterations needed for the effect of the strong nodes at the ends of the graph to propagate to the time instants in the center increase with L. Also, for the AWGN channel, the complexity of density evolution is much greater than for the BEC, where a simple one-dimensional recursion formula can be used. In the next section we therefore consider a sliding window updating schedule that reduces the number of operations required for the threshold computation.

IV. Sliding Window Decoding

In the standard parallel updating schedule, considered in the previous section, first all constraint nodes and then all symbol nodes are updated in each iteration. This is convenient for analysis, since then the computation trees of all symbols have a very regular structure. An alternative schedule, where the constraint nodes are activated one by one and a symbol node is activated whenever a message is demanded by a neighboring constraint node, was considered in [27]. It has been observed from computer simulations that such an *on-demand symbol node updating schedule* can reduce the required number of decoding iterations [9], [27].

In this section we analyze the following message passing schedule for decoding terminated LDPCC codes. Consider the Tanner graph of a terminated LDPCC code, where symbol nodes are labeled by time instants t, $t = 1, \ldots, L$ (see Figure 3). For simplicity, we assume that L is even. Suppose that two decoders operate in turn on different regions of the graph defined by two windows of size not exceeding W, $W \le L/2$. In the k-th decoding round, $k = 1, \ldots, L/2$, the first decoder operates on symbol nodes within the window $t \in [k, \min(k + W - 1, L/2)]$ and the second decoder on symbol nodes within the window $t \in [\max(L/2 + 1, L - W - k + 2), L - k + 1]$, as illustrated in Figure 7.

Within their windows, the decoders perform I_k iterations during round k according to the following updating schedule.

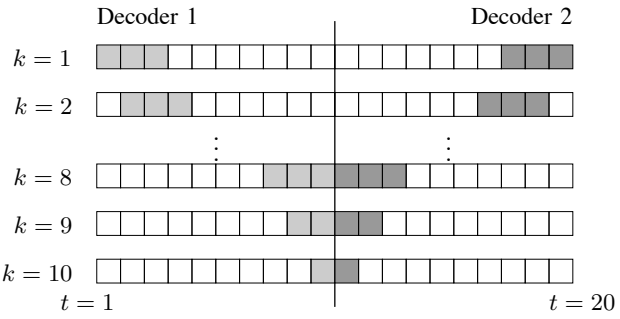

Fig. 7. Decoding windows from symbol node perspective at different rounds k for $W = 3$ and $L = 20$.

Symbol node updates are performed time instant by time instant, from left to right for the first decoder and from right to left for the second decoder. Before each symbol node update, all its neighboring constraint nodes are updated, i.e., we employ the *on-demand constraint node updating schedule*. Compared to the parallel schedule, the contribution of individual nodes to the exchanged messages flows faster through the graph.

The window defines the time instants t of the symbol nodes that are updated in a decoding round k. The activated check nodes and also the symbol nodes of their incoming messages may hence lie outside the window. As a consequence, the two decoders exchange messages when the windows approach the center of the graph, although the windows never overlap.

We now apply density evolution analysis to this schedule, considering first the BEC. Assume that M is large enough so that the decoders operate in cycle-free regions of the graph, i.e., each symbol contributes to a message at most once. The number of iterations in the kth round, I_k, is chosen such that all erasure probabilities $p_{k,k+j}^{(i)}$, $j \in \{0, J-1\}$, defined by (8) become less than some sufficiently small value B_0, where $0 < B_0 < B_{\text{br}}$ (we call this the *breakout value condition*). Then the window is shifted by increasing the starting time instant k by one and the procedure is repeated for round $k+1$. Analogously, in the kth round, the nodes in the second window are updated until the probabilities $p_{L-k+1,L-k+1-j}^{(i)}$, $j \in \{0, \ldots, J-1\}$, reach the value B_0. Then the window is shifted to the position corresponding to the $(k+1)$th round. From the symmetry of the Tanner graph, it follows that the computations of the two decoders are identical, and it is sufficient to perform the calculations for the first decoder[3] for $k = 1, \ldots, L/2$.

For the AWGN channel, density evolution can be performed analogously. In round k the PDFs $\varphi_{t,t+j}^{(i)}(z|0) = \varphi_{t,t+j}^{(i)}(-z|1)$ are evaluated numerically, iteration by iteration, for all time instants t in the window. In parallel, the hard decision error probabilities $P_{k,k+j}^{(i)}$, $j \in \{0, \ldots, J-1\}$, defined by (16) are calculated for the window's leftmost time instant $t = k$. When at some iteration I_k the probability $\max_j P_{k,k+j}^{(I_k)}$ reaches some sufficiently small value P_0, $0 < P_0 < P_{\text{br}}$ (the *breakout value condition*), the window is shifted by increasing the starting time instant k by one and the procedure is repeated for round $k+1$.

[3]Although the first decoder receives messages from the second one, the PDFs of these messages are, by symmetry, equivalent to some of those from decoder one, which are available from earlier computations and need not be evaluated separately.

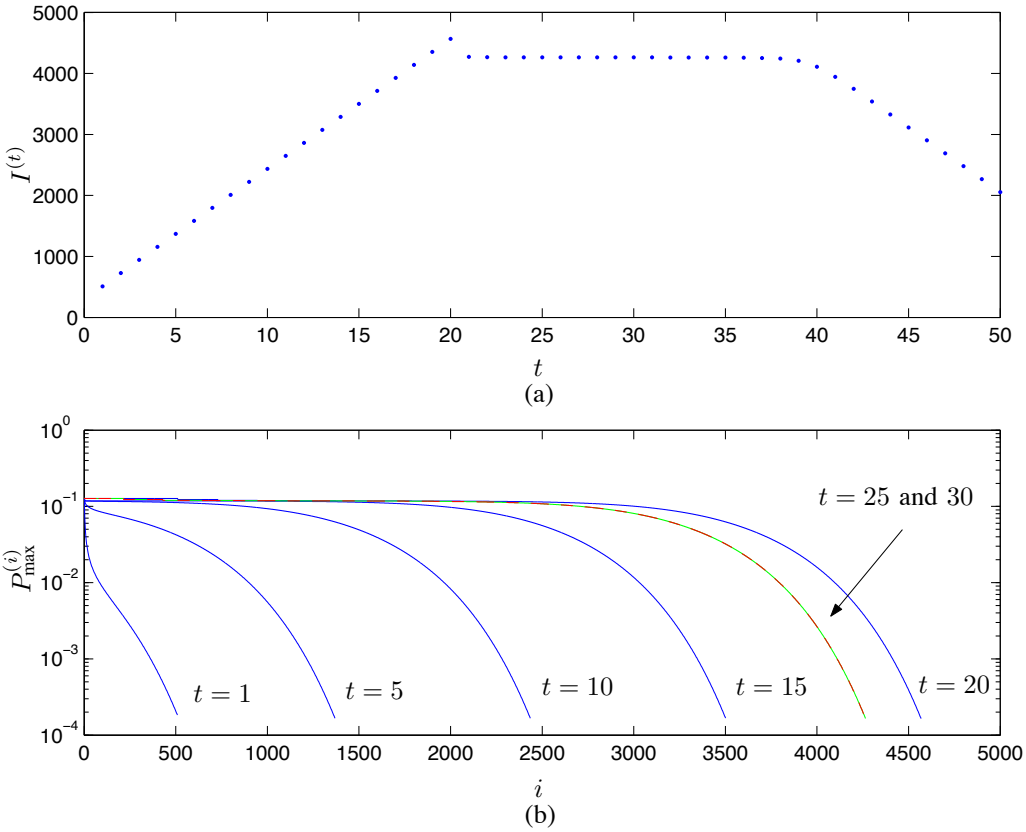

Fig. 8. (a) Total number of iterations $I^{(t)}$ for nodes at time t and (b) evolution of the bit error probability $P_{\max}^{(i)}$ at time t with iterations $i = 1, \ldots, I^{(t)}$ for $J = 3$, $L = 100$, and $W = 20$. The results correspond to an AWGN channel with $E_b/N_0 = 0.55$ dB.

In Section III it was shown that fulfilling the breakout value condition for all symbol nodes implies that the bit error probability converges to zero with an increasing number of iterations. However, this result was derived for a decoder based on the standard parallel updating schedule and the number of independent iterations guaranteed by Theorem 2 relies on this assumption. In general, the node activation order used in decoding determines the particular shape of the computation trees for the different code symbols. Since, for any finite number of node activations, the depths of the computation trees will be finite as well, it follows that these trees can be covered by trees corresponding to a parallel updating schedule with a sufficient number of iterations. Consequently, any node updating schedule can be interpreted as a parallel schedule, where certain node activations are omitted. But, under the independence assumption, such an omission of additional side information can never improve the performance of decoding, which results in the following proposition. (A similar result has also been obtained in [28].)

Proposition 1: Consider density evolution with an arbitrary node activation order. Assume that after a specific number of node activations the Bhattacharyya parameters of all symbol node messages are below some value B_0 and the breakout value condition is satisfied, i.e., $B_0 < B_{\text{br}}$. Then the value B_0 can also be reached for a standard parallel updating schedule after a sufficiently large number of iterations. □

To demonstrate the threshold computation procedure with the window updating schedule, consider an AWGN channel with an SNR of $E_b/N_0 = 0.55$ dB for the case $J = 3$, $L = 100$, and $W = 20$. By definition, I_k iterations are performed within the window in each decoding round k. It follows that the total number of iterations $I^{(t)}$ for each node at time t, $t = 1, \ldots, L/2$, is equal to

$$I^{(t)} = \sum_{k=\max(1,t-W+1)}^{t} I_k \ ,$$

which can be written recursively as

$$I^{(t)} = \begin{cases} I^{(t-1)} + I_t & \text{for } 1 < t \leq W \\ I^{(t-1)} - I_{t-W} + I_t & \text{otherwise} \end{cases} . \qquad (20)$$

In this example, the values $I^{(t)}$ are shown in Fig. 8 (a) for the time instants t within the operation region of the first decoder. The number of iterations I_k at each decoding round k is chosen such that the breakout value condition is satisfied for $t = k$. The chosen channel SNR corresponds to the estimated threshold, which explains the large number of required iterations. In the first round, $I_1 = 509$, but only $I_2 = 220$ iterations are required for the second round, and a similar value is observed for the following rounds $k = 3, \ldots, 37$. As a consequence of (20), the number $I^{(t)}$ increases until a maximum value is reached at $t = W = 20$. Then it remains constant until the effect of the second decoder reduces the number of iterations I_k at times $t = k$ close to the center of the Tanner graph.

The bit error probability $P_{\max}^{(i)}$ at time t, as a function of the number of iterations $i = 1, \ldots, I^{(t)}$, is shown in Fig. 8(b) for different t. Observe from the figure that the $P_{\max}^{(i)}$ curves for time instants $t = 25$ and $t = 30$ are almost indistinguishable. This is actually the case for all time instants t for which $I^{(t)}$ stays constant. This behavior indicates that the calculations tend to repeat themselves at different window positions. From this we may conclude that the effect of the strong nodes at the ends of the Tanner graph carries through to the center, independent of the termination length L.

These observations suggest that detecting convergence at time $t = 1$ is sufficient to determine the overall convergence threshold, which is confirmed for the BEC by the following theorem.

Theorem 4: Consider density evolution for the BEC with erasure probability ε and the window updating schedule for an arbitrary termination length L. Let the message probabilities at times $t < 1$ and $t > L$ be initialized by B_0, $0 < B_0 < B_{\text{br}}$. If, under these conditions, the value B_0 is reached at time $t = 1$ after some number of iterations I_1 in round $k = 1$, so that the window can be shifted one step further, then, for the actual initial probabilities, the value B_0 can be reached at all times t, $t = 1, \ldots, L$. □

Proof: We prove the theorem by induction. Assume that at the beginning of decoding round k the message probabilities from symbol nodes to constraint nodes satisfy

$$p_{t,t+j}^{(i)} = \begin{cases} B_0 & \text{if } t < k \\ \varepsilon & \text{if } t \geq k \end{cases}, \forall j \in \{0, J-1\} \ , \qquad (21)$$

and that after I_k^* further iterations the message probabilities became smaller than or equal to B_0 at time $t = k$ and smaller than or equal to ε at times $t > k$ within the window. After shifting the window, the same condition must then be fulfilled for round $k+1$ after $I_{k+1}^* \leq I_k^*$ iterations[4]. This follows since the message probabilities within the window before round $k+1$ are all smaller than or equal to those in (21). If the messages probabilities at times $t < 1$ are initialized by B_0 before the first round, then it follows by induction that B_0 can be reached at all other times up to the center of the Tanner graph. The fact that the symbols at times $t < 1$ are known due to termination, so that the actual initial probabilities are equal to 0 (which is smaller than B_0), can only improve the performance, which finally proves the statement of the theorem. ■

Theorem 4 allows us to determine *valid* channel values ε for which a decoder with the window updating schedule, under the assumption that all exchanged messages are statistically independent, can reach a given value $B_0 < B_{\text{br}}$ at all time instants $t = 1, \ldots, L$ for arbitrarily large termination lengths L. If a channel value is identified as valid for a specific termination length L and window size W, $W < L/2$, it follows that it is valid for all termination lengths $L' > L$ and even for non-terminated LDPCC codes. According to Proposition 1, if the breakout value can be achieved under the window updating schedule, then it can be achieved under the parallel updating schedule as well, provided that the number of iterations is chosen sufficiently large. But, for the parallel updating schedule, Theorem 2 ensures that, for an arbitrary number ℓ_0 of independent decoding iterations, we can find a suitable code if M is chosen sufficiently large. It then follows from (14) that the bit error probability of all symbols in the terminated code converges to zero at least doubly exponentially with the number of iterations under the standard parallel updating schedule. As a result, Theorem 4 can be used to obtain lower bounds on the density evolution threshold, as stated in the following corollary.

Corollary 5: Consider standard belief propagation decoding of codes obtained from the ensemble $\mathcal{C}_P(J, 2J, M)$ by termination. Any channel value ε that satisfies the condition of Theorem 4 for some given values $L, W < L/2$, and $B_0 < B_{\text{br}}$ represents a *lower bound* on the density evolution threshold ε^* that can be achieved as $M \to \infty$ for any fixed termination length $L' \geq L$. □

We conjecture that the statements of Theorem 4 and Corollary 5 are true for other binary-input memoryless channels as well. Although this is technically difficult to prove, all our calculations for the AWGN channel (see, e.g., Fig. 8) support this conjecture.

In the following section, numerical calculations of thresholds for the BEC and the AWGN channel are given.

V. Results and Discussion

Let us first consider the BEC, for which a simple evaluation of density evolution according to (7) and (8) is possible.

In Table I we present the thresholds $\varepsilon_{\text{conv}}^*$ for terminated (3,6) LDPCC codes of frame length $2LM$ with $(L - \tau)M$ information symbols, where $\tau \leq m_{\text{s}} + 1 = 3$. Assuming equality, the code rate is $R_{\text{conv}} = (L - 3)/(2L)$. The first column shows L, the second column shows the resulting rate R_{conv} of the terminated convolutional code, and the third column gives the threshold $\varepsilon_{\text{conv}}^*$. For $L = 10$, the threshold is quite high, in fact larger than the Shannon limit for rate $R = 1/2$ codes. However, in this case there is a significant rate loss and the rate R_{conv} of the terminated code is only 0.350. For larger L and correspondingly larger rates, the threshold remains constant at a value of 0.488. From Theorem 4, we conclude that the threshold stays at this value for L going to infinity and, consequently, that the rate of the terminated code can be made arbitrarily close to 0.5 without affecting the threshold.

The fourth column of Table I shows the thresholds $\varepsilon_{\text{irr-blk}}^*$ for random irregular LDPC block codes having the same degree distributions as the terminated convolutional codes (see [19] or [20] for a definition of degree distributions). Note that the degree distribution of the irregular codes depends on L and that they have the same rate as the terminated convolutional codes. We observe that the thresholds of the terminated convolutional codes are better than those of the corresponding irregular LDPC block codes. With increasing L, the degree distribution of the terminated convolutional codes tends to that of a (3,6) regular LDPC block code. Therefore

[4] Actually, the message probabilities are then smaller than B_0 or ε. But they could be set equal to B_0 and ε by introducing some artificial additional erasures. The conventional decoder can never perform worse than such a modified decoder.

L	R_{conv}	$\varepsilon^*_{\text{conv}}$	$\varepsilon^*_{\text{irr-blk}}$
10	0.350	0.504	0.501
25	0.440	0.488	0.456
50	0.470	0.488	0.442
100	0.485	0.488	0.436
150	0.490	0.488	0.433

TABLE I
BEC THRESHOLDS FOR THE ENSEMBLE $\mathcal{C}_P(J, 2J, M)$. CORRESPONDING IRREGULAR BLOCK CODE THRESHOLDS ARE SHOWN FOR COMPARISON.

L	Iterations checked at all t	Iterations checked only at $t = 1$
18	523	514
20	1113	1090
30	5918	3060
50	15912	3064
100	40899	3064

TABLE II
BEC THRESHOLD CALCULATION WITH PARALLEL UPDATING SCHEDULE FOR THE ENSEMBLE $\mathcal{C}_P(3, 6, M)$: THE INFLUENCE OF L ON THE REQUIRED NUMBER OF ITERATIONS.

it is not surprising that the thresholds of the corresponding irregular LDPC block codes tend to the threshold of a (3,6) regular LDPC block code, which is equal to $\varepsilon^*_{\text{reg-blk}} = 0.429$. However, the thresholds of the terminated LDPC convolutional codes remain unchanged as L increases. The improvement in threshold can be attributed to the structure imposed on the Tanner graph by the convolutional nature of the code.

Table II shows how L influences the required number of iterations with the parallel updating schedule. The second column gives the number of iterations required for the erasure probability of all messages to satisfy the breakout condition for the value $B_0 = 10^{-5}$. The third column gives the number of iterations for the case when the breakout condition is checked for symbol nodes at time $t = 1$ only. The results in Table II reflect the fact that, for larger L, the messages from the stronger nodes at the ends of the Tanner graph need more time to affect the symbols in the center.

Corresponding values for the sliding window approach are shown in Table III for $L = 100$ and different window sizes W. In this example, the number of iterations in the first decoding round $I^{(1)}$ stays constant for $W \geq 30$, indicating that a larger window will not further improve the bit error probability performance. As shown in the table, convergence at the threshold $\varepsilon^*_{\text{conv}}$ can be guaranteed for any $W \geq 14$. However, for $W < 30$, the rate of convergence is slower and the value $I^{(1)}$ increases.

W	$I^{(1)}$	Average Iterations	
		checked at all t	checked only at $t = 1$
14	3088	15899	865
15	2294	9987	688
20	2015	10246	806
30	2012	13820	1207
40	2012	16114	1610
on demand		26697	2004

TABLE III
BEC THRESHOLD CALCULATION WITH WINDOW UPDATING SCHEDULE FOR THE ENSEMBLE $\mathcal{C}_P(3, 6, M)$ WITH $L = 100$: THE INFLUENCE OF W ON THE REQUIRED NUMBER OF ITERATIONS. ON DEMAND UPDATING SCHEDULE WITHOUT WINDOW IS SHOWN FOR COMPARISON.

(J, K)	R_{conv}	$\varepsilon^*_{\text{conv}}$	$\varepsilon^*_{\text{blk}}$
(3,6)	0.49	0.488	0.429
(4,8)	0.49	0.497	0.383
(5,10)	0.49	0.499	0.341

TABLE IV
BEC THRESHOLDS OF TERMINATED LDPCC CODES FOR THE ENSEMBLES $\mathcal{C}_P(M, 2M, J)$ WITH DIFFERENT J.

A measure of complexity for the threshold calculation in density evolution is the average number of iterations over all time instants. For larger windows this value increases, since the nodes at a given time instant are activated during a larger number of decoding rounds. In Table III, the required number of iterations for the on demand updating schedule without a window is also shown for comparison, and from Tables II and III we see that the number of iterations can be reduced from 40899 to 26697 without a window. For a window of size $W = 15$, this number can be further reduced to 9987 if the symbol nodes at all time instants are checked. According to Proposition 1, the threshold is the same for all these approaches. Finally, using Theorem 4, only 688 iterations are required. This example illustrates the large computational savings obtained from the sliding window approach, which is especially useful for the AWGN channel, where the numerical representation of the message distributions and hence the computations within density evolution are more involved.

In Table IV, thresholds for the BEC are presented for different J. In each case L is chosen so that there is a rate loss of 2%. The first column in Table IV shows the values J and K of the underlying convolutional code, the second column the rate R_{conv} of the terminated code, the third column the threshold $\varepsilon^*_{\text{conv}}$, and the fourth column the threshold $\varepsilon^*_{\text{blk}}$ for randomly chosen regular LDPC block codes with the same J and K. We again observe that the thresholds of the terminated convolutional codes are much better than for the corresponding block codes. This is reasonable, since the constraint nodes at either end of the Tanner graph in the terminated convolutional codes have lower degrees than in the block codes. i.e., the terminated convolutional codes have a structured irregularity.

Interestingly, the terminated convolutional codes with higher J have better thresholds than those with lower J. This behavior is different from that of randomly constructed (J, K) regular LDPC block codes, where, for a fixed rate, increasing J typically results in worse thresholds, since it is necessary to increase K accordingly[5]. For randomly constructed regular LDPC block codes, the loss due to the higher constraint node degrees outweighs the gain resulting from the higher symbol node degrees, adversely affecting performance. However, in the LDPCC code case, the codes with higher J still have strong constraint nodes with low degrees at either end of the Tanner graph. Thus the symbols at the ends are better protected for codes with larger symbol degrees, and this results in better thresholds.

Oswald and Shokrollahi have constructed LDPC block code ensembles from sequences of right regular degree distributions

[5]Recall that symbol nodes with higher degrees are stronger than those with lower degrees, but higher degree constraint nodes are weaker than lower degree constraint nodes.

(4,8) should be (5,10) in the legend
this is a typo, M Lentmaier

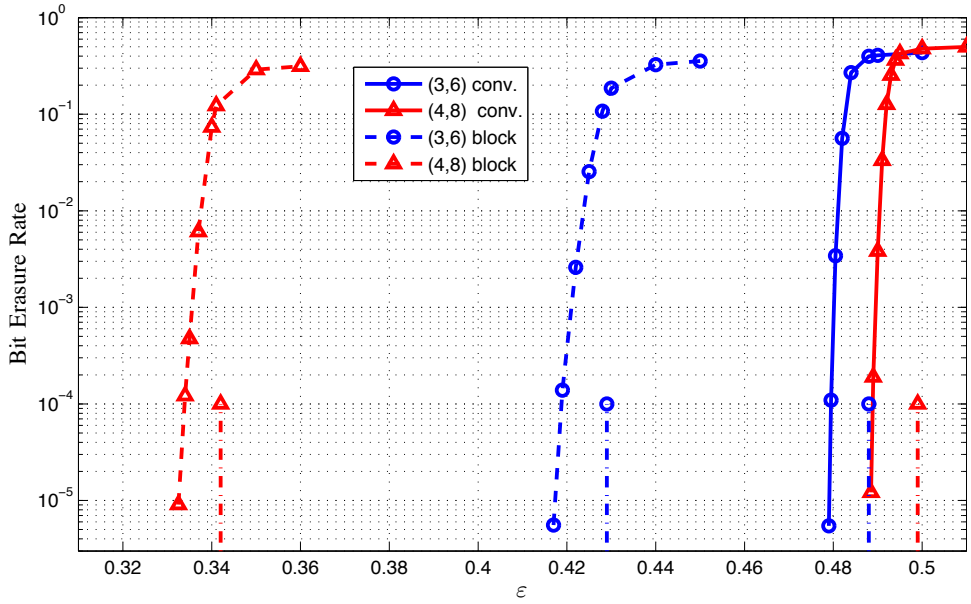

Fig. 9. Simulation results of terminated LDPCC codes from the ensembles $\mathcal{C}(6000, 12000, 3)$ and $\mathcal{C}(6000, 12000, 5)$ (solid lines) for $L = 100$ and $L = 200$, respectively. Regular LDPC block codes composed of permutation matrices of the same size $M = 6000$ are shown for comparison (dashed lines). The vertical dashed-dotted lines show the convergence thresholds for the different ensembles.

(with fixed constraint node degrees) for which the convergence thresholds of the bit erasure probability approach the Shannon limit as the maximum symbol node degree tends to infinity [29]. According to Table IV, with increasing J the convergence thresholds of rate $R = 0.49$ terminated LDPCC codes also tend toward the Shannon limit for rate $R = 1/2$ codes. However, in this case, the degree distributions are not right regular but left regular, i.e., all symbol nodes have the same degree.

We now present the thresholds of terminated LDPCC codes for a binary input AWGN channel, computed by discretized density evolution [30]. In Table V, the thresholds $(E_\mathrm{b}/N_0)^*$, calculated using the conventional parallel updating schedule described in Section III, are presented for different J. The values of L were chosen such that there is a rate loss of 2%, i.e., $R = 0.49$, and the same values can be obtained using the sliding window updating schedule described in Section IV. Also shown in the table are the threshold values $(E_\mathrm{b}/N_0)^{**}$, calculated for the case $L \to \infty$, i.e., $R = 1/2$, using the sliding window updating schedule, under the assumption that Theorem 4 can be generalized to the AWGN channel. Finally, the right hand side of the table shows the thresholds $(E_\mathrm{b}/N_0)^{**}_\mathrm{blk}$ of regular LDPC block codes with the same J.

The results of Table V for the AWGN channel are similar to the results of Table IV for the BEC. In particular, we observe that the thresholds of regular LDPCC codes are much better than those of the corresponding regular LDPC block codes and that the thresholds improve with increasing J.

In Table VI the thresholds of terminated LDPCC codes are compared to upper bounds on the ML decoding thresholds of regular LDPC block codes. The given upper bounds, which were derived in [21], [22], follow from a lower bound on the conditional entropy of the transmitted codeword given the received sequence at the channel output. These information

(J, K)	R	$(E_\mathrm{b}/N_0)^*$	R	$(E_\mathrm{b}/N_0)^{**}$	$(E_\mathrm{b}/N_0)^{**}_\mathrm{blk}$
(3,6)	0.49	0.55 dB	0.5	0.46 dB	1.11 dB
(4,8)	0.49	0.35 dB	0.5	0.26 dB	1.26 dB
(5,10)	0.49	0.30 dB	0.5	0.21 dB	2.05 dB

TABLE V
THRESHOLDS FOR THE ENSEMBLES $\mathcal{C}_P(J, 2J, M)$ WITH DIFFERENT J FOR AWGN CHANNEL WITH BINARY INPUT.

(J, K)	$\varepsilon^*_\mathrm{conv}$	$\varepsilon^*_\mathrm{blk,ML}$	$(E_\mathrm{b}/N_0)^{**}$	$(E_\mathrm{b}/N_0)^{**}_\mathrm{blk,ML}$
(3,6)	0.488	0.49134	0.46 dB	0.371 dB
(4,8)	0.497	0.49798	0.26 dB	0.240 dB
(5,10)	0.499	0.49951	0.21 dB	0.203 dB

TABLE VI
BP THRESHOLDS FOR TERMINATED CODES FROM THE ENSEMBLES $\mathcal{C}_P(M, 2M, J)$ IN COMPARISON WITH UPPER BOUND ON THE ML THRESHOLDS OF THE CORRESPONDING REGULAR BLOCK CODES.

theoretic bounds are given in terms of the degree distribution of the parity-check nodes of an arbitrary code representation and stay valid for any sequence of codes whose block error probability vanishes. Surprisingly, the values in Table VI show that terminated LDPCC codes almost achieve the bounds on the ML thresholds even under the sub-optimal and practical iterative belief propagation decoding algorithm. The results also demonstrate the tightness of the information theoretic bounds, which are easy to compute and can be useful as a starting point in searching for the iterative decoding thresholds, which is computationally intensive due to the large number of iterations required to obtain low bit error probabilities at rates close to capacity.

Finally, simulation results of some randomly chosen terminated LDPCC codes with $J = 3$ ($L = 100$) and $J = 5$ ($L = 200$) are given in Fig. 9 and compared to the corresponding regular LDPC block codes. The choice of L leads to an equal

rate $R = 0.49$ for both LDPCC codes[6]. The curves in Fig. 9 are obtained with randomly selected permutation matrices of size $M = 6000$ under standard belief propagation decoding with the parallel updating schedule. In order to demonstrate that low error rates close to the calculated thresholds are achievable, a maximum number of 5000 iterations was allowed in the simulations to allow a progression of reliable messages through all time instants $t = 1, \ldots, L$. Note that, although the overall block length $N = 2LM$ increases with L, for a fixed M the performance improves with decreasing L at the expense of a higher rate loss, as indicated in Table I. On the other hand, after some sufficiently large L, convergence to a steady behavior is expected (also seen in Table I), since the potential strength of a convolutional code in $\mathcal{C}(M, 2M, J)$ is determined by its constraint length $\nu = 2M(m_{\rm s} + 1)$, which is independent of the termination length L. While the number of required decoding iterations under the parallel updating schedule increases significantly with L (see also Table II) , the analysis in Section IV suggests that a sliding window decoder can provide a good performance complexity trade-off with an iteration number per decoded symbol that is independent of the termination length L. We see this as a promising direction for further research.

VI. Conclusion

In this paper, we have analyzed the iterative decoding thresholds of a special class of terminated rate $R = 1/2$, $(J, 2J)$ regular LDPCC codes for the BEC and the AWGN channel. We show that these thresholds not only are better than the belief propagation decoding thresholds of corresponding regular LDPC block codes, but that they are even very close to the ML decoding thresholds of the latter. To calculate the thresholds for a particular termination length, we made use of the density evolution technique that is commonly used to calculate the thresholds of LDPC block codes, noting that the particular Tanner graph structure of the terminated LDPCC codes must be taken into account. To calculate the thresholds for large termination lengths, we proposed a sliding window updating schedule. Using this approach, we proved for the BEC that the calculated thresholds are valid for arbitrarily large termination lengths, and we conjecture that this is also true for the AWGN channel and for unterminated LDPCC codes. Although we considered only rate $R = 1/2$ codes, we can expect similar behavior for other rates. It is especially interesting that the thresholds of regular LDPCC codes, in contrast to the thresholds of regular LDPC block codes, improve with increasing density of the parity-check matrices.

Appendix

Sketch of the proof of the Theorem 1.

Let the vector $\boldsymbol{v}_{L'} = (\boldsymbol{v}_1^{(0)}, \boldsymbol{v}_1^{(1)}, \ldots, \boldsymbol{v}_{L'}^{(0)}, \boldsymbol{v}_{L'}^{(1)})$, $\boldsymbol{v}_t^{(h)} \in \mathbb{F}_2^M$, for $t = 1, \ldots L'$, $h = 0, 1$, with $\boldsymbol{v}_1^{(0)} \neq \boldsymbol{0}$ denote the systematically encoded codeword corresponding to an information block of length L' starting at $t = 1$. For each time instant t, $1 \leq t \leq L'$, the vector $\boldsymbol{v}_{L'}$ consists of an information block $\boldsymbol{v}_t^{(0)}$ of length M and the corresponding length M parity block $\boldsymbol{v}_t^{(1)}$. For a code C in $\mathcal{C}_P(J, 2J, M)$, we define the *partial syndrome vector* [9] associated with $\boldsymbol{v}_{L'}$ at time t as $\boldsymbol{s}_t = (\boldsymbol{s}_{(t,1)}, \ldots, \boldsymbol{s}_{(t,m_s)})$, where

$$\boldsymbol{s}_{(t,j)} = [\boldsymbol{v}_t^{(0)}, \boldsymbol{v}_t^{(1)}]\boldsymbol{H}_j^{\rm T}(t + j) + \boldsymbol{s}_{(t-1,j+1)},\ j = 1, \ldots, m_s\ , \tag{22}$$

and, by definition, $\boldsymbol{s}_{(0,j)} = \boldsymbol{0}$ for all j, $\boldsymbol{s}_{(t,j)} = \boldsymbol{0}$, $1 \leq t \leq L'$, $j > m_s$, and each $\boldsymbol{s}_{(t,j)}$, $j = 1, \ldots, m_s$, is an M-dimensional vector. Encoding can be carried out by solving the equation

$$\boldsymbol{0} = [\boldsymbol{v}_t^{(0)}, \boldsymbol{v}_t^{(1)}]\boldsymbol{H}_0^{\rm T}(t) + \boldsymbol{s}_{(t-1,1)}, \tag{23}$$

to determine the parity-block $\boldsymbol{v}_t^{(1)}$ for each t. Thus knowledge of the partial syndrome vector $\boldsymbol{s}_{t-1}$ at time $t - 1$ and the information block $\boldsymbol{v}_t^{(0)}$ makes the future independent of the past. The syndrome former trellis can be defined using the partial syndrome vectors as states. Equations (22) and (23) define the transitions, i.e., the next state and branch labels, of this trellis. The initial syndrome former state $\boldsymbol{s}_0$ is the zero vector, i.e., we start from the all-zero state.

To terminate the code to a total length $N = 2LM$, where $L = L' + \tau$, we seek a length $2\tau M$ tail vector $\widetilde{\boldsymbol{v}}_{[L'+1,L'+\tau]} = (\widetilde{\boldsymbol{v}}_{L'+1}^{(0)}, \widetilde{\boldsymbol{v}}_{L'+1}^{(1)}, \ldots, \widetilde{\boldsymbol{v}}_{L'+\tau}^{(0)}, \widetilde{\boldsymbol{v}}_{L'+\tau}^{(1)})$, $\widetilde{\boldsymbol{v}}_{L'+t}^{(h)} \in \mathbb{F}_2^M$, for $t = 1, \ldots, \tau$, $h = 0, 1$, such that the final syndrome former state $\boldsymbol{s}_{L'+\tau}$ is the zero vector, i.e., we terminate back to the all-zero state. This is equivalent to solving the system of linear equations

$$\widetilde{\boldsymbol{v}}_{[L'+1,L'+\tau]}\boldsymbol{H}_{[L'+1,L'+\tau]}^{\rm T} = (\boldsymbol{s}_{L'}, \boldsymbol{0}, \ldots, \boldsymbol{0})_{1\times(\tau+m_s)M}, \tag{24}$$

where the $2\tau M \times (\tau + m_s)M$ matrix $\boldsymbol{H}_{[L'+1,L'+\tau]}^{\rm T}$ is given by (25).

To continue, we need the following property of the partial syndrome vector for codes in $\mathcal{C}_P(J, 2J, M)$.

Lemma 1 ([24]): For any code C in $\mathcal{C}_P(J, 2J, M)$, the components $\boldsymbol{s}_{(t,j)}$ of a partial syndrome vector $\boldsymbol{s}_t$ have even weight. □

Theorem 1 is proved if we show that, for $\tau = m_s + 1$ and for almost all codes in $\mathcal{C}_P(J, 2J, M)$, equation (24) has a solution for any valid partial syndrome vector $\boldsymbol{s}_{L'}$, i.e., a vector with components of even weight. In other words, we show that any sequence $\boldsymbol{v}_{L'}$ can be terminated with a tail of $2M \cdot \tau = 2M(m_s + 1)$ bits. For simplicity, we consider below the ensemble $\mathcal{C}_P(3, 6, M)$. In this case $\tau = m_s + 1 = 3$ and the tail bits are determined by solving the equation

$$\widetilde{\boldsymbol{v}}_{[L'+1,L'+3]}\boldsymbol{H}_{[L'+1,L'+3]}^{\rm T} = (\boldsymbol{s}_{L'}, \boldsymbol{0}, \boldsymbol{0}, \boldsymbol{0})_{1\times 5M}\ , \tag{26}$$

where the $6M \times 5M$ matrix $\boldsymbol{H}_{[L'+1,L'+3]}^{\rm T}$ is given by (27) and the sub-matrices $\boldsymbol{H}_i^{\rm T}(t)$ are composed of two $M \times M$ permutation matrices. Figure 10 shows the permutation matrices that comprise $\boldsymbol{H}_{[L'+1,L'+3]}^{\rm T}$. The existence or lack of solutions of (26) depends on the rank of $\boldsymbol{H}_{[L'+1,L'+3]}^{\rm T}$. We now proceed to show that $\boldsymbol{H}_{[L'+1,L'+3]}^{\rm T}$ has almost full rank.

For convenience, let $\boldsymbol{A}$ denote the matrix $\boldsymbol{H}_{[L'+1,L'+3]}^{\rm T}$. With $\boldsymbol{A}$ viewed as a 6×5 block matrix, let $\boldsymbol{A}_{(i,j)}$, $i = 1, \ldots, 6$, $j = 1, \ldots, 5$, denote the $M \times M$ matrix in

[6]Both selected codes can be terminated within $\tau = m_{\rm s} = J - 1$ time instants.

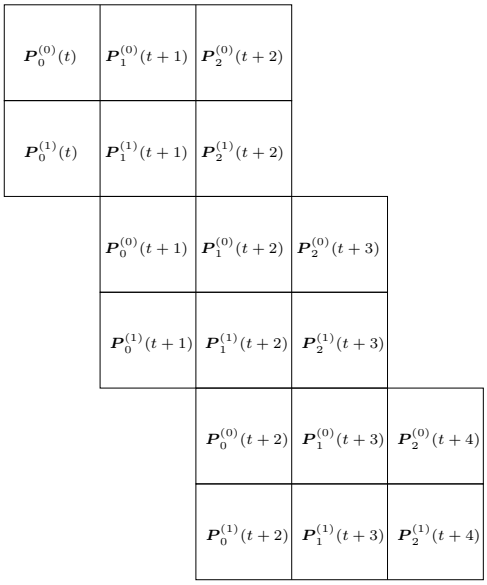

Fig. 10. The matrix $\boldsymbol{H}^{\mathrm{T}}_{[L'+1,L'+3]}$.

the ith row and jth column of $\boldsymbol{A}$. Note that the matrices $\boldsymbol{A}_{(i,j)}$ are either permutation matrices or the all-zero matrix. Consider the ensemble $\mathcal{A}$ consisting of matrices of the form $\boldsymbol{A}$ with probability distribution inherited from the ensemble $\mathcal{C}_P(3,6,M)$, so that each of the permutation sub-matrices comprising $\boldsymbol{A}$ is chosen independently and all $M!$ values are equally likely. We now perform row operations on $\boldsymbol{A}$ to reduce it to diagonal form, using the following property of permutation matrices:

For any two permutation matrices $\boldsymbol{P}_1$, $\boldsymbol{P}_2$, there exists a permutation matrix $\boldsymbol{P}_3$ such that

$$\boldsymbol{P}_1\boldsymbol{P}_3 = \boldsymbol{P}_2 \ . \tag{28}$$

This follows from the facts that a permutation matrix is invertible and that the product of any two permutation matrices is another permutation matrix. In fact, $\boldsymbol{P}_3 = \boldsymbol{P}_1^{-1}\boldsymbol{P}_2$.

The row operations on $\boldsymbol{A}$ are carried out by viewing it as a 6×5 matrix with $M \times M$ matrices as entries. We next consider a four step procedure of matrix transformation. Note that operations in the procedure do not change the rank of the matrix on which we operate.

Step 1: Let $\boldsymbol{C}_{(2,1)}$ be such that $\boldsymbol{A}_{(1,1)}\boldsymbol{C}_{(2,1)} = \boldsymbol{A}_{(2,1)}$. Multiply each of the matrices $\boldsymbol{A}_{(1,j)},\ j = 1,\ldots,6$, in the first row by $\boldsymbol{C}_{(2,1)}$ and add the products to the matrices $\boldsymbol{A}_{(2,j)},\ j = 1,\ldots,6$, in the second row. Then the matrix $\bar{\boldsymbol{A}}_{(2,1)}$ in the first column of the second row, i.e., in position $(2,1)$, of the resulting matrix is the all-zero matrix. The two non-zero entries in the second row of the resulting matrix, i.e., the matrices in positions $(2,2)$ and $(2,3)$, are sums of two permutation matrices. Further, each of these matrices can be assumed to be chosen independently and equally likely. We now repeat the same process for the third and fourth and the fifth and sixth rows of $\boldsymbol{A}$ so that the matrices $\bar{\boldsymbol{A}}_{(4,2)}$ and $\bar{\boldsymbol{A}}_{(6,3)}$ in positions $(4,2)$ and $(6,3)$ of the resulting matrix $\bar{\boldsymbol{A}}$ are the all-zero matrix. Step 2: The matrix in position $(2,2)$ of $\bar{\boldsymbol{A}}$ is the sum of two permutation matrices, say $\bar{\boldsymbol{A}}^{(1)}_{(2,2)}$ and $\bar{\boldsymbol{A}}^{(2)}_{(2,2)}$. Let $\bar{\boldsymbol{C}}^{(1)}_{(2,2)}$ and $\bar{\boldsymbol{C}}^{(2)}_{(2,2)}$ be such that $\bar{\boldsymbol{A}}_{(3,2)}\bar{\boldsymbol{C}}^{(1)}_{(2,2)} = \bar{\boldsymbol{A}}^{(1)}_{(2,2)}$ and $\bar{\boldsymbol{A}}_{(3,2)}\bar{\boldsymbol{C}}^{(2)}_{(2,2)} = \bar{\boldsymbol{A}}^{(2)}_{(2,2)}$. Multiply each of the matrices $\bar{\boldsymbol{A}}_{(3,j)},\ j = 1,\ldots,6$, by $\bar{\boldsymbol{C}}^{(1)}_{(2,2)} + \bar{\boldsymbol{C}}^{(2)}_{(2,2)}$ and add the products to the second row. Then the matrix $\tilde{\boldsymbol{A}}_{(2,2)}$ in position $(2,2)$ of the resulting matrix $\tilde{\boldsymbol{A}}$ is the all-zero matrix, the matrix $\tilde{\boldsymbol{A}}_{(2,3)}$ in position $(2,3)$ is the sum of four permutation matrices, and the matrix $\tilde{\boldsymbol{A}}_{(2,4)}$ in position $(2,4)$ is the sum of two permutation matrices. We then perform a similar operation on the fourth and fifth rows so that the matrix $\tilde{\boldsymbol{A}}_{(4,3)}$ in position $(4,3)$ of the resulting matrix $\tilde{\boldsymbol{A}}$ is the all-zero matrix.

Step 3: The matrix $\tilde{\boldsymbol{A}}_{(2,3)}$ in position $(2,3)$ of $\tilde{\boldsymbol{A}}$ is the sum of four permutation matrices, say $\tilde{\boldsymbol{A}}^{(i)}_{(2,3)},\ i = 1,\ldots,4$. Once again we determine matrices $\tilde{\boldsymbol{C}}^{(i)}_{(2,3)},\ i = 1,\ldots,4$, such that $\bar{\boldsymbol{A}}_{(5,3)}\tilde{\boldsymbol{C}}^{(i)}_{(2,3)} = \bar{\boldsymbol{A}}^{(i)}_{(2,2)},\ i = 1,\ldots,4$. Then multiplying the matrices $\bar{\boldsymbol{A}}_{(5,j)},\ j = 1,\ldots,6$, by $\sum_{i=1}^{4}\tilde{\boldsymbol{C}}^{(i)}_{(2,3)}$ and adding the products to the second row of $\tilde{\boldsymbol{A}}$ makes the matrix $\hat{\boldsymbol{A}}_{(2,3)}$ in position $(2,3)$ of the resulting matrix $\hat{\boldsymbol{A}}$ equal to the all-zero matrix, and the matrices $\hat{\boldsymbol{A}}_{(2,4)}$ and $\hat{\boldsymbol{A}}_{(2,5)}$ in positions $(2,4)$ and $(2,5)$ are sums of six and four permutation matrices, respectively.

Step 4: Now reorder the rows of $\hat{\boldsymbol{A}}$. The first and sixth row of matrices in the resulting matrix $\boldsymbol{A}'$ are left unchanged. The second row is moved to the fourth row, the third row to the second, the fourth to the fifth, and the fifth to the third. Observe that the 3×3 sub-matrix in the upper left corner of $\boldsymbol{A}'$ comprised of the matrices $\boldsymbol{A}'_{(i,j)},\ i = 1,\ldots,3,\ j = 1,\ldots,3$, has full rank, i.e., $3M$.

We now state the following lemma, which is proved in [24].

Lemma 2: The probability that the 3×3 sub-matrix $\boldsymbol{B}^{\mathrm{T}}$ in the lower right corner of $\boldsymbol{A}'$ has rank less than $2M-2$ tends to zero as $M \to \infty$. □

From Lemma 2 and the definition of the matrix $\boldsymbol{A}$ follows another lemma.

$$\boldsymbol{H}^{\mathrm{T}}_{[L+1,L+\tau]} = \begin{bmatrix} \boldsymbol{H}^{\mathrm{T}}_0(L+1) & \ldots & \boldsymbol{H}^{\mathrm{T}}_{m_s}(L+m_s+1) & & \\ & \ddots & & \ddots & \\ & & \boldsymbol{H}^{\mathrm{T}}_0(L+\tau) & \ldots & \boldsymbol{H}^{\mathrm{T}}_{m_s}(L+\tau+m_s) \end{bmatrix} \tag{25}$$

$$\boldsymbol{H}^{\mathrm{T}}_{[L+1,L+3]} = \begin{bmatrix} \boldsymbol{H}^{\mathrm{T}}_0(L+1) & \boldsymbol{H}^{\mathrm{T}}_1(L+2) & \boldsymbol{H}^{\mathrm{T}}_2(L+3) & & \\ & \boldsymbol{H}^{\mathrm{T}}_0(L+2) & \boldsymbol{H}^{\mathrm{T}}_1(L+3) & \boldsymbol{H}^{\mathrm{T}}_2(L+4) & \\ & & \boldsymbol{H}^{\mathrm{T}}_0(L+3) & \boldsymbol{H}^{\mathrm{T}}_1(L+4) & \boldsymbol{H}^{\mathrm{T}}_2(L+5) \end{bmatrix} \tag{27}$$

Lemma 3: The probability that the rank of any matrix $\boldsymbol{A}$ in the ensemble $\mathcal{A}$ is less than $5M-2$ tends to zero as $M \to \infty$.

Proof: From the row operations described in the above four step procedure, it follows that $\boldsymbol{A}$ has rank $3M$+rank($\boldsymbol{B}^{\mathrm{T}}$). Lemma 2 implies that the probability that rank $(\boldsymbol{B}^{\mathrm{T}}) < 2M-2$ tends to zero as $M \to \infty$. This proves the lemma. □

Using Lemma 3 we now prove the existence of solutions to (26).

Lemma 4: Equation (26) has a solution for almost all codes in $\mathcal{C}_P(3, 6, M)$.

Proof: Consider the matrix $\boldsymbol{H}^{\mathrm{T}}_{[L'+1,L'+3]}$. Since the sum of columns of two permutation matrices is the zero vector, it follows that the sum of the first, third, and fourth blocks of columns of $\boldsymbol{H}^{\mathrm{T}}_{[L'+1,L'+3]}$ is the zero vector. Similarly, the sum of the second, third, and fifth blocks of columns is the zero vector. This implies that there are at least two dependent columns in $\boldsymbol{H}^{\mathrm{T}}_{[L'+1,L'+3]}$, or equivalently two dependent equations in (26). From Lemma 3 it follows that, for almost all codes, these are the only two dependent equations in (26). Further, Lemma 1 implies that both components of the partial syndrome state $\boldsymbol{s}_{L'}$ have even weight, i.e., the sum of its components is zero. It is easy to check that the components of the $5M$-dimensional vector $(\boldsymbol{s}_{(L',1)}, \boldsymbol{s}_{(L',2)}, \boldsymbol{0}, \boldsymbol{0}, \boldsymbol{0})$ corresponding to the columns of $\boldsymbol{H}^{\mathrm{T}}_{[L'+1,L'+3]}$ that sum to zero (i.e., the columns corresponding to the blocks of permutation matrices that sum to zero) also sum to zero, i.e., the two dependent equations are consistent. Hence, it follows that (26) has a solution. □

Theorem 1 now follows from Lemma 4.

ACKNOWLEDGMENT

We would like to thank Dr. Igal Sason for pointing out and providing his bounds on the ML decoding thresholds. Furthermore we are grateful for the use of the high performance computing facilities of the ZIH at TU Dresden.

REFERENCES

[1] A. Sridharan, M. Lentmaier, D. J. Costello, Jr., and K. S. Zigangirov, "Convergence analysis of a class of LDPC convolutional codes for the erasure channel," in *Proceedings of the 42nd Allerton Conference on Communication, Control, and Computing*, Monticello, IL, USA, 2004, pp. 953–962.

[2] M. Lentmaier, A. Sridharan, K. S. Zigangirov, and D. J. Costello, Jr., "Terminated LDPC convolutional codes with thresholds close to capacity," in *Proc. IEEE International Symposium on Information Theory*, Adelaide, Australia, Sept. 2005, pp. 1372–1376.

[3] M. Lentmaier, D. V. Truhachev, K. S. Zigangirov, and D. J. Costello, Jr., "An analysis of the block error probability performance of iterative decoding," *IEEE Transactions on Information Theory*, vol. IT-51, no. 11, pp. 3834–3855, November 2005.

[4] R. G. Gallager, *Low-density parity-check codes*. M.I.T. Press, Cambridge, Massachusetts, 1963.

[5] R. M. Tanner, "Error-correcting coding system," *Patent No. 4,295,218*, Oct. 13, 1981.

[6] A. Jiménez Feltström and K. S. Zigangirov, "Periodic time-varying convolutional codes with low-density parity-check matrix," *IEEE Transactions on Information Theory*, vol. IT-45, no. 5, pp. 2181–2190, Sept. 1999.

[7] K. Engdahl and K. S. Zigangirov, "On the theory of low density convolutional codes I," *Problems of Information Transmission (Problemy Peredachi Informatsii)*, vol. 35, no. 4, pp. 295–310, Oct.-Dec. 1999.

[8] R. M. Tanner, D. Sridhara, A. Sridharan, T. E. Fuja, and D. J. Costello, Jr., "LDPC block and convolutional codes based on circulant matrices," *IEEE Transactions on Information Theory*, vol. IT-50, no. 12, pp. 2966–2984, Dec. 2004.

[9] A. E. Pusane, A. Jiménez Feltström, A. Sridharan, M. Lentmaier, K. S. Zigangirov, and D. J. Costello, Jr., "Implementation aspects of LDPC convolutional codes," *IEEE Transactions on Communications*, vol. 56, no. 7, pp. 1060–1069, July 2008.

[10] A. Sridharan, D.V. Truhachev, M. Lentmaier, D.J. Costello, Jr., and K.Sh. Zigangirov, "Distance bounds for an ensemble of LDPC convolutional codes," vol. 53, no. 12, pp. 4537–4555, Dec. 2007.

[11] D. J. C. MacKay and M. C. Davey, "Evaluation of Gallager codes for short block length and high rate applications," in *Codes, Systems and Graphical Models*, ser. IMA Volumes in Mathematics and its Applications, B. Marcus and J. Rosenthal, Eds. , New York: Springer-Verlag, 2001, vol. 123, ch. 5, pp. 113–130.

[12] A. Sridharan, M. Lentmaier, D. V. Truhachev, D. J. Costello, Jr., and K. S. Zigangirov, "On the minimum distance of low-density parity-check codes with parity check matrices constructed from permutation matrices," *Problems of Information Transmission (Problemy Peredachi Informatsii)*, vol. 41, no. 1, pp. 33–44, Jan.-March, 2005.

[13] T. Richardson and R. Urbanke, "The capacity of low-density parity check codes under message-passing decoding," *IEEE Transactions on Information Theory*, vol. IT-47, no. 2, pp. 599–618, Feb. 2001.

[14] M. Lentmaier, D. V. Truhachev, and K. S. Zigangirov, "Analysis of the asymptotic iterative decoding performance of turbo codes," in *Proc. IEEE International Symposium on Information Theory*, Washington, USA, July 2001, p. 190.

[15] ——, "On the theory of low density convolutional codes II," *Problems of Information Transmission (Problemy Peredachi Informatsii)*, vol. 37, pp. 15–35, Oct.-Dec. 2001.

[16] K. S. Zigangirov, M. Lentmaier, and D. V. Truhachev, "Two approaches to the analysis of low-density parity-check codes," in *Proc. IEEE International Symposium on Information Theory*, Lausanne, Switzerland, July 2002, p. 29.

[17] A. Khandekar and R. J. McEliece, "A lower bound on the iterative decoding threshold of irregular LDPC code ensembles," in *Proceedings CISS 2002*, Princeton, NJ, March 2002.

[18] A. Khandekar, "Graph-based codes and iterative decoding," Ph.D. dissertation, California Institute of Technology Pasadena, CA, 2002.

[19] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann, "Practical erasure resilient codes," in *Proceedings of the 29th annual ACM Symposium on Theory of Computing*, STOC, 1997, pp. 150–159.

[20] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. IT-47, no. 2, pp. 619–637, Feb. 2001.

[21] G. Wiechman and I. Sason, "Parity-check density versus performance of binary linear block codes: New bounds and applications," *IEEE Transactions on Information Theory*, vol. 53, no. 2, pp. 550–579, February 2007.

[22] I. Sason, "On universal properties of capacity-approaching LDPC code ensembles," *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 2956–2990, July 2009.

[23] R. Johannesson and K. S. Zigangirov, *Fundamentals of convolutional coding*. IEEE Press, 1999.

[24] A. Sridharan, "Design and analysis of LDPC convolutional codes," Ph.D. dissertation, University of Notre Dame, Feb. 2005.

[25] J. Thorpe, "Low-density parity-check (LDPC) codes constructed from protographs," in *IPN Progress Report 42-154, JPL*, Aug. 2003.

[26] N. Wiberg, "Codes and decoding on general graphs," Ph.D. dissertation, Linköping University, Sweden, 1996.

[27] A. E. Pusane, M. Lentmaier, K. S. Zigangirov, and D. J. Costello, Jr., "Reduced complexity decoding strategies for LDPC convolutional codes," in *Proc. IEEE International Symposium on Information Theory*, Chicago, USA, July 2004, p. 490.

[28] A. W. Eckford, "Low-density parity-check codes for Gilbert-Elliot and Markov-modulated channels," Ph.D. dissertation, University of Toronto, 2004.

[29] P. Oswald and A. Shokrollahi, "Capacity-achieving sequences for the erasure channel," *IEEE Transactions on Information Theory*, vol. IT-48, no. 12, pp. 3017–3028, December 2002.

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

Michael Lentmaier received the Dipl.-Ing. degree in electrical engineering from University of Ulm, Ulm, Germany in 1998, and the Ph.D. degree in telecommunication theory from Lund University, Lund, Sweden in 2003. He then worked as a Post-Doctoral Research Associate at University of Notre Dame, Indiana, and at University of Ulm, Germany. From 2005 to 2007 he was with the Institute of Communications and Navigation of the German Aerospace Center (DLR) in Oberpfaffenhofen, working on high resolution channel estimation techniques for multipath mitigation in satellite navigation receivers. Since January 2008 he is a senior researcher and lecturer at the Vodafone Chair Mobile Communications Systems at Dresden University of Technology (TU Dresden). His research interests include design and analysis of coding systems, graph based iterative algorithms and Bayesian methods applied to decoding, detection and estimation.

Arvind Sridharan was born in Madras, India. He received the B.Tech degree in electrical engineering from the Indian Institute of Technology, Madras, India in 1999 the M.S. and Ph.D degrees in electrical engineering from the University of Notre Dame, Notre Dame, IN, in 2001 and 2005, respectively. Since February 2005 he has been with the coding and signal processing group at Seagate Technology in Longmont, Colorado. His research interests include the design and analysis of coding systems for recording, iterative decoding and detection algorithms and information theory.

Kamil Sh. Zigangirov was born in the U.S.S.R. in 1938. He received the M.S. degree in 1962 from the Moscow Institute for Physics and Technology,Moscow, U.S.S.R., and the Ph.D. degree in 1966 from the Institute of Radio Engineering and Electronics of the U.S.S.R. Academy of Sciences, Moscow, U.S.S.R. From 1965 to 1991, he held various research positions at the Institute for Problems of Information Transmission of the U.S.S.R. Academy of Sciences, Moscow, first as a Junior Scientist, and later as a Main Scientist. During this period, he visited several universities in the United States, Sweden, Italy, and Switzerland as a Guest Researcher. He organized several symposia on information theory in the U.S.S.R. In 1994, he received the Chair of Telecommunication Theory at Lund University, Lund, Sweden. From 2003 to 2009, he was a Visiting Professor at the University of Notre Dame, Notre Dame, IN, the Dresden Technical University, Dresden, Germany, and at the University of Alberta, Edmonton, AB, Canada. His scientific interests include information theory, coding theory, detection theory, and mathematical statistics. In addition to papers in these areas, he published a book on sequential decoding of convolutional codes (in Russian) in 1974. With R. Johannesson, he coauthored the textbook Fundamentals of Convolutional Coding (Piscataway, NJ: IEEE Press, 1999). His book Theory of CDMA Communication was published by IEEE Press in 2004.

Daniel J. Costello, Jr. was born in Seattle, WA, on August 9, 1942. He received the B.S.E.E. degree from Seattle University, Seattle, WA, in 1964, and the M.S. and Ph.D. degrees in Electrical Engineering from the University of Notre Dame, Notre Dame, IN, in 1966 and 1969, respectively.

Dr. Costello joined the faculty of the Illinois Institute of Technology, Chicago, IL, in 1969 as an Assistant Professor of Electrical Engineering. He was promoted to Associate Professor in 1973, and to Full Professor in 1980. In 1985 he became Professor of Electrical Engineering at the University of Notre Dame, Notre Dame, IN, and from 1989 to 1998 served as Chair of the Department of Electrical Engineering. In 1991, he was selected as one of 100 Seattle University alumni to receive the Centennial Alumni Award in recognition of alumni who have displayed outstanding service to others, exceptional leadership, or uncommon achievement. In 1999, he received a Humboldt Research Prize from the Alexander von Humboldt Foundation in Germany. In 2000, he was named the Leonard Bettex Professor of Electrical Engineering at Notre Dame.

Dr. Costello has been a member of IEEE since 1969 and was elected Fellow in 1985. Since 1983, he has been a member of the Information Theory Society Board of Governors, and in 1986 he served as President of the BOG. He has also served as Associate Editor for Communication Theory for the IEEE Transactions on Communications, Associate Editor for Coding Techniques for the IEEE Transactions on Information Theory, and Co-Chair of the IEEE International Symposia on Information Theory in Kobe, Japan (1988), Ulm, Germany (1997), and Chicago, IL (2004). In 2000, the IEEE Information Theory Society selected him as a recipient of a Third-Millennium Medal. He was co-recipient of the 2009 IEEE Donald G. Fink Prize Paper Award, which recognizes an outstanding survey, review, or tutorial paper in any IEEE publication issued during the previous calendar year.

Dr. Costello's research interests are in the area of digital communications, with special emphasis on error control coding and coded modulation. He has numerous technical publications in his field, and in 1983 he co-authored a textbook entitled "Error Control Coding: Fundamentals and Applications", the 2nd edition of which was published in 2004.