Unifying Analysis and Design of Rate-Compatible Concatenated Codes

Alexandre Graell i Amat, *Senior Member, IEEE,* Lars K. Rasmussen, *Senior Member, IEEE,* and Fredrik Brännström, *Member, IEEE*

***Abstract*—An improved concatenated code structure, which generalizes parallel and serially concatenated convolutional codes is presented and investigated. The structure is ideal for designing low-complexity rate-compatible code families with good performance in both the waterfall and error floor regions. As an additional feature, the structure provides a unified analysis and design framework, which includes both parallel and serially concatenated codes as particular cases. We derive design criteria for the generalized class of concatenated convolutional codes based on union bounds for the error probability and extrinsic information transfer (EXIT) charts for the decoding threshold.**

I. Introduction

Recently, a powerful class of serially concatenated convolutional codes (SCCCs) was proposed and analyzed in [1–4], which significantly outperforms standard SCCCs, especially for high code rates. In contrast to standard SCCCs [5] characterized by an outer code concatenated with a rate $R_{\mathrm{I}} \leq 1$ inner code, the codes proposed in [1–4] are obtained by puncturing both inner-code parity bits and systematic bits, thereby obtaining rates beyond the outer code rate. The key idea is to puncture the outer code bits *after* they are used to perform the inner encoding. This idea has been proposed and validated by simulation in [1, 2], and formally addressed and justified in [3, 4]. This allows for a constant block length outer code, maintaining its distance spectrum properties and thus keeping the interleaver gain constant for all code rates [4]. Furthermore, this approach allows for the construction of rate-compatible SCCCs. A performance analysis based on bounds for the error probability in the error floor (EF) region was performed in [4], demonstrating significantly lower error floors for high code rates. The corresponding code design procedure proposed in [4] was later extended in [6], incorporating additional design constraints for the waterfall (WF) region based on extrinsic information transfer (EXIT) chart analysis. The ideas in [4, 6] were further extended by the authors in [7], allowing for a more general concatenated code structure.

A. Graell i Amat is with the Electronics Department, IT/TELECOM Bretagne, 29238 Brest Cedex 3, France. (e-mail: alexandre.graell@telecom-bretagne.eu).

L. K. Rasmussen was with the Institute for Telecommunications Research, University of South Australia, Mawson Lakes, SA 5095, Australia. He is now with the Communication Theory Laboratory, School of Electrical Engineering, and the ACCESS Linnaeus Center, Royal Institute of Technology, 100 44 Stockholm, Sweden (e-mail: Lars.Rasmussen@ieee.org).

F. Brännström was with Quantenna Communications Inc., 3450 W. Warren Avenue Fremont, CA 94538, USA. He is now with the Division of Communication Systems and Information Theory, Department of Signals and Systems, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden (e-mail: fredrik.brannstrom@chalmers.se).

A. Graell i Amat was supported by a Marie Curie Intra-European Fellowship within the 6th European Community Framework Programme. L. K. Rasmussen was supported by the Australian Research Council under ARC Grant DP0881160, the Swedish Research Council under VR grant 621-2009-4666, and the European Research Council under the Seventh Framework Programme ERC Grant 228044. F. Brännström was supported by the Swedish Research Council under VR grant 621-2006-4872.

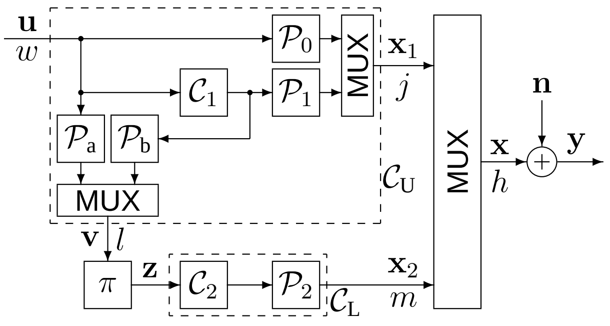

Fig. 1. A block diagram of the generalized code structure.

In this paper, we conduct a thorough investigation of the general concatenated code structure in [7]. The aim of our investigation is to find low-complexity rate-compatible code families with superior performance in both the WF and EF regions as compared to previous code structures. By introducing additional puncturing into traditional code structures, we obtain additional degrees of freedom for code design. The structure provides a unified framework for the analysis of general concatenated codes, including both parallel and serial concatenations. We derive design criteria for the proposed generalized class of concatenated codes based on union bounds for the error probability, leading to performance improvements in the EF region. We then analyze convergence properties of such codes using EXIT chart techniques, and suggest design criteria for optimizing the performance in the WF region without sacrificing performance in the EF region. To demonstrate our design approach, we construct rate-compatible code examples from 4-state and 8-state constituent codes, exhibiting good performance in both the EF and WF regions over a wide range of code rates. The proposed rate-compatible codes offer performance improvements as compared to previously suggested rate-compatible concatenated convolutional codes, and are competitive as compared to recently proposed rate-compatible irregular repeat-accumulate codes [8].

II. Generalized Concatenated Codes

The proposed generalized concatenated code (GCC) structure is shown in Fig. 1. For simplicity we consider the concatenation of two constituent convolutional codes. The extension of the analysis and design approach to constituent block codes and to a larger number of constituents is straightforward. In Fig. 1 $\mathcal{C}_1$ and $\mathcal{C}_2$ denote rate-1 convolutional encoders. Further, $\mathcal{P}_{\mathrm{a}}$, $\mathcal{P}_{\mathrm{b}}$, and $\mathcal{P}_0$, $\mathcal{P}_1$, $\mathcal{P}_2$ represent puncturers of the respective bit streams and $\mathcal{C}$ denotes the overall concatenated code.

A. Rate-Compatible Puncturing

A puncturer $\mathcal{P}$ of a single codebit stream is commonly defined by a puncturing pattern $\mathbf{p}$ with a certain pattern length, N_{p}. For example, if $N_{\mathrm{p}} = 4$ and puncturer $\mathcal{P}$ is chosen to puncture every fourth bit, the pattern is described as $\mathbf{p} = [1, 1, 1, 0]$, where 0 represents a punctured position. The puncturing pattern is then repeated periodically for application to the codebit stream, and therefore the pattern length is sometimes referred to as the puncturing period.

Let δ_{p} denote the number of bits remaining at the output of the puncturer within a puncturing period. The permeability rate of the puncturer is defined as $\rho_{\mathrm{p}} = \frac{\delta_{\mathrm{p}}}{N_{\mathrm{p}}}$, where $0 \leq \delta_{\mathrm{p}} \leq N_{\mathrm{p}}$, and thus, $0 \leq \rho_{\mathrm{p}} \leq 1$. For the example above, $\delta_{\mathrm{p}} = 3$ and $\rho_{\mathrm{p}} = 3/4$.

Two codes of different rate, belonging to a rate-compatible code family, are said to be rate-compatible if the higher rate code is embedded into the lower rate code of the family. A rate-compatible code family can then be defined by a series of nested puncturing patterns [9]. For example, the series $[1, 1, 1, 1], [1, 1, 1, 0], [0, 1, 1, 0], [0, 1, 0, 0]$ represent a rate-compatible code family with permeability rates $1, \frac{3}{4}, \frac{1}{2}, \frac{1}{4}$, and code rates $R_u, \frac{4}{3}R_u, 2R_u, 4R_u$, respectively, where R_u is the rate of the unpunctured code. A more compact notation for a rate-compatible code family is defined by a *puncturing order* $\bar{\mathbf{p}}$ and a series of permeability rates. A puncturing order has as entries $\bar{p} \in \{1, \ldots, N_{\mathrm{p}}\}$; the position indices of the codebits within a puncturing period. For the example above, the puncturing order is $\bar{\mathbf{p}}_1 = [4, 1, 3, 2]$, since bit position 4 is punctured first, followed by positions 1 and 3. A specific puncturing pattern $\mathbf{p}$ is thus defined by the corresponding pair $\{\bar{\mathbf{p}}, \rho_{\mathbf{p}}\}$, e.g., $\mathbf{p} = [1, 1, 1, 0] \leftrightarrow \{\bar{\mathbf{p}}, \rho_{\mathbf{p}}\} = \{[4, 1, 3, 2], \frac{3}{4}\}$.

In Fig. 1 $\mathcal{P}_{\mathrm{a}}$, $\mathcal{P}_{\mathrm{b}}$, and $\mathcal{P}_k$ have puncturing patterns $\mathbf{p}_{\mathrm{a}}$, $\mathbf{p}_{\mathrm{b}}$, and $\mathbf{p}_{\mathrm{k}}$ of length N_{a}, N_{b}, and N_k, together with permeability rates ρ_{a}, ρ_{b}, and ρ_k, for $k = 0, 1, 2$, respectively.

B. Concatenated Code Structure

In Fig. 1 the binary information data is collected in a vector $\mathbf{u} \in \{\pm 1\}^K$ of length K, encoded by $\mathcal{C}_1$ and then punctured by a puncturer $\mathcal{P}_{\mathrm{b}}$. The information data is also directly punctured by $\mathcal{P}_{\mathrm{a}}$. These two punctured bit streams are multiplexed and collected in a vector $\mathbf{v} \in \{\pm 1\}^{N_\pi}$ of length $N_\pi = K(\rho_{\mathrm{a}} + \rho_{\mathrm{b}})$. The vector $\mathbf{v}$ is interleaved by π before being forwarded to encoder $\mathcal{C}_2$. Furthermore, the information bits in $\mathbf{u}$ and the output of $\mathcal{C}_1$ are also punctured by the puncturers $\mathcal{P}_0$ and $\mathcal{P}_1$, respectively. The outputs of these puncturers are multiplexed and collected in the vector $\mathbf{x}_1$ of length $N_{\mathrm{U}} = K(\rho_0 + \rho_1)$. The output of $\mathcal{C}_2$ is punctured by $\mathcal{P}_2$ and collected in the vector $\mathbf{x}_2$ of length $N_{\mathrm{L}} = N_\pi \rho_2 = K(\rho_{\mathrm{a}} + \rho_{\mathrm{b}})\rho_2$. These vectors are then finally multiplexed to form the codeword sequence of the overall code $\mathcal{C}$ as $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2] \in \{\pm 1\}^N$. Here $N = N_{\mathrm{U}} + N_{\mathrm{L}} = K/R$ denotes the total number of transmitted bits in $\mathbf{x}$ and R is the rate of the overall code $\mathcal{C}$ given by

$$R = \frac{K}{N} = \frac{K}{N_{\mathrm{U}} + N_{\mathrm{L}}} = \frac{1}{\rho_0 + \rho_1 + (\rho_{\mathrm{a}} + \rho_{\mathrm{b}})\rho_2}. \qquad (1)$$

The proposed GCC can be seen as the concatenation of two encoders, $\mathcal{C}_{\mathrm{U}}$ and $\mathcal{C}_{\mathrm{L}}$ (the two dashed boxes in Fig. 1). $\mathcal{C}_{\mathrm{U}}$ is referred to as the *upper encoder* and $\mathcal{C}_{\mathrm{L}}$ to the *lower encoder*. We also identify within encoder $\mathcal{C}_{\mathrm{U}}$ two separate encoders. Referring to Fig. 1, we define $\mathcal{C}_{\mathrm{ch}}$ as the upper encoder seen by the channel, i.e., the pairs $\mathbf{u} \longrightarrow \mathbf{x}_1$, and the encoder $\mathcal{C}_\pi$ as the upper encoder seen by the interleaver, i.e., the pairs $\mathbf{u} \longrightarrow \mathbf{v}$. The corresponding codeword sequence for the upper code is thus $\mathbf{x}_{\mathrm{U}} = [\mathbf{x}_1, \mathbf{v}]$ with $\mathcal{C}_{\mathrm{ch}}$ and $\mathcal{C}_\pi$ having code rates

$$R_{\mathrm{ch}} = \frac{K}{N_{\mathrm{U}}} = \frac{1}{\rho_0 + \rho_1}, \qquad R_\pi = \frac{K}{N_\pi} = \frac{1}{\rho_{\mathrm{a}} + \rho_{\mathrm{b}}}, \qquad (2)$$

respectively.

The code structure of Fig. 1 generalizes parallel and serially concatenated codes and provides a unified framework for the analysis of concatenated codes. Notice that the permeability rates of the five puncturers in Fig. 1 can be independently chosen between zero and one, resulting in a plethora of parallel, serial, and hybrid code structures. For example, consider the parallel concatenation of two encoders, $\mathcal{C}_1$ and $\mathcal{C}_2$. In the form of Fig. 1 this is obtained by setting $\rho_{\mathrm{a}} = 1$ and $\rho_{\mathrm{b}} = 0$. Similarly, consider the serial concatenation of two systematic encoders (punctured or unpunctured), outer $\mathcal{C}_{\mathrm{O}}$ and inner $\mathcal{C}_{\mathrm{I}}$. In the form of Fig. 1, this concatenation is equivalent to setting $\mathcal{C}_{\mathrm{ch}} \equiv \mathcal{C}_\pi \equiv \mathcal{C}_{\mathrm{O}}$ (i.e., $\mathbf{p}_0 = \mathbf{p}_{\mathrm{a}}$ and $\mathbf{p}_1 = \mathbf{p}_{\mathrm{b}}$) and setting $\mathcal{C}_{\mathrm{L}}$ to the parity part of $\mathcal{C}_{\mathrm{I}}$. The second example above allows us to point out a key difference of the GCC structure with respect to an SCC. In contrast to a standard SCC, we allow the *outer encoder seen by the interleaver* $\mathcal{C}_\pi$ to be different from the *outer encoder seen by the channel* $\mathcal{C}_{\mathrm{ch}}$. This provides more degrees of freedom for the construction of concatenated codes, resulting in performance improvements. It is worth pointing out that other classes of concatenations, as for example the partially concatenated convolutional code proposed in [10], are also particular cases of this code structure. The code structure in [10] is obtained by setting $\{\mathbf{p}_{\mathrm{a}}, \mathbf{p}_{\mathrm{b}}\}$ and $\{\mathbf{p}_0, \mathbf{p}_1\}$ to be complementary.

III. Upper Bounds to the Error Probability

For later use, let $d_{\mathrm{min}}^{\mathcal{C}}$, $d_{\mathrm{min}}^{\mathcal{C}_\pi}$, and $d_{\mathrm{min}}^{\mathcal{C}_{\mathrm{ch}}}$ denote the minimum distances of $\mathcal{C}$, $\mathcal{C}_\pi$, and $\mathcal{C}_{\mathrm{ch}}$, respectively. Also, let w, l, j, m, and h be the weights of the vectors $\mathbf{u}$, $\mathbf{v}$, $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}$, respectively, as indicated in Fig. 1. We denote by $A_{w,h}^{\mathcal{C}}$ the input-output weight enumerator (IOWE) of encoder $\mathcal{C}$, giving the number of codewords $\mathbf{x}$ of weight h generated by an input word $\mathbf{u}$ of weight w. Similarly, we denote by $A_{l,m}^{\mathcal{C}_{\mathrm{L}}}$ the IOWE of encoder $\mathcal{C}_{\mathrm{L}}$ with input weight l and output weight m. We also define the weight enumerator $A_{w,l,j}^{\mathcal{C}_{\mathrm{U}}}$, giving the number of codewords $\mathbf{x}_{\mathrm{U}} = [\mathbf{x}_1, \mathbf{v}]$ of $\mathcal{C}_{\mathrm{U}}$ with weight j at the output $\mathbf{x}_1$ of $\mathcal{C}_{\mathrm{ch}}$, and weight l at the output $\mathbf{v}$ of $\mathcal{C}_\pi$, generated by an input word $\mathbf{u}$ of weight w. We call $A_{w,l,j}^{\mathcal{C}_{\mathrm{U}}}$ the Interleaver-IOWE (Int-IOWE) of code $\mathcal{C}_{\mathrm{U}}$, since it enumerates the weights

$$P_b \leq \frac{1}{2} \sum_{h=d_{\min}^{\mathcal{C}}}^{N} \text{erfc}\left(\sqrt{\frac{hRE_b}{N_0}}\right) \sum_{w=1}^{K} \sum_{l=d_{\min}^{\mathcal{C}_\pi}}^{N_\pi} \sum_{j=d_{\min}^{\mathcal{C}_{\text{ch}}}}^{h} \sum_{n^{\text{U}}=1}^{n_{\text{M}}^{\text{U}}} \sum_{n^{\text{L}}=1}^{n_{\text{M}}^{\text{L}}} N_\pi^{n^{\text{U}}+n^{\text{L}}-l-1} \frac{(l!)^2}{p^{n^{\text{U}}+n^{\text{L}}} n^{\text{U}}! n^{\text{L}}!} \frac{w}{R_\pi} A_{w,l,j,n^{\text{U}}}^{\mathcal{C}_{\text{U}}} A_{l,h-j,n^{\text{L}}}^{\mathcal{C}_{\text{L}}} \tag{6}$$

at the output of $\mathcal{C}_{\text{ch}}$, j, conditioned on the weights, l, at the input of the interleaver, i.e., the output weights of $\mathcal{C}_\pi$.

Using the union bound, the bit-error rate (BER) in the EF region is upper bounded for an AWGN channel by [5, 11]

$$P_b \leq \frac{1}{2} \sum_{h=d_{\min}^{\mathcal{C}}}^{N} \sum_{w=1}^{K} \frac{w}{K} A_{w,h}^{\mathcal{C}} \text{erfc}\left(\sqrt{\frac{hRE_b}{N_0}}\right). \tag{3}$$

Using the concept of a *uniform interleaver* [11] the average IOWE of the code ensemble $\mathcal{C}$, denoted by $\bar{A}_{w,h}^{\mathcal{C}}$ can be written as a function of $A_{w,l,j}^{\mathcal{C}_{\text{U}}}$ and $A_{l,m}^{\mathcal{C}_{\text{L}}}$ as

$$\bar{A}_{w,h}^{\mathcal{C}} = \sum_{j=d_{\min}^{\mathcal{C}_{\text{ch}}}}^{h} \sum_{l=d_{\min}^{\mathcal{C}_\pi}}^{N_\pi} \frac{A_{w,l,j}^{\mathcal{C}_{\text{U}}} A_{l,h-j}^{\mathcal{C}_{\text{L}}}}{\binom{N_\pi}{l}}, \tag{4}$$

where $h = j + m$.

Let n^{U} denote the number of error events concatenated in a codeword of $\mathcal{C}_{\text{U}}$ with output weights j and l generated by a weight w information sequence, and define n_{M}^{U} as the maximum value of n^{U}. Let $A_{w,l,j,n^{\text{U}}}^{\mathcal{C}_{\text{U}}}$ denote the corresponding number of these codewords. Further, let n^{L} denote the number of error events concatenated in a codeword of $\mathcal{C}_{\text{L}}$ with weight m generated by a weight l input sequence, and let $A_{l,m,n^{\text{L}}}^{\mathcal{C}_{\text{L}}}$ denote the corresponding number of these codewords [5]. Define also n_{M}^{L} as the maximum value of n^{L}. To evaluate $A_{w,l,j}^{\mathcal{C}_{\text{U}}}$ and $A_{l,m}^{\mathcal{C}_{\text{L}}}$, consider a rate p/n convolutional code with memory ν. For N_π much larger than ν, we can write,

$$\begin{aligned} A_{w,l,j}^{\mathcal{C}_{\text{U}}} &\leq \sum_{n^{\text{U}}=1}^{n_{\text{M}}^{\text{U}}} \binom{N_\pi/p}{n^{\text{U}}} A_{w,l,j,n^{\text{U}}}^{\mathcal{C}_{\text{U}}}, \\ A_{l,m}^{\mathcal{C}_{\text{L}}} &\leq \sum_{n^{\text{L}}=1}^{n_{\text{M}}^{\text{L}}} \binom{N_\pi/p}{n^{\text{L}}} A_{l,m,n^{\text{L}}}^{\mathcal{C}_{\text{L}}}. \end{aligned} \tag{5}$$

Following further derivations, (5) can be combined with (3)–(4), and the BER in the EF region is upper bounded by the expression (6) at the top of the page, since $\frac{N!}{(N-l)!} \leq N^l$ and $\frac{(N-l)!}{N!} \leq \frac{l!}{N^l}$ using the property that $\frac{1}{N-l+1} \leq \frac{l}{N}$ for $N \geq l$. The analysis in (3)–(6) is general and unifies the analysis carried out in [11] and [5] for PCCCs and SCCCs, respectively.

For a large interleaver length N_π the term with the largest exponent of N_π in (6) provides the dominant contribution to the bound on the error probability. This largest exponent is defined as

$$\alpha_{\text{M}} \triangleq \max_{h,w,l,j} \{n^{\text{U}} + n^{\text{L}} - l - 1\}. \tag{7}$$

Let $h(\alpha_{\text{M}})$ denote the output weight associated with the largest exponent of N_π. Using α_{M} as the exponent of N_π in (6) and approximating the summations only by the terms corresponding to $h = h(\alpha_{\text{M}})$, the following result, asymptotic with respect to E_b/N_0, is obtained:

$$P_b \overset{\sim}{\leq} C N_\pi^{\alpha_{\text{M}}} \text{erfc}\left(\sqrt{h(\alpha_{\text{M}}) \frac{RE_b}{N_0}}\right), \tag{8}$$

where C is a constant independent of N_π.

Denote by $d_i^{\mathcal{C}}$ the minimum weight of code $\mathcal{C}$ sequences generated by input sequences of weight i. For instance, $d_2^{\mathcal{C}}$ and $d_3^{\mathcal{C}}$ are the minimum weight of sequences of $\mathcal{C}$ generated by input sequences of weight 2 and 3, respectively. To further evaluate α_{M} and $h(\alpha_{\text{M}})$, we consider only the case of recursive constituent encoders where an input weight of one results in an infinite output weight. The evaluation can be divided into two separate cases, namely the case where $\rho_{\text{b}} = 0$ and $\rho_{\text{a}} = 1$, and the case where $\rho_{\text{b}} \neq 0$, respectively, since the case $\rho_{\text{b}} = 0$ and $\rho_{\text{a}} \neq 1$ is discarded because $|\mathbf{v}| < K$. Notice that the first case corresponds to a PCCC, whereas $\rho_{\text{b}} \neq 0$ comprises SCCC and other concatenated structures, like the partially concatenated codes proposed in [10].

1) *Case* $\rho_{\text{b}} \neq 0$: In this case, $d_{\min}^{\mathcal{C}_\pi} \geq 2$ and α_{M} in (7) is given by

$$\alpha_{\text{M}} = -\left\lfloor \frac{d_{\min}^{\mathcal{C}_\pi} + 1}{2} \right\rfloor. \tag{9}$$

The corresponding $h(\alpha_{\text{M}})$ is

$$h(\alpha_{\text{M}}) = \frac{d_{\min}^{\mathcal{C}_\pi} d_2^{\mathcal{C}_{\text{L}}}}{2} + d^{\mathcal{C}_{\text{ch}}}(d_{\min}^{\mathcal{C}_\pi}) \tag{10}$$

for $d_{\min}^{\mathcal{C}_\pi}$ even, and

$$h(\alpha_{\text{M}}) = \frac{(d_{\min}^{\mathcal{C}_\pi} - 3) d_2^{\mathcal{C}_{\text{L}}}}{2} + d_3^{\mathcal{C}_{\text{L}}} + d^{\mathcal{C}_{\text{ch}}}(d_{\min}^{\mathcal{C}_\pi}) \tag{11}$$

for $d_{\min}^{\mathcal{C}_\pi}$ odd. In (10)-(11) $d^{\mathcal{C}_{\text{ch}}}(d_{\min}^{\mathcal{C}_\pi})$ is the minimum weight of $\mathcal{C}_{\text{ch}}$ code sequences corresponding to the minimum distance, $d_{\min}^{\mathcal{C}_\pi}$, of $\mathcal{C}_\pi$. Notice that for even $d_{\min}^{\mathcal{C}_\pi}$ the weight at the output of $\mathcal{C}_L$ is the weight of a codeword $\mathbf{x}_2 \in \mathcal{C}_L$ that concatenates $d_{\min}^{\mathcal{C}_\pi}/2$ error events with weight $d_2^{\mathcal{C}_{\text{L}}}$, which explains the term $\frac{d_{\min}^{\mathcal{C}_\pi} d_2^{\mathcal{C}_{\text{L}}}}{2}$ in (10). On the other hand, in (11) for odd $d_{\min}^{\mathcal{C}_\pi}$ at the output of $\mathcal{C}_L$ we have $(d_{\min}^{\mathcal{C}_\pi} - 3)/2$ error events generated by a weight-2 input sequence, which result in the term $\frac{(d_{\min}^{\mathcal{C}_\pi} - 3) d_2^{\mathcal{C}_{\text{L}}}}{2}$, and one error event generated by a weight-3 input sequence, which results in the term $d_3^{\mathcal{C}_{\text{L}}}$ [5].

2) *Case* $\rho_{\text{b}} = 0$ *and* $\rho_{\text{a}} = 1$ (PCCC): In this case, $d_{\min}^{\mathcal{C}_\pi} = 1$, $l = w$ and α_{M} in (7) is given by

$$\alpha_{\text{M}} = \max_{w} \left\{2 \left\lfloor \frac{w}{2} \right\rfloor - w - 1\right\} = -1 \tag{12}$$

with

$$h(\alpha_{\text{M}}) = d_2^{\mathcal{C}_{\text{ch}}} + d_2^{\mathcal{C}_{\text{L}}}, \tag{13}$$

which is the result reported in [11].

A. Design Criteria for the Error Floor

Combining (8) with the results for α_{M} and $h(\alpha_{\mathrm{M}})$ in (9)–(13), some design guidelines can be formulated. To minimize the approximate bound in (8), α_{M} must be minimized and $h(\alpha_{\mathrm{M}})$ maximized, corresponding to maximizing $d_{\mathrm{min}}^{\mathcal{C}_\pi}$, $d_2^{\mathcal{C}_{\mathrm{L}}}$, $d_3^{\mathcal{C}_{\mathrm{L}}}$, $d^{\mathcal{C}_{\mathrm{ch}}}(d_{\mathrm{min}}^{\mathcal{C}_\pi})$, and $d_2^{\mathcal{C}_{\mathrm{ch}}}$ in (9)–(13). This is done by the following design considerations:

- The weight enumerator (WE) of $\mathcal{C}_\pi$ ($A_l^{\mathcal{C}_\pi}$) giving the number of codewords of weight l ($l = \{d_{\mathrm{min}}^{\mathcal{C}_\pi}, d_{\mathrm{min}}^{\mathcal{C}_\pi} + 1, \ldots, N_\pi\}$), must be optimized, i.e, the minimum distance $d_{\mathrm{min}}^{\mathcal{C}_\pi}$ and the successive distances must be maximized and their multiplicities minimized.
- The IOWE $A_{l,m}^{\mathcal{C}_{\mathrm{L}}}$ of the lower encoder $\mathcal{C}_{\mathrm{L}}$ must be optimized, i.e, $d_2^{\mathcal{C}_{\mathrm{L}}}$, $d_3^{\mathcal{C}_{\mathrm{L}}}$ and successive terms must be maximized and their multiplicities minimized.
- Since $h(\alpha_{\mathrm{M}})$ increases with $d^{\mathcal{C}_{\mathrm{ch}}}(d_{\mathrm{min}}^{\mathcal{C}_\pi})$ (and $d_2^{\mathcal{C}_{\mathrm{ch}}}$ in the case of a PCCC) the output weight spectrum of $\mathcal{C}_{\mathrm{ch}}$ conditioned on the output weight of $\mathcal{C}_\pi$ must be optimized in terms of maximizing $d^{\mathcal{C}_{\mathrm{ch}}}(d_{\mathrm{min}}^{\mathcal{C}_\pi})$ and successive terms, i.e, the Int-IOWE of code $\mathcal{C}_{\mathrm{U}}$ must be optimized.

Note that the design considerations are valid for all concatenations in the form of Fig. 1, comprising both PCCCs and SCCCs.

IV. Decoding Threshold Analysis

The bounds in Section III provide accurate predictions of the performance in the EF, but fail in predicting the performance in the WF. We therefore consider EXIT chart analysis [12] to predict the performance of concatenated codes in the WF region.

Let $A(\mathbf{x})$ denote the *a priori* information in logarithmic scale (L-values) for the transmitted bits $\mathbf{x}$ based on the matched filter output $\mathbf{y}$. Since the channel is an AWGN channel, the *a priori* information for $\mathbf{x}$ can be expressed as [12]

$$A(\mathbf{x}) \triangleq \frac{4RE_b}{N_0}\mathbf{y}. \tag{14}$$

Further, let $A(\mathbf{v})$, $A(\mathbf{z})$, $A(\mathbf{x}_1)$, and $A(\mathbf{x}_2)$ denote the *a priori* information for the vectors $\mathbf{v}$, $\mathbf{z}$, $\mathbf{x}_1$, and $\mathbf{x}_2$ in Fig. 1, respectively. In convergence analysis using EXIT charts it is common to model the *a priori* information as a Gaussian random variable [12]. For example, let A denote the *a priori* information for the random variable $X \in \{+1, -1\}$. The Gaussian assumptions for A can then be stated as

$$A = \frac{\sigma^2}{2}X + W, \tag{15}$$

where W is a zero-mean Gaussian random variable with variance σ^2. The average mutual information (MI) between A and X, $I_A \triangleq I(A;X)$, depends only on σ, i.e., $I_A = J(\sigma)$ where J is defined as [12]

$$J(\sigma) = 1 - \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{+\infty} e^{-\frac{(\xi-\sigma^2/2)^2}{2\sigma^2}} \log_2\big(1+e^{-\xi}\big)\,\mathrm{d}\xi. \tag{16}$$

Let $I_{A(v)}$, $I_{A(z)}$, $I_{A(x)}$, $I_{A(x_1)}$, and $I_{A(x_2)}$ denote the average *a priori* MI for the elements in $\mathbf{v}$, $\mathbf{z}$, $\mathbf{x}$, $\mathbf{x}_1$, and $\mathbf{x}_2$ for $\mathcal{C}_{\mathrm{U}}$ and $\mathcal{C}_{\mathrm{L}}$ in Fig. 1, respectively. Note that $A(\mathbf{x})$, $A(\mathbf{x}_1)$, and $A(\mathbf{x}_2)$ in (14) are already Gaussian according to (15) with variance $\sigma^2 = 8RE_b/N_0$. Their average *a priori* MI is therefore

$$I_{A(x)} = I_{A(x_1)} = I_{A(x_2)} = J\Big(\sqrt{8RE_b/N_0}\Big). \tag{17}$$

Let $E(\mathbf{v})$ and $E(\mathbf{z})$ denote the extrinsic information generated by the soft *a posteriori* probability (APP) decoder for $\mathcal{C}_{\mathrm{U}}$ and $\mathcal{C}_{\mathrm{L}}$ in Fig. 1, respectively. $E(\mathbf{z})$ depends on the component code $\mathcal{C}_2$, the chosen puncturing pattern $\mathbf{p}_2$, the *a priori* information $A(\mathbf{z})$, and the *a priori* information $A(\mathbf{x}_2)$. The extrinsic MI $I_{E(z)}$ can therefore be found via a separate Monte Carlo simulation of the lower encoder/decoder for different values of σ_z and σ_{x_2} for $A(\mathbf{z})$ and $A(\mathbf{x}_2)$ modelled as in (15), where

$$\sigma_z = J^{-1}\big(I_{A(z)}\big), \tag{18}$$

$$\sigma_{x_2} = J^{-1}\big(I_{A(x_2)}\big) = \sqrt{8RE_b/N_0}, \tag{19}$$

as described in, e.g., [12,13]. The J function in (16) or its inverse J^{-1} in (18)–(19) cannot be expressed in closed form, but closed form approximations can be found in [13,14]. It follows that the extrinsic MI $I_{E(z)}$ for a specific code $\mathcal{C}_2$ and puncturing pattern $\mathbf{p}_2$ can be generated for all $I_{A(z)} \in [0,1]$ and $I_{A(x_2)} \in [0,1]$. In a similar way, the extrinsic MI $I_{E(v)}$ for the upper code, which depends on $\{\mathcal{C}_1, \mathbf{p}_{\mathrm{a}}, \mathbf{p}_{\mathrm{b}}, \mathbf{p}_0, \mathbf{p}_1\}$, can be generated for all $I_{A(v)} \in [0,1]$ and $I_{A(x_1)} \in [0,1]$. Hence,

$$I_{E(v)} \triangleq T_v\big(I_{A(v)}, I_{A(x_1)}, \mathbf{p}_{\mathrm{a}}, \mathbf{p}_{\mathrm{b}}, \mathbf{p}_0, \mathbf{p}_1\big), \tag{20}$$

$$I_{E(z)} \triangleq T_z\big(I_{A(z)}, I_{A(x_2)}, \mathbf{p}_2\big), \tag{21}$$

where T_v and T_z are referred to as the EXIT functions for the upper and lower codes, respectively.

Since $I_{A(z)} = I_{E(v)}$ and $I_{A(v)} = I_{E(z)}$, the convergence behavior of codes with the structure of Fig. 1 can be obtained by tracking the iterative evolution of the MIs

$$I_{E(v)} = T_v\Big(I_{E(z)}, J\Big(\sqrt{8RE_b/N_0}\Big), \mathbf{p}_{\mathrm{a}}, \mathbf{p}_{\mathrm{b}}, \mathbf{p}_0, \mathbf{p}_1\Big), \tag{22}$$

$$I_{E(z)} = T_z\Big(I_{E(v)}, J\Big(\sqrt{8RE_b/N_0}\Big), \mathbf{p}_2\Big). \tag{23}$$

The corresponding convergence analysis is conducted by projecting the EXIT functions (22)-(23) onto an EXIT chart, as suggested in [6,13], when only parity bits are periodically punctured. The respective EXIT chart can then be used to predict the convergence threshold and the performance for E_b/N_0 below the convergence threshold. Note that the two curves in the EXIT chart for $\mathcal{C}_{\mathrm{U}}$ and $\mathcal{C}_{\mathrm{L}}$ depend on E_b/N_0, in contrast to the EXIT chart for a standard serially concatenated code [14], where only the curve for the inner code depends on E_b/N_0.

Recently, the influence of periodic puncturing of systematic bits has been investigated for turbo codes [15]. In this case, conventional EXIT chart analysis breaks down, and a more complex three-dimensional EXIT chart analysis is required. Puncturing of systematic bits (bits in $\mathcal{P}_0$) is possible within the general framework presented in this paper; however, the search for puncturing patterns of systematic bits is more involved since the resulting codes must be checked for invertibility.

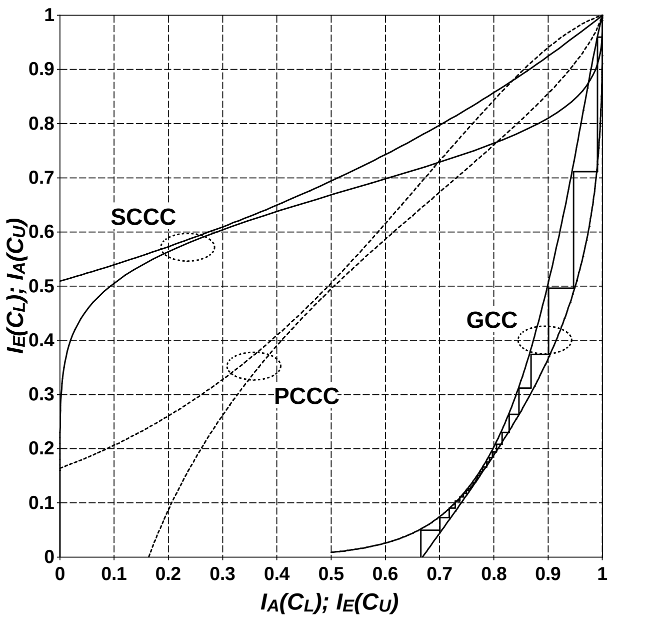

Fig. 2. EXIT charts for the $R = 2/3$ GCC, $R_\pi = 2/3$, and (ρ_1, ρ_2) optimized for the WF, and comparison to PCCCs and SCCCs.

When only periodic puncturing of parity bits are considered, the accuracy of conventional EXIT chart analysis has previously been demonstrated, e.g., Fig. 8 in [6] and Fig. 2 in [7].

As an example, in Fig. 2 we plot the EXIT chart for the $R = 2/3$ GCC with 4-state component encoders with generator polynomials $(5/7)_8$ in octal notation. $R_\pi = 2/3$ and permeability rates (ρ_1, ρ_2) are optimized for the WF region. Moreover, $\rho_0 = 1$, $\rho_a = 1$ and $\rho_b = 1/2$. A tunnel between the two curves opens at 1.70 dB, indicating convergence around this value. The iterative decoding trajectory for the GCC is also included in Fig. 2 to show the accuracy of the EXIT chart prediction. A random interleaver of length $N_\pi = 90000$ bits was used. Notice that if systematic bits are punctured periodically the EXIT chart technique has to be adapted to accurately model the convergence behavior [15]. For comparison, we also plot the EXIT charts for an 8-state PCCC and an SCCC consisting of the serial concatenation of an 8-state outer encoder and inner accumulator. The same encoder with generator polynomial $(15/13)_8$ was used for the PCCC and the outer encoder of the SCCC. The best convergence threshold is obtained by the PCCC (1.51 dB), while the tunnel for the SCCC opens at 1.61 dB.

V. Design Procedure for Rate-compatible Codes

For the proposed code structure the error performance depends on the puncturing patterns and, subsequently, on the permeability rates. Therefore, the objective is to optimize these parameters to ensure good performance in the EF and WF regions, while satisfying the rate-compatibility constraint. It is a design challenge to find turbo-like codes that perform well in both the EF and WF regions. A joint design approach based on both bounds and EXIT chart analysis may be appropriate. Such a joint approach, however, is yet to be formulated. As an alternative, the corresponding individual design criteria resulting from applying the two techniques independently can be considered jointly. Unfortunately, design criteria derived from analytical bounds and from EXIT charts, respectively, are in general competing rather than complementing, i.e., improving code performance in the EF region often leads to a penalty in terms of decoding threshold, and vice versa [6]. Moreover, finding the puncturing patterns $\mathbf{p}_\mathrm{a}$, $\mathbf{p}_\mathrm{b}$, $\mathbf{p}_0$, $\mathbf{p}_1$, and $\mathbf{p}_2$ for each code rate to minimize the decoding threshold using EXIT charts is prohibitively complex. Therefore, a low-complexity compromise is considered here.

A first observation is that for a given R_π, puncturing patterns $\mathbf{p}_\mathrm{a}$, $\mathbf{p}_\mathrm{b}$ should be kept fixed for all code rates, to guarantee rate compatibility. Here, we optimize patterns $\mathbf{p}_\mathrm{a}$ and $\mathbf{p}_\mathrm{b}$ according to the considerations in Section III-A. The key steps are as follows. Given patterns $\mathbf{p}_\mathrm{a}$ and $\mathbf{p}_\mathrm{b}$, step one is to find rate-compatible puncturing orders $\bar{\mathbf{p}}_0$, $\bar{\mathbf{p}}_1$, and $\bar{\mathbf{p}}_2$, based on the design criteria in Section III-A; thus ensuring good performance in the EF. Based on the puncturing orders found in step one, EXIT charts are applied in step two to find optimal permeability rates ρ_0, ρ_1, and ρ_2, minimizing the decoding thresholds. From the two-step approach, we obtain rate-compatible puncturing patterns, leading to rate-compatible codes that offer good performance in both the EF and the WF regions over a wide range of code rates.

VI. Numerical Results

For the examples here, we have chosen $\mathcal{C}_1$ and $\mathcal{C}_2$ to be identical, rate-one ($R_1 = R_2 = 1$) 4-state recursive convolutional codes, with polynomials $(5/7)_8$ in octal notation. Also, we consider the zero-termination of encoders $\mathcal{C}_1$ and $\mathcal{C}_2$. In all results, we assume $\rho_0 = 1$, i.e., the code is fully systematic, and $\rho_\mathrm{a} = 1$. We also considered the case $\rho_0 \neq 1$ and $\rho_\mathrm{a} \neq 1$; an improvement of the performance in the EF region was observed at the expense of a degradation of the WF. Note that if the systematic bits are punctured periodically, i.e., $\rho_0 \neq 0$, the conventional EXIT chart technique fails to predict the performance. In this case an adapted EXIT chart technique should be used to model the behavior [15]. Moreover, $\rho_0 = 1$ ensures that the code is invertible. We have further chosen $N_1 = 5! = 120$, providing a reasonable resolution for choosing R and R_π (hence $d_{\text{min}}^{\mathcal{C}_\pi}$). As representative examples, we focus on the cases of $R_\pi = 2/3$, and $R_\pi = 5/6$, i.e., $\rho_\mathrm{b} = 1/R_\pi - \rho_\mathrm{a} = 1/2$ and $1/5$, respectively (see (2)).

The puncturing patterns $\mathbf{p}_\mathrm{b} = [10]$ and $\mathbf{p}_\mathrm{b} = [01000]$ are used for $R_\pi = 2/3$, and $R_\pi = 5/6$, respectively. In Table I we give the puncturing order $\bar{\mathbf{p}}_1$ for $N_1 = 120$. The table lists the indices of the successive bit positions to be punctured (see Section II-A). For instance, if three bits in $\mathcal{P}_1$ are punctured, i.e., $\rho_1 = 117/120$, the bits in positions 0, 34 and 68 will be punctured. The puncturing order $\bar{\mathbf{p}}_2$ for $R_\pi = 2/3$ ($N_2 = N_1/R_\pi = 180$) and for $R_\pi = 5/6$ ($N_2 = N_1/R_\pi = 144$) are reported in Tables II and III, respectively.

The optimal choice for (ρ_1, ρ_2) depends on the adopted strategy, i.e., minimization of the EF or optimization of the decoding threshold, and on R_π. The optimal permeability rates ρ_2 of $\mathcal{P}_2$ for $R_\pi = 2/3$ are shown in Fig. 3. The corresponding permeability rates ρ_1 of $\mathcal{P}_1$ can be obtained from (1). The dashed curve with star markers gives the optimal rates ρ_2 for

TABLE I
PUNCTURING ORDER $\bar{\mathbf{p}}_1$ LISTED AS THE INDICES OF THE SUCCESSIVE BIT POSITIONS TO BE PUNCTURED AT THE OUTPUT OF $\mathcal{P}_1$ ($N_1 = 120$)

Number of punctured bits	Bit index									
1 - 10	0	34	68	94	16	50	108	80	24	58
11 - 20	8	42	100	86	114	74	28	62	4	46
21 - 30	104	90	20	38	54	12	118	76	66	98
31 - 40	30	110	84	70	2	48	18	36	56	92
41 - 50	10	116	78	102	26	60	40	88	112	6
51 - 60	72	22	52	96	32	106	14	64	82	44
61 - 70	119	51	85	25	103	67	11	39	93	75
71 - 80	111	59	19	33	5	45	97	79	115	63
81 - 90	15	29	89	55	107	71	1	41	21	83
91 - 100	101	9	49	35	69	113	91	57	27	77
101 - 110	3	43	105	17	61	95	31	81	7	47
111 - 120	117	65	99	23	109	73	13	53	87	37

TABLE II
PUNCTURING ORDER $\bar{\mathbf{p}}_2$ LISTED AS THE INDICES OF THE SUCCESSIVE BIT POSITIONS TO BE PUNCTURED AT THE OUTPUT OF $\mathcal{P}_2$ ($N_2 = 180$)

Number of punctured bits	Bit index									
1 - 10	0	42	84	126	154	20	105	63	140	167
11 - 20	31	94	53	116	73	10	147	133	174	161
21 - 30	36	100	78	48	111	25	4	67	89	121
31 - 40	15	58	144	157	171	130	137	151	178	39
41 - 50	97	81	28	108	70	7	164	45	119	56
51 - 60	17	91	33	76	103	51	114	22	124	61
61 - 70	86	12	142	169	159	134	149	2	66	176
71 - 80	128	41	98	27	110	74	49	8	82	35
81 - 90	155	165	93	138	19	118	59	106	146	173
91 - 100	13	87	69	123	44	54	24	101	131	162
101 - 110	5	79	152	113	30	64	38	179	141	170
111 - 120	95	16	72	127	47	158	88	55	109	1
121 - 130	143	122	26	80	135	175	99	65	163	40
131 - 140	18	153	115	6	71	46	92	57	129	34
141 - 150	107	148	83	11	166	29	139	117	62	102
151 - 160	156	77	177	43	23	132	9	90	50	120
161 - 170	145	172	75	21	104	52	168	32	136	85
171 - 180	112	3	60	150	68	160	14	96	37	125

TABLE III
PUNCTURING ORDER $\bar{\mathbf{p}}_2$ LISTED AS THE INDICES OF THE SUCCESSIVE BIT POSITIONS TO BE PUNCTURED AT THE OUTPUT OF $\mathcal{P}_2$ ($N_2 = 144$)

Number of punctured bits	Bit index									
1 - 10	0	42	84	114	20	63	130	99	31	73
11 - 20	52	10	122	92	107	137	36	58	78	25
21 - 30	4	47	15	68	118	103	88	126	141	134
31 - 40	96	111	39	55	28	7	81	66	17	45
41 - 50	75	33	60	22	50	12	71	1	124	105
51 - 60	90	132	116	139	95	85	109	101	128	120
61 - 70	38	26	8	56	79	64	43	18	49	30
71 - 80	136	112	70	3	13	35	142	93	87	76
81 - 90	61	23	98	53	82	123	106	131	117	41
91 - 100	6	67	16	100	48	27	140	125	108	80
101 - 110	62	37	94	135	11	54	113	21	86	2
111 - 120	74	32	119	46	69	133	104	5	89	24
121 - 130	121	59	40	97	77	9	129	57	29	115
131 - 140	143	44	83	19	102	72	127	51	14	110
141 - 144	34	91	138	65						

minimizing the SNR required to reach an EF of 10^{-9}, with an information block length of $K = 2400$ bits. The solid curve with square markers displays the corresponding optimal rates ρ_2, ensuring rate-compatible puncturing, while the solid curve with circle markers shows piece-wise linear approximations to the optimal rates for optimizing the decoding thresholds in the waterfall region, reaching a BER level of 10^{-5} after 10 iterations. We show this approximation rather than the true optimal rates for the sake of clarity in the figures. Using the approximate rates, rate-compatible puncturing is assured without any noticeable loss in terms of decoding threshold.

If ρ_2 is chosen to have a value close to the curve with stars, the performance is good in the EF region. Similarly, if ρ_2 is chosen to have a value close to the curve with circles, the performance is good in the WF region. Any non-decreasing curve between the two curves corresponds to a rate-compatible code with a trade-off in performance somewhere in between the two extreme cases. For $R_\pi = 2/3$ the optimal permeability rates in the EF are not rate-compatible. We observe that the optimal rates and the optimal rate-compatible rates in the EF

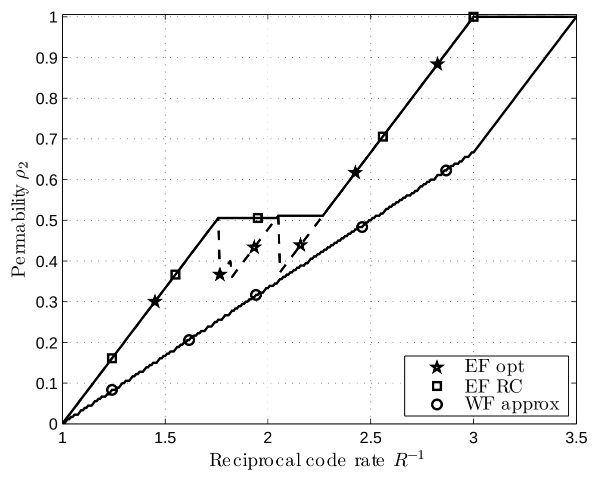

Fig. 3. The permeability rates ρ_2 as a function of the reciprocal overall code rate R^{-1} for the case $R_\pi = 2/3$, and information block length $K = 2400$ bits.

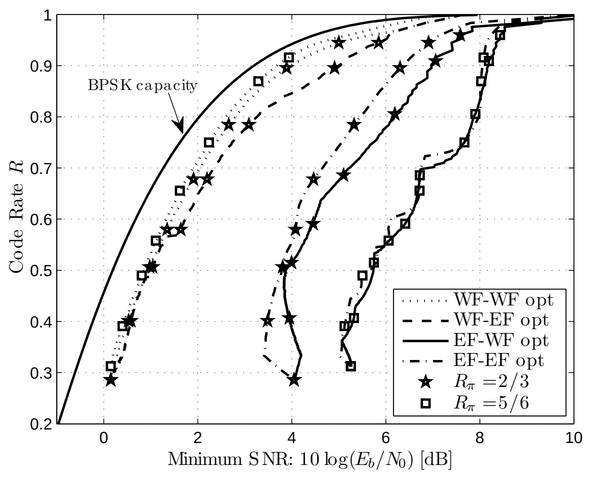

Fig. 4. Minimum required E_b/N_0 predicted by EXIT charts and error bounds for $K = 2400$ bits.

differ for code rates between $R = 4/9$ and $R = 4/7$. We further observe that for high code rates ($R > 4/7$), the optimal strategy in the EF is to keep as many bits as possible in x_2, i.e., with $\rho_1 = 0$ and ρ_2 obtained from (1) (the same strategy holds for $R_\pi = 1/2$); however, for lower code rates the best strategy requires parity bits to be included in $\mathbf{x}_1$ before all parity bits in $\mathbf{x}_2$ have been exhausted. For optimization in the WF region, ρ_1 and ρ_2 should be more evenly balanced. Based on the permeability rates approximation, we have $\rho_1 = \frac{3}{2}\rho_2$ for $\rho_2 \leq \frac{2}{3}$ and $\rho_1 = 1$ for $\rho_2 > \frac{2}{3}$ (see Fig. 3), which translates into $|\mathbf{x}_1| - K = (K + K\rho_1) - K = K\frac{3}{2}\rho_2 = \frac{K}{R_\pi}\rho_2 = |\mathbf{x}_2|$ until $\rho_1 = 1.0$, i.e., the parity bits in the two streams are perfectly balanced for as long as possible.

Similar plots can be obtained for other values of R_π, and the optimal strategy for choosing ρ_2 will in general depend on R_π. However, we can make the following observations. For low R_π, the resulting code turns into a *more serial* code, or, in other words, into a more *asymmetric* code. In this case, lower EFs can be obtained by increasing ρ_2 since the contribution of the *lower dimension* $\mathcal{C}_2$ on the output weight is more significant than that of $\mathcal{C}_1$. On the other hand, for high R_π the impact of the two code dimensions, $\mathcal{C}_1$ and $\mathcal{C}_2$, on the output weight tends to equalize. Note that in the extreme case of $R_\pi = 1$, the code marginalizes to a parallel concatenated code. Hence, if $\mathcal{C}_1 = \mathcal{C}_2$ the code is perfectly symmetric. It is therefore expected that for high R_π the permeability rates ρ_1 and ρ_2 tend to balance and that puncturing strategies for the EF and the WF regions are similar.

In Fig. 4, the minimum SNR required to reach a target BER waterfall level at 10^{-5} or an EF of 10^{-9} after 10 iterations is shown for $R_\pi = 2/3$ and $R_\pi = 5/6$. The notation "WF-WF opt" and "WF-EF opt" denote the WF performance based on permeability rate optimization for the WF and the EF regions, respectively. Likewise, the notation "EF-WF opt" and "EF-EF opt" denote the EF performance based on permeability rate optimization for the WF and the EF regions, respectively. The BPSK capacity limit has been included for reference. As expected, decreasing R_π (i.e., moving to a *more serial* code) leads to better performance in the EF but to a poorer WF. We observe that the best WF performance is for $R_\pi = 5/6$, while the case of $R_\pi = 2/3$ suffers a WF performance loss of 0.1 dB. The WF performance for the EF-optimized scheme with $R_\pi = 2/3$ is virtually the same as the WF-optimized scheme for low code rates up to $R = 4/7$. For higher code rates, a more substantial performance loss is incurred as expected based on the EF-optimized permeability rates. The WF performance for the EF-optimized scheme with $R_\pi = 5/6$ has not been included in the figure to maintain clarity. The performance exhibits an oscillating behavior of up to 1dB from the WF-optimized curve.

In terms of EF, the code with $R_\pi = 2/3$ performs significantly better than the code with $R_\pi = 5/6$. For $R_\pi = 2/3$ we observe a loss of around 0.5 dB in the EF region for the WF-optimized scheme as compared to the corresponding EF-optimized structure. In contrast, there are only small differences in EF performance for the WF- and EF-optimized schemes with $R_\pi = 5/6$.

To further demonstrate the observations and conclusions discussed above, we consider the BER prediction in the WF region using EXIT charts combined with EF bounds for $R = 1/2$ and $R = 8/9$. In Fig. 5, we show the predictions for $R_\pi = 2/3$ and $R_\pi = 5/6$ codes with 4-state constituents, and information block length for the EF curve of $K = 2400$ bits. For $R_\pi = 5/6$ we observe no significant differences in the EF performance at $R = 1/2$ and $R = 8/9$ for the WF- and EF-optimized schemes; however, a 0.1 dB advantage for the WF-optimized scheme is found in the WF region (as predicted above). For $R_\pi = 2/3$ and $R = 1/2$ we see only minor differences in both WF and EF regions between the two optimized schemes. For $R = 8/9$, however, there is a 1 dB advantage in both the WF region and in the EF region for the respective optimized schemes. Since a 1 dB gain is more significant in the WF region than in the EF region, these examples suggest that WF-optimized schemes provide a more favorable trade-off. From our analysis, $R_\pi = 2/3$ seems to be a good compromise in WF and EF.

In Fig. 6 we plot the performance of the proposed GCC

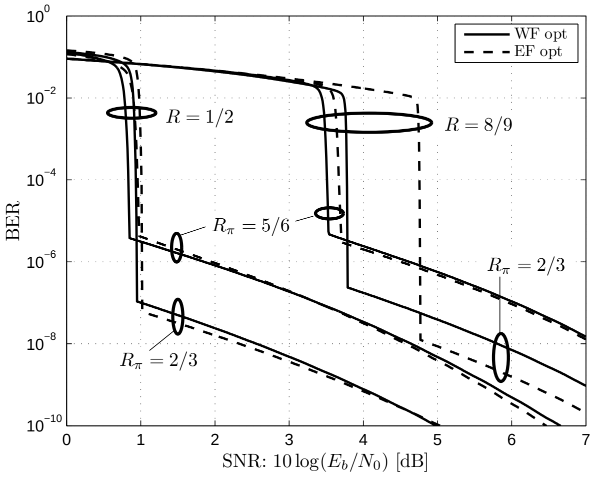

Fig. 5. BER approximations in the WF and EF regions based on EXIT charts and bounds are shown as functions of the SNR. $K = 2400$ bits.

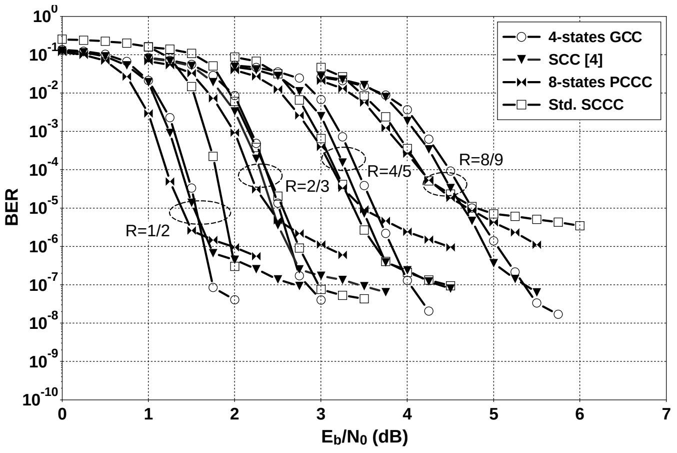

Fig. 6. BER performance of GCC with 4-state constituent codes. (ρ_1, ρ_2) are optimized for the WF region. $K = 2400$ bits, 10 iterations, random interleavers.

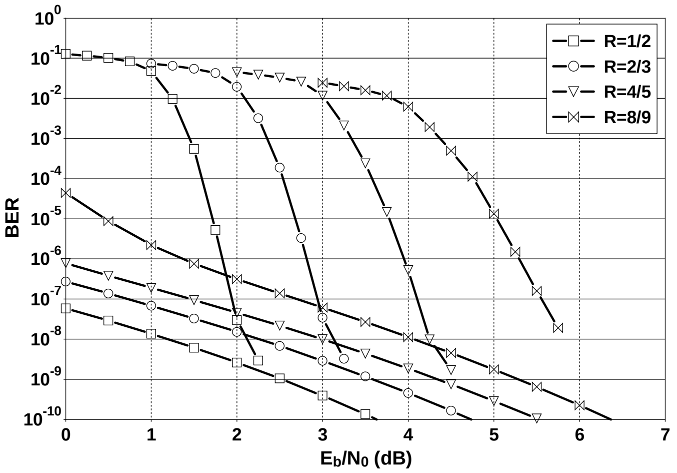

Fig. 7. BER performance of GCC with 8-state constituent codes. (ρ_1, ρ_2) are optimized for the WF region. $K = 2400$ bits, 10 iterations, random interleavers.

with $R_\pi = 2/3$ for several code rates ranging from $1/2$ to $8/9$. $K = 2400$ bits, random interleavers, and 10 decoding iterations are assumed. The codes are optimized for the WF region. Good performance is obtained for all rates in both the EF and WF regions. Moreover, it is observed that increasing the code rate only leads to a higher decoding threshold and not to a higher EF. This *flat* behavior stems from the fact that by moving the heavy puncturing from the outer encoder to the inner encoder, the interleaver gain for low rates is also kept for high rates, leading to a low and similar floor for all code rates. For comparison, we report in the same figure the performance of the 8-state PCCC with polynomials $(15/13)_8$ in [16] and that of a standard SCCC built from the concatenation of an 8-state outer code punctured to the suitable rate and an accumulator. The outer encoder is the $(15/13)_8$ encoder. In both cases zero-termination was used. Note that the SCCC code is not rate-compatible. As compared to the more complex PCCC, the proposed GCC has better performance in the error floor region for all code rates. On the other hand the GCC and the standard SCCC give almost similar performance for $R = 2/3$, while for $R = 1/2$ the standard SCCC shows a significant convergence loss. However, this code is expected to have a low EF, due to the high distance of the outer code. For higher rates, the standard SCCC encounters an error floor, due to the heavy puncturing of the outer code. Finally, we also report in the same figure the performance of the serially concatenated codes in [6]. The GCC code compares favorably in the EF with respect to the code in [6]. This is due to the fact that the structure considered here is more general than the one in [6]. Therefore, since the puncturing patterns are optimized to minimize the EF, better performance are expected in this region. Unfortunately, a slight loss in convergence is observed.

Finally, in Fig. 7 we give BER results for GCCs with 8-state constituent encoders with generator polynomials $(15/13)_8$. $R_\pi = 2/3$ and the puncturing ratios (ρ_1, ρ_2) are optimized for the WF region. In the figure we also plot the bound to the error probability to enlighten the error floor. Compared to 4-state GCCs in Fig. 6, 8-state GCCs achieve lower error floors thanks to a higher minimum distance, at a expense of a slight loss ($\sim 0.2 - 0.25$ dB for all rates) in the WF region.

VII. Conclusions

In this paper, a general and flexible code structure, generalizing parallel and serially concatenated codes was introduced. Bounds for the error performance in the EF region were provided together with design considerations of the constituent codes including their rate-compatible puncturing patterns. EXIT chart analysis was applied to predict the decoding threshold in the WF region. The bounds and the decoding thresholds were then combined such that proper design of the concatenated code can be performed, yielding good performance in both the EF and the WF region over a broad range of code rates.

Acknowledgment

The authors wish to thank the anonymous reviewers for the careful review of our paper and for helpful suggestions that improved the quality of the manuscript.

References

[1] F. Babich, G. Montorsi, and F. Vatta, “Partially systematic rate-compatible punctured SCCCs,” *IEEE Commun. Lett.*, vol. 8, no. 4, pp. 241–243, Apr. 2004.

[2] ——, “Design of rate-compatible punctured serial concatenated convolutional codes,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2004, pp. 552–556.

[3] A. Graell i Amat, G. Montorsi, and F. Vatta, “Analysis and design of rate compatible serial concatenated convolutional codes,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Sep. 2005, pp. 607–611.

[4] ——, “Design and performance analysis of a new class of rate compatible serially concatenated convolutional codes,” *IEEE Trans. Commun.*, vol. 57, no. 8, pp. 2280–2289, Aug. 2009.

[5] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, “Serial concatenation of interleaved codes: Performance analysis, design, and iterative decoding,” *IEEE Trans. Inform. Theory*, vol. 44, no. 5, pp. 909–926, May 1998.

[6] A. Graell i Amat, F. Brännström, and L. K. Rasmussen, “On the design of rate-compatible serially concatenated convolutional codes,” *European Trans. Telecommun.*, vol. 18, no. 5, pp. 519–527, Aug. 2007.

[7] F. Brännström, A. Graell i Amat, and L. K. Rasmussen, “A general structure for rate-compatible concatenated codes,” *IEEE Commun. Lett.*, vol. 11, no. 5, pp. 437–439, May 2007.

[8] Y. Zhang and W. E. Ryan, “Structured IRA codes: Performance analysis and construction,” *IEEE Trans. Commun.*, vol. 55, no. 5, pp. 837–844, May 2007.

[9] J. Hagenauer, “Rate-compatible punctured convolutional codes (RCPC codes) and their applications,” *IEEE Trans. Commun.*, vol. 36, no. 4, pp. 389–400, Apr. 1998.

[10] J. Freudenberger, M. Bossert, and S. Shavgulidze, “Partially concatenated convolutional codes,” *IEEE Trans. Commun.*, vol. 52, no. 1, pp. 1–5, Jan. 2004.

[11] S. Benedetto and G. Montorsi, “Unveiling turbo codes: some results on parallel concatenated coding schemes,” *IEEE Trans. Inform. Theory*, vol. 42, no. 3, pp. 409–428, Mar. 1996.

[12] S. ten Brink, “Convergence behaviour of iteratively decoded parallel concatenated codes,” *IEEE Trans. Commun.*, vol. 49, no. 10, pp. 1727–1737, Oct. 2001.

[13] F. Brännström, L. K. Rasmussen, and A. J. Grant, “Convergence analysis and optimal scheduling for multiple concatenated codes,” *IEEE Trans. Inform. Theory*, vol. 51, no. 9, pp. 3354–3364, Sep. 2005.

[14] S. ten Brink, *Design of Concatenated Coding Schemes Based on Iterative Decoding Convergence*. Stuttgart, Germany: Ph.D. Thesis, University of Stuttgart, 2001.

[15] F. Babich, A. Crismani, and R. G. Maunder, “Exit chart aided design of periodically punctured turbo codes,” *Electron. Lett.*, vol. 46, no. 14, pp. 1001–1003, Jul. 2010.

[16] F. Babich, G. Montorsi, and F. Vatta, “Some notes on rate-compatible punctured turbo codes (RCPTC) design,” *IEEE Trans. Commun.*, vol. 52, no. 5, pp. 681–684, May 2004.