LDPC Codes and Convolutional Codes with Equal Structural Delay: A Comparison

Thorsten Hehn and Johannes B. Huber
Institute for Information Transmission (LIT)
University of Erlangen-Nuremberg, Germany
{hehn, huber}@LNT.de

***Abstract*—We compare convolutional codes and LDPC codes with respect to their decoding performance and their structural delay, which is the inevitable delay solely depending on the structural properties of the coding scheme. Besides the decoding performance, the data delay caused by the channel code is of great importance as this is a crucial factor for many applications. Convolutional codes are known to show a good performance while imposing only a very low latency on the data. LDPC codes yield superior decoding performance but impose a larger delay due to the block structure. The results obtained by comparison will also be related to theoretical limits obtained from random coding and the sphere packing bound. It will be shown that convolutional codes are still the first choice for applications for which a very low data delay is required and the bit error rate is the considered performance criterion. However, if one focuses on a low signal-to-noise ratio or if the obtained frame error rate is the base for comparison, LDPC codes compare favorably.**

Keywords: Low-delay Data Transmission, Convolutional Codes, Low-Density Parity-Check Codes, Progressive Edge Growth, Belief Propagation.

I. Introduction

Almost all communications systems require low data latency as two or more users are interacting with each other. Speech transmission over the telephone or interactive services on the Internet are only two examples. Error correcting codes [1] have to guarantee low error rates at reasonable signal-to-noise ratios but they also cause a delay of information symbols due to encoding, decoding, and due to the fact that for decoding of one source symbol, a multitude of received symbols needs to considered at the receiver. In the last decade, research in the area of channel coding has been strongly focused on decoding and design of LDPC codes [2], where especially the design of long codes has drawn a lot of attention. LDPC codes are already employed in communication standards like DVB-S2 [3] or IEEE 802.3an [4]. The codes used in these standards are either long ($n = 64800$ and $n = 16200$, DVB-S2) or of moderate length ($n = 2048$, IEEE 802.3an). Here, n denotes the length of a block code. Also, some short codes of length $n \leq 1000$ have been included in standards ($576 \leq n \leq 2304$, IEEE 802.16e [5]).

Convolutional codes are still very popular and also widely used in communications standards. For example, convolutional codes of rate $1/2$ with 256 states are intended for channel coding in the UMTS standard [6]. Besides *turbo coding schemes* [7] with rather short interleavers, these codes are one option for reliable transmission over several channels defined in the UMTS standard, including the "speech channel". Reasons for this choice surely include the short delay imposed by this type of codes.

We compare convolutional codes and LDPC codes with respect to their decoding performance and the imposed delay[1]. For the LDPC codes we consider codes constructed by the *progressive edge growth* (PEG) algorithm [8]. Comparison of these codes to results of recent and current research on channel coding [2], [9] leads to the conclusion that PEG codes are an excellent choice for block codes of short to moderate length. The decoding performance achieved by PEG codes also compares favorably to the performance of standardized codes of equal length, e.g. the channel code of rate $R = 1/2$ and length $n = 576$ which is applied in the IEEE 802.16e standard, cf. [10]. The convolutional codes [11] investigated in this work are chosen in such a way that they provide maximum free distance [12]. In order to be able to compare the delays of convolutional codes and LDPC codes, we require LDPC codes of short code lengths, and set $n = \nu \cdot 50$, $\nu = 1, 2, 4, 6, \ldots, 20$. This paper shows how the tradeoff between low delay and good error-correction performance can be adjusted for each scheme and points out favorable schemes and codes for given data delays. This work differs substantially from the results presented in [13] and [14] as we consider the structural delay as a performance criterion and investigate LDPC codes.

The paper is organized as follows. Section II introduces the used channel model, Section III defines the structural delay and determines its value for the considered coding schemes. Section IV specifies the investigated codes, Section V discusses theoretical limits of the decoding performance when the code length is specified. Section VI presents the results of this comparison.

II. Channel model

In this section the underlying channel model is described and the transmitter and receiver are investigated. Time t is expressed as multiples of T_b (time interval per bit) or T_s (time interval per symbol), respectively, $t = \tau_\mathrm{b} \cdot T_\mathrm{b} = \tau_\mathrm{s} \cdot T_\mathrm{s}$, $T_\mathrm{b} > 0$, $T_\mathrm{s} > 0$, i.e. $\tau_\mathrm{b} \in \mathbb{N}$, $\tau_\mathrm{s} \in \mathbb{N}$ are time indices without dimension. Assume the presence of a source which generates independent and identically-distributed binary source symbols

[1]Turbo codes also obtain a very good decoding performance but use rather large interleaver lengths, resulting in a comparably high structural delay. For this reason, Turbo codes are not included in this comparison.

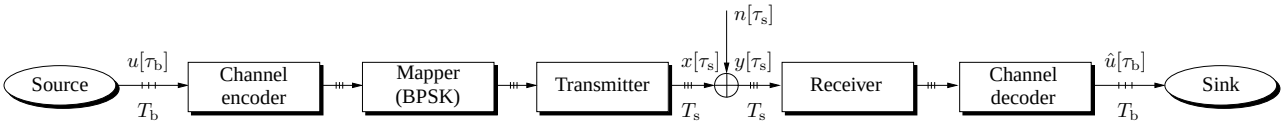


Fig. 1. Investigated transmission scheme

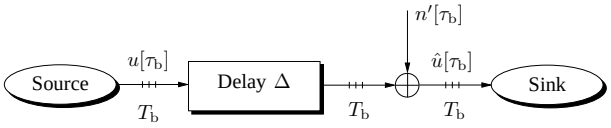


Fig. 2. Schematic representation of the transmission channel by a unit imposing delay on the information symbols and an equivalent distortion $n'[\tau_\mathrm{b}]$.

with a data rate of $1/T_\mathrm{b}$ $\frac{\mathrm{bit}}{\mathrm{s}}$ and let $u[\tau_\mathrm{b}]$ denote the source symbol observed at $t = \tau_\mathrm{b} \cdot T_\mathrm{b}$. A channel encoder accepts a binary information symbol at a clock rate of $1/T_\mathrm{b}$ and presents encoded data symbols with a rate of $1/T_\mathrm{s}$ $\frac{\mathrm{symbol}}{\mathrm{s}}$ (baud) at its output. We choose $T_\mathrm{s} = T_\mathrm{b} \cdot R$ to guarantee a continuous data flow through the encoder. Here, R is the rate of the channel coding system with dimension $\frac{\mathrm{bit}}{\mathrm{symbol}}$. Note that for simplicity we restrict our investigations to mapping of the binary code symbols to antipodal channel symbols (binary phase-shift keying, BPSK). Accordingly, the rate of the modulation scheme equals 1. Figure 1 visualizes the coherence between the operating frequencies T_b and T_s.

BPSK symbols are transmitted over a discrete-time AWGN channel at a rate of $1/T_\mathrm{s}$ baud. Note that the generalization of the presented investigations to other modulation formats is straightforward. The value $x[\tau_\mathrm{s}]$ denotes the transmitted symbol at $t = \tau_\mathrm{s} \cdot T_\mathrm{s}$ (point in time). Signaling without intersymbol interference (ISI-free transmission) [15] over the AWGN channel is assumed. The signal-to-noise ratio (SNR) is measured as $10\log_{10}(E_\mathrm{b}/N_0)$ where we follow the standard notation. Hence, E_b denotes the average energy required to transmit one bit of information and N_0 denotes the one-sided power density of the continuous-time white Gaussian noise. The receiver detects and processes data at a rate of $1/T_\mathrm{s}$ baud. A channel decoder accepts these values and determines an estimate $\hat{u}[\tau_\mathrm{b}]$ of the encoded data. These values are passed to the information sink with a rate of $1/T_\mathrm{b}$ $\frac{\mathrm{bit}}{\mathrm{s}}$. Figure 1 and Figure 2 depict the used channel model as well as a schematic representation using a delay unit and equivalent noise, respectively.

III. Structural delay

When using channel codes, the constraints imposed on the symbols within a codeword allow for error correction but also delay each information symbol by a certain number of time-steps. In the following, we discuss the three most important types of data transmission and determine the delay due to the structure of the communications system. Additional delay caused by limited performance of processing units is not considered in this paper. Thus, data processors with infinite clock rates are assumed. We assume that symbol $x[\tau_\mathrm{s}]$ transmitted at sampling instant τ_s is present at the receiver input within the same sampling instant. However, we respect the fact that the transmitter has to transmit one symbol after another at the given clock rate $1/T_\mathrm{s}$.

Definition 3.1: Consider the data processing units in Figure 1. Assume that any symbol (information symbol or parity symbol) can be transmitted from the output of a data processing unit to the input of the subsequent one within the same sampling instant τ_b or τ_s, respectively. Note that this is in particular assumed for the processing units transmitter and receiver. Assume also that all mathematical operations can be performed with unlimited processing speed. The number Δ of time-steps of duration T_b between the appearance of a symbol at the source and its delivery to the sink is the structural delay Δ of the considered coding scheme.

As stated above, we restrict our attention to the structural delay which is a lower bound on the actual delay. This is a consequence of the fact that the latencies due to processing and wave propagation are not considered. The delay due to wave propagation depends on the transmission setup and is not a parameter of the communications system. According to our experience, processing delays for convolutional codes with up to 1024 states, decoded by the Viterbi algorithm, and block codes of length $n \leq 1000$, decoded by the belief-propagation (BP) algorithm after 200 iterations, are of comparable scale. Simulations on standard computers show that up to data rates of 1 Mbit/s the structural delay and processing delay lie in the same range. From these observations we conclude that structural delay is already an important parameter and the progress in microelectronics will further emphasize its meaning. Moreover, structural delay is a feature of the coding scheme itself, regardless of current and future ways of implementation. Assuming an infinite processor speed, as done in this work, allows for a basic comparison unaffected by the current state of progress in the field of microelectronics. Detailed results regarding the coherence between structural and processing delay will be presented in a subsequent paper, currently in preparation.

In this section, we first discuss the structural delay of block codes, denoted by Δ_bc and then the structural delay of convolutional codes, denoted by Δ_cc.

A. Structural delay of block codes

We discuss the structural delay of $[n, k, d]$ block codes of rate R. One major drawback of block codes is the fact that the decoder has to receive the whole codeword before it can initiate the decoding process.

To provide guidelines for a multitude of practical implementations, we investigate three transmission scenarios.

1) Serial processing and non-systematic encoding: In the case of non-systematic encoding and serial processing of the

input data, the transmitter cannot start the transmission before the encoder has access to all $k = R \cdot n$ information symbols which are accepted at a clock rate of $1/T_\mathrm{b}$. On the receiver side we assume that the estimates of the information symbols are generated serially in the same order as they were presented to the channel encoder.

In order to emphasize basic coherences, the transmission of one single codeword is analyzed. An information symbol presented to the encoder at sampling instant τ_b^*, $\tau_\mathrm{b}^* \in \{1, \ldots, k\}$, is not processed before $k - 1 - (\tau_\mathrm{b}^* - 1)$ further information symbols are readily available. Its corresponding estimate encounters a delay of $\tau_\mathrm{b}^* - 1$ time-steps of duration T_b at the decoder as $\tau_\mathrm{b}^* - 1$ symbols are depleted before the current symbol of interest. This sums to a delay of $k - 1 = Rn - 1$ time-steps of duration T_b. Furthermore each symbol is delayed for n time-steps of duration T_s ($n \cdot T_\mathrm{s} = n \cdot R \cdot T_\mathrm{b}$) due to transmission of the codeword at a clock rate of $1/T_\mathrm{s}$. This adds to a total structural delay of

$$\Delta_{\mathrm{bc,sp}} = 2Rn - 1$$

time-steps of duration T_b for *serial processing and non-systematic encoding* of block codes of length n.

2) Serial processing and systematic encoding: If one applies systematic encoding, the delay can be lowered as the transmission can start before the whole source-word, consisting of k symbols, is available at the channel encoder. In this case we assume that the generator matrix consists of an identity matrix within the first k columns. This means that in each block all information symbols are transmitted at the beginning. As soon as this is accomplished, all parity symbols are assumed to be readily available for transmission. We can start the transmission of the information symbols when $n \cdot R \cdot \varepsilon$ source symbols are present. Here, ε is a factor depending on the rate of the channel code. We are interested in a continuous transmission. To this end, let us consider the transmission of the ε-th part of a block of k source symbols. This incident requires $n \cdot R \cdot \varepsilon$ time-steps of length T_s. To avoid discontinuation, the ε-th share of the remaining source symbols, i.e. $n \cdot R \cdot (1 - \varepsilon) \cdot \varepsilon$ source symbols have to be generated by the source during transmission of $\varepsilon \cdot n \cdot R$ source symbols. As the source requires a time-step of length T_b to generate a source symbol,

$$n \cdot R \cdot (1 - \varepsilon) \cdot \varepsilon \cdot T_\mathrm{b} = n \cdot R \cdot \varepsilon \cdot T_\mathrm{s}$$

has to hold. Thus,

$$n \cdot R \cdot (1 - \varepsilon) \cdot \varepsilon \cdot T_\mathrm{b} = n \cdot R \cdot \varepsilon \cdot R \cdot T_\mathrm{b}$$

needs to be true. The non-trivial solution of this equation reads $\varepsilon = 1 - R$. The structural delay of all these symbols can be calculated the following way. Let τ_b^*, $\tau_\mathrm{b}^* \in \{1, \ldots, k\}$, indicate the sampling instant in which a source symbol of interest has been generated. Using a block code of rate R, the whole codeword is available at the receiver at $t = n \cdot R \cdot (1 - R) \cdot T_\mathrm{b} + n \cdot T_\mathrm{s} = n \cdot R \cdot (1-R) \cdot T_\mathrm{b} + n \cdot R \cdot T_\mathrm{b} = n \cdot R \cdot (2-R) \cdot T_\mathrm{b}$. To this point, the delay of each source symbol is dependent on its position and adds up to $n \cdot R \cdot (2 - R) - \tau_\mathrm{b}^*$ time-steps of duration T_b. Now consider that the delay for depleting the decoder imposes another delay of $(\tau_\mathrm{b}^* - 1)$ time-steps of duration T_b. The overall structural delay for *serial processing and systematic encoding* does not depend on τ_b^* and is given by

$$\Delta_{\mathrm{bc,ssp}} = ((2 - R) \cdot R \cdot n - 1)$$

time-steps of duration T_b.

3) Block (Parallel) Processing: Let us consider the special case of block processing. Here, we assume that the whole source-word is available at the channel encoder within one sampling instant, i.e. parallel processing. We also assume that these blocks can be passed to the sink within one sampling instant. This assumption is relevant if the data is taken from a very fast memory and passed to such a device at the receiver side. Note that one does not have to distinguish between non-systematic and systematic encoding in this case and obtains a delay of n time-steps of duration T_s. Block processing can be modeled by letting $T_\mathrm{b} \to 0$ while keeping T_s constant. In this case the delay is denoted as multiples of $\frac{1}{R} \cdot T_\mathrm{s}$. One obtains

$$\Delta_{\mathrm{bc,bp}} = R \cdot (n - 1)$$

time-steps of duration $\frac{1}{R} \cdot T_\mathrm{s}$. This value represents the lowest possible delay for block codes.

B. Structural delay of convolutional codes

Encoding a convolutional code is usually realized by a *tapped delay line* (TDL) consisting of delay units and binary addition units. Let the number of delay units be given by $M = \mathrm{ld}(|\mathcal{S}|)$, where $\mathcal{S}$ denotes the set of possible memory configurations (states) of the encoder, and the operator $|\cdot|$ determines the cardinality of a set. When starting the encoding process, the encoder is set to the zero state. If we do not take the calculation time of the involved processors into account, the encoded symbols are available at the encoder output as soon as the first source bit is passed to the encoder. Note that this is true for both recursive and non-recursive encoder implementations for codes of rate $R = 1/\mu$, $\mu \in \mathbb{N}$. Encoding as well as mapping and transmission can be accomplished within the same sampling instant τ_b as $T_\mathrm{s} = T_\mathrm{b} \cdot R$ is chosen. Recapitulating, there is no structural delay when encoding a convolutional code.

For decoding convolutional codes, either a maximum-likelihood symbol-by-symbol estimator (MLSSE) or a maximum-likelihood sequence estimator (MLSE) is deployed in the usual case. The MLSSE decoder is most efficiently realized by the BCJR algorithm [16] which uses a recursive forward and backward calculation of state probabilities to find optimum a-posteriori probabilities for the transmitted symbols.

The BCJR decoder generates estimates for the information symbols only, thus its output works at a data rate of $1/T_\mathrm{b}$ $\frac{\mathrm{bit}}{\mathrm{s}}$. It stores state probabilities in sampling instant τ_b determined by recursive forward calculation as $\boldsymbol{\alpha}[\nu]$, $\nu = 0, \ldots, k$, and those determined by backward calculation as $\boldsymbol{\beta}[\nu]$. Note that the decoder therefore requires to be informed about the state probabilities before the encoding of the sequence has started and after this task has been completed, i.e. $\boldsymbol{\alpha}[0]$ and $\boldsymbol{\beta}[k]$, as well as the complete received sequence from $\tau_\mathrm{b} = 1$ to $\tau_\mathrm{b} = k$,

where k again denotes the length of the transmitted information sequence. Knowledge of $\boldsymbol{\alpha}[0]$ and $\boldsymbol{\beta}[k]$ are ensured as it is accepted that the encoder starts and terminates encoding in the zero-state. The latter is accomplished by appending tail bits to the sequence.

When using the BCJR decoder as described above, its structural delay equals the length of the transmitted information sequence, k, due to backward calculation of $\boldsymbol{\beta}$. In order to achieve the lowest structural delay possible, we consider from now on a decoding scheme which introduces a block structure in the decoding process to estimate *one* symbol at a time instead. A sliding window approach is used to single out a sub-block of the data received during the whole transmission, cf. [17]. By this means the decoding becomes suboptimal but simulation results show that the performance is still very close to the optimal case [18]. When using this method, the vector $\boldsymbol{\beta}$ needs to be estimated at the current sampling instant. For this reason we allow the decoder to access information symbols and the corresponding parity information within a window of length $w_{\boldsymbol{\beta}}$ in advance. In order to estimate the source symbol transmitted at $t = \tau_\mathrm{b} \cdot T_\mathrm{b}$, it is sufficient to know $y[1]$ to $y[\tau_\mathrm{b} + 1 + w_{\boldsymbol{\beta}} - 1]$ which allows for accurate estimation of $\boldsymbol{\beta}[\tau_\mathrm{b}]$. As all states are equally probable, the entries in $\boldsymbol{\beta}[\tau_\mathrm{b} + w_{\boldsymbol{\beta}}]$ are chosen to be equal what allows for estimation of $\boldsymbol{\beta}[\tau_\mathrm{b}]$ to $\boldsymbol{\beta}[\tau_\mathrm{b} + w_{\boldsymbol{\beta}} - 1]$ by backward recursion. Using the estimation of $\boldsymbol{\beta}[\tau_\mathrm{b}]$ it is possible to estimate $\hat{u}[\tau_\mathrm{b}]$. This method is known as the sliding-window BCJR [17, p. 133]. Its structural delay is significantly lower than the delay of the standard method and amounts to $w_{\boldsymbol{\beta}}$. Bear in mind that $w_{\boldsymbol{\beta}}$ is a user-defined parameter. Also note that the delay is constant for all transmitted symbols and further observe that no case differentiation between serial processing and block processing has to be accomplished. Considering the encoder and the decoder of this transmission scheme, the structural delay for convolutional codes adds up to $\Delta_{\mathrm{cc,\,BCJR}} = w_{\boldsymbol{\beta}}$.

Another option to decode a convolutional code is the Viterbi decoder [19]. In this case, no backward recursion is required but the structural delay of this decoder also equals the length of the transmitted sequence, k, which is impractical in many situations. Again, a window-based algorithm can be applied which allows to select the optimal path through the trellis up to time-step τ_b with high probability when the decoder has conducted a path estimation up time-step $\tau_\mathrm{b} + w_\mathrm{p}$ [18]. Similar to w_β in the BCJR case, w_p denotes the number of time-steps for which the decoder needs to have access to the corresponding received values in advance to provide an accurate decision on the current symbol. The structural delay when using a sliding window Viterbi decoder results to $\Delta_{\mathrm{cc,\,Viterbi}} = w_\mathrm{p}$. Both $w_{\boldsymbol{\beta}}$ and w_p are usually chosen as multiples of the constraint length for non-recursive encoders [12], i.e. multiples of $\log_2(|\mathcal{S}|)+1$. For the sake of simplicity, we will henceforth denote the delay imposed by a convolutional code by Δ_cc, regardless of the decoder.

IV. Investigated Codes

For the sake of simplicity and in order to present exemplary results of high relevance we restrict our comparison to codes of rate $1/2$. We compare convolutional codes and PEG optimized LDPC codes in terms of their structural delay and the required SNR given as $10\log_{10}(E_\mathrm{b}/N_0)$ in order to obtain a certain reliability. Both LDPC codes and convolutional codes are widely used in communications standards and require to perform well at low decoding delays.

A. Convolutional codes

We choose maximum free distance convolutional codes of rate $1/2$ [20], [12]. Table I shows the generator polynomials $g_{0,\mathrm{oct}}$, $g_{1,\mathrm{oct}}$, of these codes in standard octal representation [21] along with the number of states $|\mathcal{S}|$ and the free distance d_free [12]. Codes with up to 2^{10} states are taken into account.

TABLE I
Generator polynomials, the number of states $|\mathcal{S}|$ and the free distance d_free for the convolutional codes of rate $1/2$ used in the comparison.

$g_{0,\mathrm{oct}}$	$g_{1,\mathrm{oct}}$	Number of states $\lvert\mathcal{S}\rvert$	d_free
2	3	2^1	3
5	7	2^2	5
13	17	2^3	6
31	27	2^4	7
65	57	2^5	8
155	117	2^6	10
345	237	2^7	10
435	657	2^8	12
1671	1233	2^9	12
2731	2157	2^{10}	14

B. PEG-optimized LDPC codes

We study PEG-optimized block codes of rate $1/2$ and length $n = \nu \cdot 50$, $\nu = 1, 2, 4, 6, \ldots, 20$. As stated in Section I, these codes represent an excellent choice for block codes within this length. All codes are obtained from the same choice of variable-node and check-node degree distribution pair. This pair compares favorably to other degree distributions in terms of decoding performance when constructing codes of lengths up to $n = 1000$. The normalized degree distributions of the variable nodes and check nodes from node perspective are taken from [22] and given by

$$\begin{aligned} L(x) &\approx 0.504 \cdot x^2 + 0.296 \cdot x^3 + 0.057 \cdot x^5 \\ &\quad + 0.036 \cdot x^6 + 0.005 \cdot x^7 + 0.029 \cdot x^9 \\ &\quad + 0.065 \cdot x^{11} + 0.008 \cdot x^{12} \\ R(x) &= 1 \cdot x^7. \end{aligned} \tag{1}$$

This distribution was optimized for the AWGN channel by Density Evolution. Note that the chosen degree distribution is check-regular, i.e. $R(x)$ is a polynomial with only one degree. It is advantageous to construct PEG codes from a check-regular distribution as the check-node degree distribution is not passed to the PEG algorithm but evolves during the construction process. It can generally be stated that PEG-optimized LDPC codes exhibit a concentrated (almost regular) check-node degree distribution [8].

In the following, we investigate the shortest cycle length for each variable node of the PEG-optimized LDPC codes. A cycle is closed-loop path in the Tanner graph [23] of the code. The variable nodes of the codes with $n \leq 200$ show shortest cycles of length 4 and 6, whereas the variable nodes of the codes with $300 \leq n \leq 1000$ have shortest cycles of length 6 and 8. This is due to the construction method of PEG codes and the fact that all codes are based on the same degree distribution. The minimum distance of this family of codes was estimated by the methods in [24]. It was found that it has a bad distance [2, Ch. 13].

V. Theoretical limits

To assess our results more generally, we discuss theoretical bounds on the power efficiency. The *Gallager bound* [25] considers an ensemble of block codes of rate R and length n and provides a tight upper bound on the expected frame error rate (FER), where the expectation is taken over all codes in the ensemble. For this reason it is also known as the random coding bound. It flags aspired values on the power efficiency, but, as it is based on random coding, realizations outperforming this bound may exist. Another bound considered in this work is the *sphere packing bound* [26] which marks a lower bound on the FER and is thus a non-existence bound on the power efficiency.

We introduce a design parameter $0 \leq \rho \leq 1$. Upper bounds on the ensemble-average FER are given by

$$\mathrm{E}\{\mathrm{FER}\} \leq \mathrm{e}^{-n \cdot (E_0(\rho, \Pr(\boldsymbol{X}), 10\log_{10}(E_\mathrm{b}/N_0)) - \rho \cdot R)}, \quad (2)$$

where $\Pr(\boldsymbol{X})$ denotes the vector of probabilities of possible channel inputs. Maximization over the design parameter ρ and the vector $\Pr(\boldsymbol{X})$ leads to

$$\mathrm{E}\{\mathrm{FER}\} \leq \exp(-n \cdot E_r(R)), \quad (3)$$

where the random coding exponent

$$E_\mathrm{r}(R) = \max_{0 \leq \rho \leq 1} \max_{\Pr(\boldsymbol{X})} \left(E_0(\rho, \Pr(\boldsymbol{X}), 10\log_{10}(E_\mathrm{b}/N_0)) - \rho R\right),$$

and the Gallager function

$$E_0(\rho, \Pr(\boldsymbol{X}), 10\log_{10}(E_\mathrm{b}/N_0)) = -\mathrm{ld}\left[\int_{-\infty}^{\infty} \left(\sum_{i=1}^{M_x} [f_y(y \mid x_i)]^{\frac{1}{1+\rho}} \Pr(x_i)\right)^{1+\rho} \mathrm{d}y\right],$$

are defined as usual. Furthermore $f_y(y \mid x_i)$, $i = 1, \ldots, M_x$, denotes the probability density function of the channel transition and M_x is the cardinality of the input alphabet. Note that $f_y(y \mid x_i)$, $i = 1, \ldots, M_x$, depends on $10\log_{10}(E_\mathrm{b}/N_0)$.

The Gallager bound is an existence bound stating that it is possible to transmit information using block codes of finite length n, given $E_\mathrm{r}(R) > 0$ and that a FER of up to $\exp(-n \cdot E_\mathrm{r}(R))$ is accepted. To use the random coding bound as a comparative value in the following figures, we use Equation (3) to determine the lowest SNR that allows to obtain the desired FER when a code length n is specified. To this end we have to optimize the parameters ρ and $10\log_{10}(E_\mathrm{b}/N_0)$ at the same time. This is done by an iterative computer search [27]. We also focus our attention on the case where a given bit error rate (BER) needs to be met. As the considered bound only provides values for the FER, we assume that random-codes produce at most d bit errors in a wrong frame, where d denotes an estimate on the minimum distance of a typical random code from the considered ensemble. In order to find d, we use the Gilbert-Varshamov-bound [1, p. 34],

$$\sum_{i=0}^{d-2} \binom{n-1}{i} < 2^{n-k}. \quad (4)$$

For the considered values of length n and rate R we find the largest d such that Equation (4) is still satisfied. Using the estimated "average" minimum distance, we state $\mathrm{BER} \approx \frac{d}{n} \cdot \mathrm{FER}$.

Let us now present the *sphere packing bound* for the code lengths and rates investigated. Again, we assume that a given FER needs to be met. Similar technical modifications as for the Gallager bound allow to present this bound for given BER values. This bound shows the power efficiency that an optimal coding scheme would require to meet the desired error rate and thus marks a non-existence bound for block codes. Unlike the channel capacity, this bound depends on the code length and thus represents a tighter non-existence bound when n, R, and FER (or BER, respectively) are specified. To obtain the results presented in this paper we closely followed [28] and [29]. The sphere packing bound is given by

$$\mathrm{FER} > P_\mathrm{SPB}(n, \theta, A)$$

with

$$P_\mathrm{SPB}(n, \theta, A) = Q(\sqrt{n}A) + \frac{(n-1)\mathrm{e}^{-\frac{nA^2}{2}}}{\sqrt{2\pi}} \int_{\theta}^{\pi/2} (\sin(\phi))^{n-2} f_n(\sqrt{n}A\cos\phi)\,\mathrm{d}\phi,$$

where $A = \sqrt{\frac{2E_\mathrm{s}}{N_0}}$,

$$f_n(x) = \frac{1}{2^{\frac{n-1}{2}}\Gamma(\frac{n+1}{2})} \int_0^{\infty} z^{n-1}\mathrm{e}^{-z^2/2+zx}\,\mathrm{d}z, x \in \mathbb{R}, n \in \mathbb{N}.$$

Here, $Q(\cdot)$ denotes the complementary Gaussian error integral, and $\Gamma(\cdot)$ denotes the Gamma function. In order to determine θ we choose an approximation by finding a θ^* that satisfies

$$\frac{\Gamma\left(\frac{n}{2}\right)(\sin(\theta^*))^{n-1}}{2\Gamma\left(\frac{n+1}{2}\right)\sqrt{\pi}\cos(\theta^*)}\left(1 - \frac{\tan^2(\theta^*)}{n}\right) = 2^{-n \cdot R}.$$

For details on this bound we refer the reader to [26].

VI. Results

This section compares results for PEG-optimized LDPC codes and convolutional codes in terms of decoding performance and imposed delay.

A. Convolutional codes

In order to estimate the performance of convolutional codes over a wide range of delay values, we choose the window lengths $w_{\boldsymbol{\beta}}$ and w_{p} to multiples of the constraint length, $f \cdot (\log_2(|\mathcal{S}|)+1)$, $f \in \{1,\ldots,5\}$ for both the BCJR decoder and the Viterbi decoder. This results in a structural delay of $\Delta_{\mathrm{cc}} = f \cdot (\log_2(|\mathcal{S}|)+1)$, $f \in \{1,\ldots,5\}$. It is known that the decoding performance cannot be improved significantly by choosing $f > 5$ [18]. Bear in mind that all positive integer values would be possible choices for Δ_{cc}, regardless of $|\mathcal{S}|$. However, most of these choices do not correspond to integer values of the parameter f.

When measuring the BER and FER, we only take information symbols into consideration. For the FER we denote the length of the frame on which the FER is measured by L_{F}. We measure the FER-performance for various lengths of L_{F} by recording the error gap distribution $g[\kappa]$, $\kappa = 1,\ldots,k$ in simulations. This distribution specifies the probabilities[2] that there exist $\kappa-1$ correctly decoded symbols between two successive errors, what allows for calculation of the FER on a frame of length L_{F} by [30], [31]

$$\mathrm{FER} = \mathrm{BER} \cdot \sum_{\kappa=1}^{L_{\mathrm{F}}} \left(1 - \sum_{\kappa'=1}^{\kappa-1} g[\kappa']\right).$$

The BERs obtained by the convolutional codes are given in Figure 3. To illustrate representative examples, the factors $f = 1$ and $f = 5$ are chosen and the number of states was set to $|\mathcal{S}| = 2^1, 2^2, \ldots, 2^{10}$. The results were obtained by the sliding-windows BCJR approach, which performs slightly better than the Viterbi decoder. Results for the latter decoder are not shown due to space limitations.

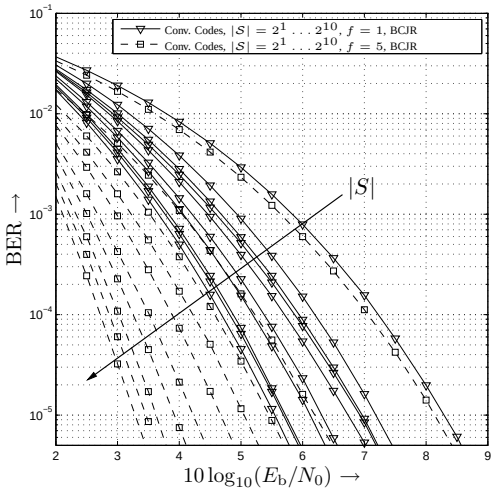


Fig. 3. BER for rate $1/2$ convolutional codes with $|\mathcal{S}| = 2^1$ to $|\mathcal{S}| = 2^{10}$ states and $f = 1$ and $f = 5$, respectively. Results are obtained with a sliding-window BCJR decoder.

It is obvious that both f and $|\mathcal{S}|$ have a strong impact on the decoding performance. As we aim at good performance results when the delay is low, we determine pairs of f and $|\mathcal{S}|$ which compare advantageous to other combinations. Figure 4(a) and Figure 4(b) depict the required signal-to-noise ratio ($10\log_{10}(E_{\mathrm{b}}/N_0)$) to obtain $\mathrm{BER} = 10^{-5}$ and $\mathrm{FER} = 10^{-3}$, respectively, for all considered convolutional codes and all considered choices of f when using the BCJR decoder. Applying a Viterbi decoder instead of the BCJR decoder leads to a similar result but slightly higher required signal-to-noise ratio. These results are not shown due to space limitations. For convolutional codes, a wrong decision at the receiver may be caused by a very noisy reception of the current symbol or the Δ_{cc} symbols used for the state-estimation or path-estimation. Hence, a protocol of higher order that initiates the retransmission of the erroneous part of the stream (detected by e.g. an outer coding scheme) needs to demand the retransmission of Δ_{cc} symbols besides the current one, what motivates us to set $L_{\mathrm{F}} = \Delta_{\mathrm{cc}} + 1$ with $\Delta_{\mathrm{cc}} = f \cdot (\log_2(|\mathcal{S}|)+1)$.

Figure 4 provides the insight that there exist specific choices for f and $|\mathcal{S}|$, which are most advantageous to meet a given error rate at a tolerated delay. That means that these pairs of f and $|\mathcal{S}|$ meet the given constraints at an SNR that is lower than the SNR required by all other combinations. As stated, the Viterbi decoder leads to slightly poorer performance results. Note that the BCJR decoder should in particular be preferred when a given FER needs to be underrun despite the fact that the Viterbi decoder is optimal for estimating a sequence. This is due to the fact that only sub-blocks of the sequence which are short compared to the whole sequence are considered for decoding.

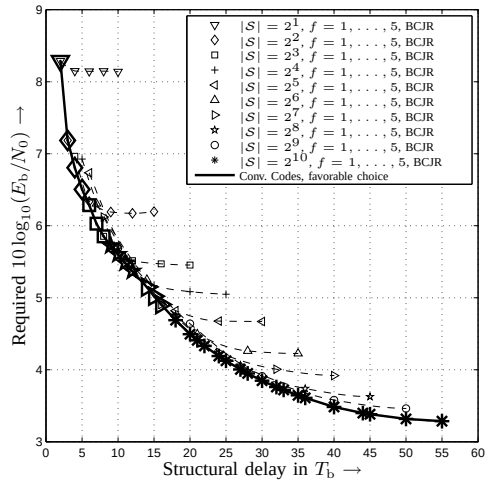
Example 6.1: Assume that one intends to meet $\mathrm{BER} = 10^{-5}$ and tolerates a delay of 8 time-steps T_{b}. At the receiver, a sliding-window BCJR decoder is applied. Figure 4(a) indicates that this is possible with a convolutional code with $|\mathcal{S}| = 2^1$ states and a choice of $f = 4$. In this case, an SNR of at least $10\log_{10}(E_{\mathrm{b}}/N_0) = 8.15$ dB would be required. Another choice to obtain $\Delta_{\mathrm{cc}} = 8$ would be a 2^4-state code. Here, the delay is not a multiple of the constraint length; to be precise $f = 8/5$ has to be chosen to obtain $w_\beta = 8$ for this code. The required SNR is significantly lower, $10\log_{10}(E_{\mathrm{b}}/N_0) = 6.10$ dB, as can be observed in Figure 4(a). Simulations for non-integer values of f confirm this value. The bold curve in Figure 4(a) shows that the optimal choice in this setting is a code with $|\mathcal{S}| = 2^3$ states, with $f = 2$, and a resulting SNR of $10\log_{10}(E_{\mathrm{b}}/N_0) = 5.85$ dB.

Note that codes with a high number of states do not necessarily outperform codes with a lower number of states in this comparison. $\Delta_{\mathrm{cc}} = 8$ and $\mathrm{BER} = 10^{-5}$ can also be obtained using a code with $|\mathcal{S}| = 2^7$ states and $f = 1$. However, a higher SNR of $10\log_{10}(E_{\mathrm{b}}/N_0) = 6.12$ dB is required in that case.

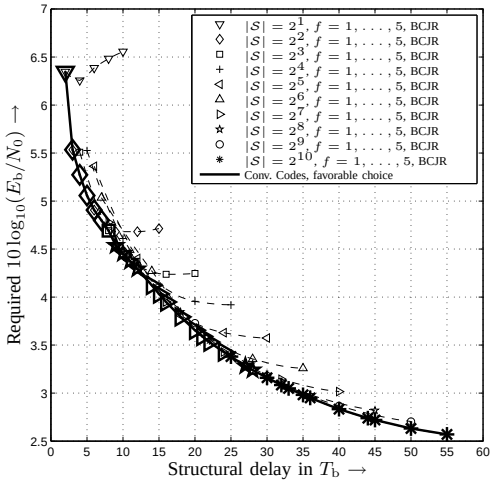
B. Block codes

We discuss the performance of block codes, to be precise LDPC codes, with respect to a given structural delay. Figure 5 shows performance results for PEG-optimized LDPC codes which realize the degree distribution $L(x)$ and closely approach $R(x)$, as well as theoretical bounds for all codes

[2] In this context, probabilities are approximated by relative frequencies obtained by simulation.



(a) $\mathrm{BER} = 10^{-5}$



(b) $\mathrm{FER} = 10^{-3}$

Fig. 4. The required signal-to-noise ratio ($10\log_{10}(E_\mathrm{b}/N_0)$) to obtain $\mathrm{BER} = 10^{-5}$ or $\mathrm{FER} = 10^{-3}$, respectively, with different **convolutional codes** and different factors f is plotted over the structural delay. In all cases the **BCJR** decoder was applied.

of rate $1/2$ and codes realizing the given degree distribution. All codes are decoded by a BP decoder performing at most 200 iterations on parity-check matrices of full rank and size $(n-k) \times n$. The number of iterations is chosen such that a further increase of the number of iterations does not improve the decoding performance of the considered codes significantly. Again, only information symbols are considered for calculating the BER and the FER. The depicted BER corresponds to the case of systematic encoding. For non-systematic encoding, a slightly worse BER but equal FER are obtained. This is due to the post-processing step using a dense generator matrix. In this case, the FER is measured on the estimated source-words of length $L_\mathrm{F} = R \cdot n$. As the construction of a PEG code includes random decisions, it is worth pointing out that the performance of the PEG codes does not depend on the construction process within the considered error rates. To confirm this, ensembles of codes of length $n = \nu \cdot 50$ with $\nu = 1, 2, 4, 6, \cdots, 20$ were constructed with the PEG algorithm. All codes follow the degree distribution $L(x)$, but different seeds were used for the random number generators involved in the construction process. For each ensemble, simulation results show a strictly concentrated performance within the error rates of interest. Thus, the distinct choices of the random PEG process made during the construction phase of the code show only very little impact on the performance in all situations investigated within this work.

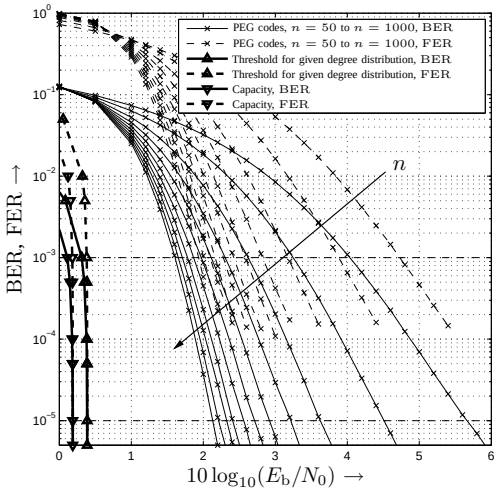
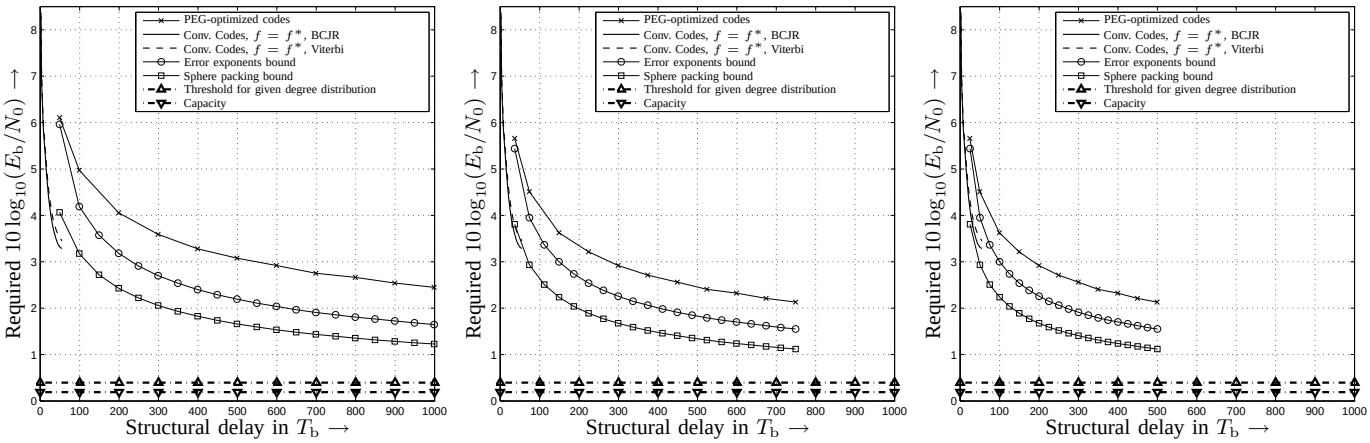


Fig. 5. Simulation results for the considered PEG-optimized LDPC codes using a BP decoder and full-rank matrices of size $(n-k) \times n$. At most 200 iterations per codeword were performed and code lengths $n = \nu \cdot 50$, $\nu = 1, 2, 4, 6, \ldots, 20$ were considered. The capacity curve and the curve entitled "Threshold for given degree distribution" mark the theoretical limits for codes of rate and $1/2$ and codes of rate $1/2$ with degree distribution $L(x)$, $R(x)$, respectively.

C. Comparison

Let us now compare convolutional codes to optimized LDPC codes in terms of required SNR when both the structural delay and the required error rate are specified. Figure 6 shows the required SNR for given structural delay and $\mathrm{BER} = 10^{-5}$. Three types of processing and encoding are considered, namely serial processing and non-systematic encoding, serial processing and systematic encoding, as well as block processing.

It is to observe that there is a significant gap between the convolutional codes and the PEG-optimized LDPC codes. Considering the case that a BER of 10^{-5} at $10\log_{10}(E_\mathrm{b}/N_0) = 4$ dB should be obtained, one would have to accept a structural delay of 27 time-steps of duration T_b for convolutional codes. In order to obtain this performance at $10\log_{10}(E_\mathrm{b}/N_0) = 4$ dB, the most appropriate PEG code imposes 211 time-steps of duration T_b for serial processing and non-systematic encoding, 118 time-steps for serial processing and systematic encoding, and 79 time-steps of duration T_b when block processing is applied. It is to observe that serial processing and non-systematic encoding leads to



(a) Serial processing, non-syst. encoding (b) Serial processing, systematic encoding (c) Block (parallel) processing

Fig. 6. Performance comparison for $\mathrm{BER} = 10^{-5}$ of PEG-optimized LDPC codes and convolutional codes.

significantly poorer performance results when compared to serial processing and systematic encoding. This does not only result from the higher delay caused by the encoding, but also from the post-processing with a dense matrix, what leads to a high number of wrong bits in an erroneously decoded frame.

It is to observe that PEG-optimized LDPC codes can in principle obtain the required BER at lower SNR. However, significantly longer delays have to be accepted then. Considering again Figure 6 and comparing the results to theoretical limits, it can be seen that the convolutional codes outperform the Gallager bound (random-coding bound) and are even located left of the sphere packing bound when serial processing is chosen. Possible explanations for this behavior include the fact that convolutional codes do not obey a block structure and, consequently, the sphere packing bound does not mark a non-existence bound for them. Thus, we conclude that for rate $1/2$ and very low tolerated structural delay, block codes cannot outperform convolutional codes in principle. Let us now consider the performance of PEG codes with serial processing and systematic encoding in more detail. The optimized LDPC codes approach the Gallager bound very closely for short lengths. For moderate lengths of $200 \leq n \leq 1000$, an almost constant gap of approximately 0.6 dB to the Gallager bound exists. The sphere packing bound is achieved by about 1.85 dB for $n = 50$ and around 1 dB for $n = 1000$.

Similar results are obtained when the FER is considered as the measure of performance. Of course, here no performance degradation is noticed when using non-systematic encoding instead of systematic encoding. Due to space limitation, results on the FER obtained with different transmission scenarios are omitted.

D. Obtaining acceptable FERs *for longer frames*

So far, we have only considered frame lengths up to $\Delta_\mathrm{cc} + 1 = 56$ for convolutional codes. In practical implementations, superordinate protocols may consider more symbols as one block. For that reason, we investigate the required signal-to-noise ratio to obtain a specified FER when $1 \leq L_\mathrm{F} \leq 500$ is chosen. Again, we compare convolutional codes and block codes. Considering convolutional codes and frames of lengths $L_\mathrm{F} \leq 56$, the choice of f and the code is given by the precedent analysis. For $L_\mathrm{F} > 56$, we apply the convolutional code with $|\mathcal{S}| = 1024$ and $f = 5$. Recalling the results presented in Figure 4 it is clear that this guarantees the best possible choice of code from the set of given codes for each value of L_F. Concerning the block code, the parameters n and k can easily be chosen to fit the requirements of a given frame length L_F. Figure 7 shows the required signal-to-noise ratio to obtain a FER of 10^{-3} on frames of different length.

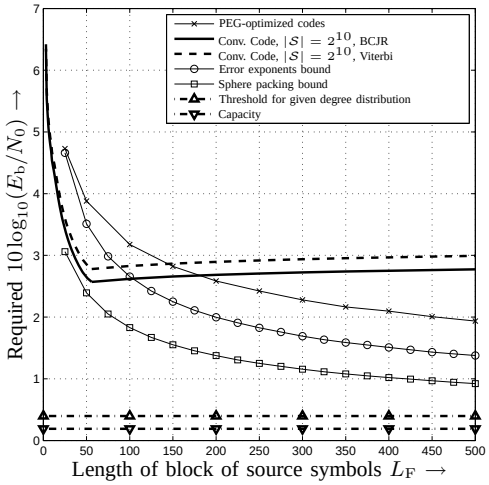


Fig. 7. Required SNR to obtain $\mathrm{FER} = 10^{-3}$ on blocks of source symbols of given length.

It can be observed in Figure 7 that the required SNR for given FER rises with $L_\mathrm{F} > 56$ if convolutional codes are used. This is reasonable, as from this point on the error-correction performance of the code stays constant but the decoding task

becomes more and more complicated with increasing L_{F}. Thus, it is advantageous to use block codes if one is interested in obtaining reasonable FERs on frames of higher length. Nevertheless, convolutional codes are the first choice in this scenario up to $L_{\mathrm{F}} = 144$ and $L_{\mathrm{F}} = 181$ when using the Viterbi decoder and the BCJR decoder, respectively. For block lengths of $L_{\mathrm{F}} \geq 145$ and $L_{\mathrm{F}} \geq 182$, respectively, block codes are a reasonable choice as they provide the better decoding performance.

Convolutional codes with BCJR decoding allow to obtain $\mathrm{FER} = 10^{-3}$ at $10\log_{10}(E_{\mathrm{b}}/N_0) \approx 2.57$ dB for $L_{\mathrm{F}} = 56$. LDPC codes need to exceed $L_{\mathrm{F}} \approx 200$ to obtain this error rate at equal or lower SNR. Consequently, there exists a *decoding gap* within the range of $56 \leq L_{\mathrm{F}} \leq 200$. Convolutional codes with more than 2^{10} states could be used to shift this gap to higher values of L_{F}. However, these codes are considered as not being practical. It is a more desirable solution to close this gap using block codes with optimized decoders. Reasons for this include the fact that block codes with appropriate parameters can be decoded with low computational effort. First results on this topic were accomplished in [32] and [33]. Detailed investigations on codes of moderate length will be presented elsewhere.

VII. Conclusions

We conclude this paper by stating that, despite of the significant progress within the last decade, block codes cannot keep up with convolutional codes if a good decoding performance and a very short structural delay are required at the same time. However, if one allows for a higher structural delay, LDPC codes can clearly outperform the considered convolutional codes. LDPC codes are also advantageous when longer frames are taken into account and the FER is the performance criterion of interest. For practical implementations it is favorable to use convolutional codes for applications requiring a very low delay and PEG-optimized LDPC codes if very low SNRs are required. PEG-optimized LDPC codes approach the bound derived from the Gallager error exponent very tightly for very short block lengths ($n \approx 50$) and do not differ more than 0.7 dB from it for $n = 1000$. When using block codes, systematic encoding should always be preferred to non-systematic encoding. The reasons for this are the lower delay when the information symbols are transmitted at the beginning of the codeword and the better error correcting performance due to the absence of the post-processing step using the generator matrix.

References

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

[2] D. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[3] JTC Joint Technical Committee, "Digital video broadcasting (DVB); second generation framing structure, channel coding and modulation systems for broadcasting, interactive services, news gathering and other broadband satellite applications," European Telecommunications Standards Institute, Tech. Rep., March 2005.

[4] IEEE Std. 802.3an, "IEEE Standard for information technology, Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications," Institute of Electrical and Electronics Engineers, Tech. Rep., June 2006.

[5] IEEE Std. 802.16e, "IEEE standard for local and metropolitan area networks, Part 16: Air interface for fixed and mobile broadband wireless access systems. Amendment 2: Physical and medium access control layers for combined fixed and mobile operation in licensed bands," Institute of Electrical and Electronics Engineers (IEEE), Tech. Rep., December 2005.

[6] E. 3rd Generation Partnership Project (3GPP)., "Universal mobile telecommunications system (UMTS); multiplexing and channel coding (FDD) 3GPP TS 25.212, version 6.10.0, release 6," European Telecommunications Standards Institute (ETSI), Tech. Rep., 2006.

[7] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Geneva, Switzerland, May 1993, pp. 1064–1070.

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

[9] J. Craddock, M. Flanagan, and A. Fagan, "On a class of high-girth LDPC codes based on finite multidimensional lattices," in *Proceedings of the International ITG Conference on Source and Channel Coding (SCC)*, Ulm, Germany, January 2008.

[10] B. Baumgartner, M. Reinhard, G. Richter, and M. Bossert, "Performance of forward error correction for IEEE 802.16e," in *10th International OFDM Workshop (InOWo)*, Hamburg, Germany, August 2005.

[11] P. Elias, "Coding for noisy channels," in *IRE National Convention Record*, ser. Part 4, 1955, pp. 37–47.

[12] R. Johannesson and K. S. Zigangirov, *Fundamentals of Convolutional Coding*. IEEE Press, 1999.

[13] S. Dolinar, D. Divsalar, and F. Pollara, "Turbo code performance as a function of code block size," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, Cambridge, MA, USA, August 1998, p. 32.

[14] Communications Systems and Research Section, JPL. Code imperfectness. Website. [Online]. Available: http://www331.jpl.nasa.gov/public/imperfectness.html

[15] S. Haykin, *Digital Communications*. John Wiley and Sons, Inc., 1988.

[16] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions Information Theory*, vol. 20, pp. 284–287, March 1974.

[17] J. Huber, *Trelliscodierung (in German)*, ser. Nachrichtentechnik. Erlangen, Germany: Springer-Verlag Berlin Heidelberg, 1992.

[18] J. Heller and I. Jacobs, "Viterbi decoding for satellite and space communication," *IEEE Transactions on Communication Technology*, vol. 19, no. 5, pp. 835–848, October 1971.

[19] A. Viterbi, "Error-bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, April 1967.

[20] S. Huettinger, "Analysis and design of power-efficient coding schemes," Ph.D. dissertation, University of Erlangen-Nuremberg, Erlangen, Germany, 2004.

[21] J. G. Proakis, *Digital Communications*, S. W. Director, Ed. McGraw-Hill Higher Education, 2001.

[22] R. Urbanke. LdpcOpt - a fast and accurate degree distribution optimizer for LDPC ensembles. Website. [Online]. Available: http://lthcwww.epfl.ch/research/ldpcopt/index.php

[23] M. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533–547, September 1981.

[24] X.-Y. Hu, M. Fossorier, and E. Eleftheriou, "On the computation of the minimum distance of low-density parity-check codes," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Paris, France, June 2004.

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

[26] C. Shannon, "Probability of error for optimal codes in a Gaussian channel," *Bell System Technical Journal*, vol. 38, pp. 611–656, May 1959.

[27] S. Peeters, "Vergleich von Kanalcodierverfahren," Studienarbeit (in German), University of Erlangen-Nuremberg, Erlangen, Germany, August 1998.

[28] G. Wiechman and I. Sason, "An improved sphere-packing bound for finite-length codes on symmetric memoryless channels," *Submitted to IEEE Transactions on Information Theory*, March 2007.

[29] G. Wiechman, "Private communication," April 2007, by E-mail.

[30] J.-P. Adoul, "Error intervals and cluster density in channel modeling (corresp.)," *IEEE Transactions on Information Theory*, vol. 20, no. 1, pp. 125–129, January 1974.

[31] J. B. Huber, “Codierung fuer gedaechtnisbehaftete kanaele (in german),” Ph.D. dissertation, Hochschule der Bundeswehr Muenchen, Munich, Germany, April 1982.
[32] T. Hehn, J. Huber, S. Laendner, and O. Milenkovic, “Multiple-bases belief-propagation decoding for short block-codes,” in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, Nice, France, June 2007, pp. 311–315.
[33] T. Hehn, J. Huber, O. Milenkovic, and S. Laendner, “Multiple-bases belief-propagation decoding of high-density cyclic codes,” *submitted to IEEE Transactions on Communications*, 2007.