Path Pruning for Unequal Error Protection Turbo Codes

Werner Henkel and Neele von Deetzen
International University Bremen (IUB)
Campus Ring 1
D-28759 Bremen, Germany
Email: {w.henkel, n.vondeetzen}@iu-bremen.de

***Abstract*— For some applications in communications, it is desirable to provide unequal error protection for different types of information data. In image coding, e.g., types of bits exist which have to be protected with different protection levels when transmitting these data over noisy channels. In this paper, we concentrate on Turbo codes ([1]) with unequal error protection capabilities which are achieved by modifying the code rate $R = k/n$. After pointing out that puncturing and path pruning are modifications of the rate in the denominator and numerator, respectively, the latter is studied in more detail. Design rules are given and EXIT charts for performance analysis of Turbo codes with path-pruned convolutional component codes are presented.**

I. Introduction

The usual way to adapt the rate and distance of a convolutional code based on a given mother code is puncturing leading to the so-called rate-compatible convolutional codes [2]. Output bits are omitted according to some puncturing pattern, thereby influencing the denominator of the rate $R = k/n$, i.e., modifying the number of output bits. These punctured codes are also used in Turbo coding schemes (see, e.g., [3], [4]) to realize unequal error protection (UEP) properties.

For rate changes, of course, the numerator may be modified as well. When doing this, paths are pruned from the trellis. Note that k defines the number of paths (2^k in the binary case) emerging from a state and merging into a state. Interestingly, not much work has been done on this, e.g., [5].

Multilevel coded modulation is an example where necessarily unequal error protection is required for the different levels. When additionally requiring rotational invariance properties, containment conditions have to be fulfilled, especially the first code has to be contained in the second code, etc.. In [6], multilevel convolutionally encoded modulation was studied and design rules were set up to ensure rotational invariance by suitable choices of the generators of the component convolutional codes.

From [6], we extract the design proposals and study them now in the light of unequal error-protecting Turbo codes. In the following section, we provide design methods for path-pruned convolutional codes together with an example. Thereafter, we provide some EXIT charts to show the effect of the path pruning. A list of codes is not provided in here, but a few examples are given in [6]. However, note that there the all-ones sequence was additionally required to be part of the code. This ensures that an inversion of all bits still leads to a valid code sequence. One may anyway want to preserve this property.

II. Path Pruning

In this section we show how to determine sub-codes, i.e., path-pruned codes from a given mother code or vice versa. To ensure sub-code properties, we have to fulfil

$$\forall_{I^{(s)}} \exists_{I^{(m)}} : I^{(s)} \cdot G^{(s)}(D) = I^{(m)} \cdot G^{(m)}(D) \,, \tag{1}$$

i.e., $\mathcal{C}^{(s)} \subset \mathcal{C}^{(m)}$, where $\mathcal{C}^{(s)}$ and $\mathcal{C}^{(m)}$ are the sub-code and the mother code, respectively, and $I^{(\cdot)}$ and $G^{(\cdot)}(D)$ are the information sequence and the code generator of the respective code.

Let us first assume the number of information bits of the sub-code to be $k^{(s)} = 1$. For simplicity, the components of the input vector of the mother code

$$I^{(m)} = (I_1^{(m)}, I_2^{(m)}, \, \dots \, , I_{k^{(m)}}^{(m)}) \tag{2}$$

are chosen to be combinations of shifted versions of $I^{(s)}$, i.e.,

$$I_l^{(m)} = I^{(s)} \cdot g_l(D) \,, \tag{3}$$

where $g_l(D)$ describes a polynomial of degree $\deg(g_l(D)) = j_l$. The $k^{(m)}$ polynomials $g_l(D)$ can now be arranged in a vector $G^{(p)}(D)$ of length $k^{(m)}$ with polynomial entries. This vector is assumed to be delayfree [8].
We can now write Eq. (3) in vector notation not only for the single components $I_l^{(m)}$ but for the whole $I^{(m)}$.

$$I^{(m)} = I^{(s)} \cdot G^{(p)}(D) \tag{4}$$

If we now insert this into Eq. (1) we obtain

$$\begin{aligned} I^{(s)} \cdot G^{(s)}(D) &= I^{(m)} \cdot G^{(m)}(D) & (5)\\ &= I^{(s)} \cdot G^{(p)}(D) \cdot G^{(m)}(D) & (6)\\ \Rightarrow G^{(s)}(D) &= G^{(p)}(D) \cdot G^{(m)}(D) \,. & (7) \end{aligned}$$

Now we consider the case of $k^{(s)} \geq 1$. Likewise, one chooses

$$I_l^{(m)} = \sum_{i=1}^{k^{(s)}} I_i^{(s)} \cdot g_{il}(D) \,, \tag{8}$$

$$\text{where} \quad l = 1, \, \dots \, , k^{(m)} \,, \quad i = 1, \, \dots \, , k^{(s)} \,.$$

1-4244-0092-9/06/$20.00

This means that the single components of the input to the mother encoder consist of combinations of the (shifted) inputs to the sub-code. Again, this expression can more easily be described by a matrix $G^{(p)}(D)$, now of dimensions $[k^{(s)} \times k^{(m)}]$ according to

$$I^{(m)} = I^{(s)} \cdot G^{(p)}(D) \; , \tag{9}$$

where $G^{(p)}(D)$ is of the following shape:

$$G^{(p)}(D) = \begin{bmatrix} g_{11}(D) & \dots & g_{1k^{(m)}}(D) \\ \vdots & & \vdots \\ g_{k^{(s)}1}(D) & \dots & g_{k^{(s)}k^{(m)}}(D) \end{bmatrix} . \tag{10}$$

Again, this implies Eq. (7):

$$G^{(s)}(D) = G^{(p)}(D) \cdot G^{(m)}(D) \; .$$

This means that we can construct different sub-codes from a given mother code by multiplying the mother code generator by generator matrices $G^{(p)}(D)$. From now on we will call this matrix *pruning matrix*, since it determines which state transitions in the trellis of the mother code will be pruned. In order to keep the complexity low, it is desirable to have the same trellises for the mother code and the sub-code except for the pruned state transitions and thus, the number of delay elements should be the same in both codes. Therefore, the pruning matrix has to fulfil the condition (for simplification, we write $G^{(\cdot)}$ instead of $G^{(\cdot)}(D)$)

$$\sum_{i=1}^{k^{(m)}} \max_{\substack{j \\ 1\le j\le n^{(m)}}} \deg(G^{(m)}_{i,j}) \overset{!}{=} \sum_{i=1}^{k^{(s)}} \max_{\substack{j \\ 1\le j\le n^{(m)}}} \deg(G^{(s)}_{i,j}) \tag{11}$$

$$= \sum_{i=1}^{k^{(s)}} \max_{\substack{j \\ 1\le j\le n^{(m)}}} \deg(G^{(p)}_{i,-} \odot G^{(m)}_{j,|}) \tag{12}$$

$$= \sum_{i=1}^{k^{(s)}} \max_{\substack{j \\ 1\le j\le n^{(m)}}} \deg(\sum_{k=1}^{k^{(m)}} g^{(p)}_{i,k} \cdot g^{(m)}_{k,j}) \; , \tag{13}$$

where $G^{(\cdot)}_{i,|}$ and $G^{(\cdot)}_{i,-}$ represent the ith column and the ith row of a matrix, respectively, and $\odot$ represents the scalar product of two vectors. The left and the right side of Eq. (11) represent the number of delay elements contained in the mother code and the sub-code, respectively, given the generator matrices. First, the number of delays per input stream is determined and then these are summed up over all input streams. Equations (12) and (13) take Eq. (7) into account and express the sub-code generator by the pruning code generator and the mother code generator in order to find a condition for $G^{(p)}(D)$ dependent on $G^{(m)}(D)$.

As an example, we choose

$$G^{(s)}(D) = \begin{pmatrix} 1+D+D^2 & 1+D+D^2 & 1+D^2 \end{pmatrix} \; . \tag{14}$$

Letting $I_1^{(m)} = I^{(s)}$ and $I_2^{(m)} = D \cdot I^{(s)}$ (and thus $G^{(p)}(D) = [1 \;\; D]$) leads to a set of solutions one of which is

$$G^{(m)}(D) = \begin{pmatrix} 1 & 1+D & 1+D \\ 1+D & D & 1+D \end{pmatrix} . \tag{15}$$

One may, of course, also define $G^{(m)}(D)$ and determine $G^{(s)}(D)$ for given constraint lengths $L^{(s)}$ and $L^{(m)}$, which would then be unique.

In [6], a few results of a computer search are listed.

From the described procedure, we obtain $C^{(s)}$ as a subcode of $C^{(m)}$ which means that state transitions and thus paths in the trellis (code sequences) are omitted (pruned) from the trellis of $C^{(m)}$ in a symmetric way. Let us view the trellises of the quoted example. In figures 1 and 2, the shift registers and the trellis diagrams of the mother and the sub-code are shown. $C^{(s)}$ is shown as solid lines as part of $C^{(m)}$ in Fig. 2. We present the recursive systematic versions of the generators from above, since we will use the convolutional codes inside Turbo codes. It is obvious, that the state transitions of the sub-code are contained in the trellis of the mother code.

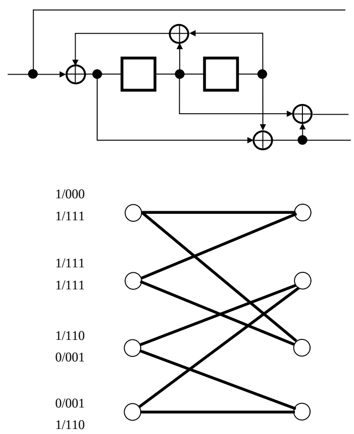

Fig. 1. Encoder and trellis of $C^{(s)}$

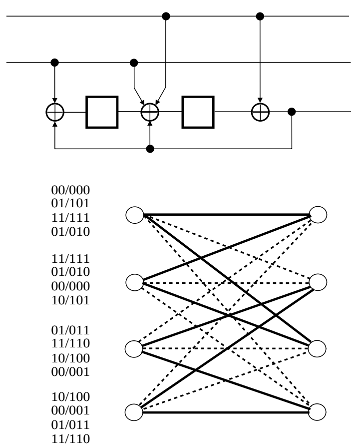

Fig. 2. Encoder and trellis of $C^{(m)}$; subcode $C^{(s)}$ marked in bold

These path pruned convolutional codes are now the basis for unequal error protection Turbo codes. We use the standard Turbo coding scheme shown in Fig. 3.

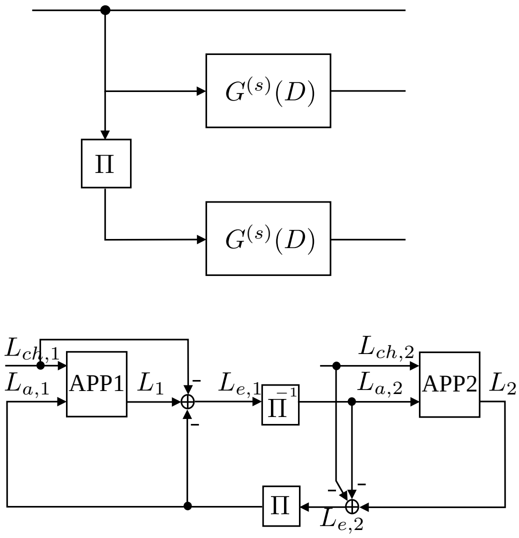

Fig. 3. Turbo encoder and decoder

Since the code rate of the constituent encoders, i.e., the sub-codes is

$$R^{(s)} = R^{(p)} \cdot R^{(m)} = \frac{k^{(p)}}{k^{(m)}} \cdot \frac{k^{(m)}}{n^{(m)}} = \frac{k^{(p)}}{n^{(m)}} \ ,$$

the overall code rate of the Turbo code is

$$\frac{1}{R_{TC}} = \frac{1}{R_1^{(s)}} + \frac{1}{R_2^{(s)}} - 1$$

or equivalently

$$R_{TC} = \frac{k^{(p)}}{n_1^{(m)} + n_2^{(m)} - k^{(p)}} \ ,$$

where the indices 1 and 2 represent the two constituent encoders, which have to have the same $k^{(p)}$ but not necessarily the same $n^{(m)}$. Furthermore, the rate of the pruned Turbo code has an upper and a lower bound given by

$$\frac{1}{n_1^{(m)} + n_2^{(m)} - 1} \leq R_{TC} \leq \frac{k^{(m)} - 1}{n_1^{(m)} + n_2^{(m)} - k^{(m)} + 1} \ , \qquad (16)$$

since $1 \leq k^{(s)} \leq k^{(m)}$ in order to assure a code rate smaller than 1 for the constituent codes.

Let us assume a frame of data containing several blocks with different importance levels and therefore different requirements in terms of bit error rates. With the above presented UEP Turbo codes we can encode these blocks successively, switching between different pruning codes according to rate and bit error rate requirements of the blocks. Let the mother code rate for example be $R^{(m)} = 4/5$ and the data frame consist of four blocks of increasing importance. One could then use the mother code without a pruning code or with a pruning code of rate $R^{(p,1)} = 1$ for the first block, and switch to a rate $R^{(p,2)} = 3/4$ pruning code for the second block. The third and the fourth block could be encoded by using pruning matrices of rate $R^{(p,3)} = 2/4$ and $R^{(p,4)} = 1/4$, respectively. Thus, the overall Turbo Code rates for the four blocks would be $2/3$, $3/7$, $1/4$, and $1/9$, which obviously leads to unequal error protection.

In the following section we show EXIT charts [7] and bit error rate curves to investigate the properties of the pruned codes.

III. EXIT Charts

In this section, we show by means of another example that the mother code and the sub-code have different correction capabilities and therefore are suitable for UEP. In our simulations, we set the frame length to 32768 bits. Both recursive systematic encoders have $k^{(m)} \cdot m^{(m)} = k^{(s)} \cdot m^{(s)} = 2$ memory elements. The generators of rate $R^{(s)} = 1/3$, $R^{(p)} = 1/2$ and $R^{(m)} = 2/3$ are

$$G^{(s)}(D) = \begin{pmatrix} 1 + D^2 & D + D^2 & 1 + D + D^2 \end{pmatrix} \ , \qquad (17)$$

$$G^{(p)}(D) = \begin{pmatrix} 1 + D & D \end{pmatrix} \ , \qquad (18)$$

and

$$G^{(m)}(D) = \begin{pmatrix} 1 & D & 1 \\ 1 + D & 0 & D \end{pmatrix} \ . \qquad (19)$$

Our simulations were done for an AWGN channel and we used a Log-MAP algorithm for decoding.

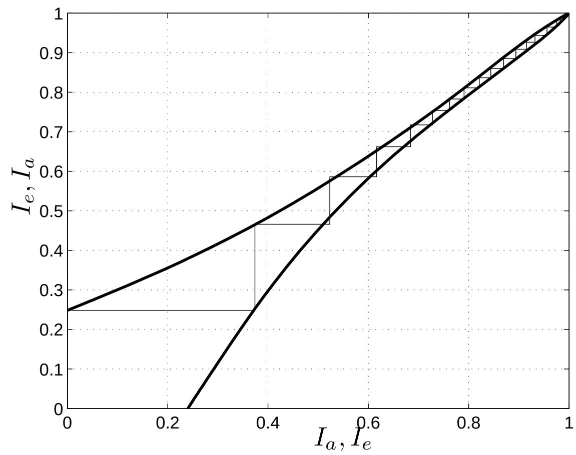

Fig. 4. EXIT chart of the sub-code $C^{(s)}$ at $E_b/N_0 = 0.1$ dB

Figure 4 shows the EXIT chart for the sub-code $C^{(s)}$ at a signal-to-noise ratio of 0.1 dB, where the decoder converges and the a priori and the extrinsic mutual information, is close to 1. In Figure 5, we see the EXIT chart of the mother code at two different signal-to-noise ratios. The solid curves show the mutual information for $E_b/N_0 = 1.8dB$ dB, where the mother code just converges to the point $(1; 1)$. The dotted curves represent the mutual information for $E_b/N_0 = 0.1dB$ dB, i.e., the signal-to-noise ratio for which the sub-code converges. This is to show that the mother code still has very bad performance while the corresponding sub-code already converges.

Figure 6 shows the bit error rate curves of the mother code and the sub-code over E_b/N_0 for a frame length of 8192 bits,

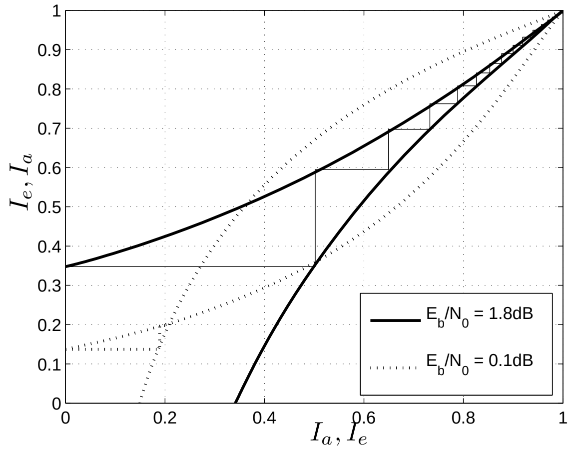

Fig. 5. EXIT chart of the mother code $C^{(m)}$ at $E_b/N_0 = 1.8$ dB and $E_b/N_0 = 0.1$ dB

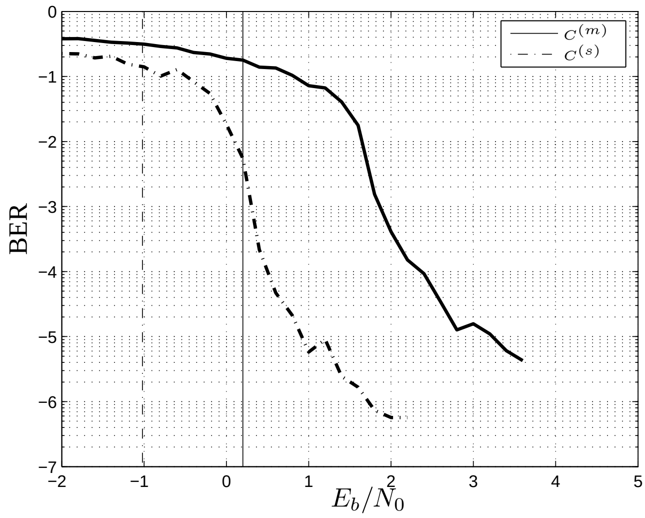

Fig. 6. Bit error rate curves of $C^{(m)}$ and $C^{(s)}$ over E_b/N_0

where the solid and the dashed curve represent the mother code and the sub-code, respectively. Additionally, we inserted the Shannon limits for the respective codes. As we expected, the performance of the mother code is worse than that of the sub-code and there is a difference of approximately 1.7 dB between the two curves.

Since the transmit power of a code word is usually constant, regardlass of the code rate, the bit error rate curves over E_s/N_0 might be more suitable. These are shown in Figure 7. We see a difference between the performances of around 5.5 dB.

Finally, we should mention that the examples shown in this paper are useful for applications that do not intend to approach the Shannon limit too closely. These codes have not yet been optimised concerning distance properties and shapes of EXIT

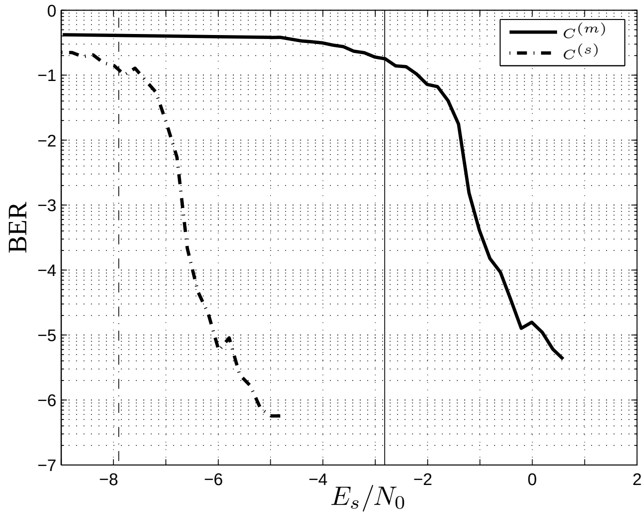

Fig. 7. Bit error rate curves of $C^{(m)}$ and $C^{(s)}$ over E_s/N_0

curves and thus, better results might be achieved when taking into account the influence of the pruning code on the distance spectrum and the EXIT curves of the sub-code.

IV. Conclusions

We have shown that a pruning procedure originally designed for multilevel coded modulation is a worthwhile and readily available alternative to puncturing to adapt the rate and distance for different protection levels in UEP Turbo codes. Pruning can simply be accomplished by a concatenation of the mother code and a pruning code which leads to a selection of only some paths in the trellis. EXIT charts and bit error rate curves of an exemplary code have been presented.

Acknowledgement

This work is part of the FP6 / IST project M-Pipe and is co-funded by the European Commission.

References

[1] Berrou, C., Glavieux, A., Thitimajshima, P., "Near Shannon limit error-correcting coding and decoding: turbo codes", *Proc. IEEE International Conference on Communication (ICC)*, May 1993, Geneva, Switzerland, May 1993, pp. 1064-1070.

[2] Hagenauer, J.: "Rate-Compatible Punctured Convolutional Codes (RCPC Codes) and their Applications", *IEEE Trans. on Comm.*, Vol. 36, No. 4, April 1988, S. 389-400.

[3] Caire, G., Lechner, G., "Turbo Codes with unequal error protection," *Electronics Letters*, March 28, 1996, Vol. 32, No. 7, pp. 629-631.

[4] Barbulescu, A.S., Pietrobon, S.S., "Rate compatible turbo codes," *Electronics Letters*, Vol. 31, March 1995, pp. 530-537.

[5] Wang, C.-H., Chao, C.-C., "Path-Compatible Pruned Convolutional (PCPC) Codes: A New Scheme for Unequal Error Protection", *ISIT*, Cambridge, MA, USA, Feb.1998.

[6] Koch, M., Henkel, W., "90°-Rotationally Invariant Multilevel Convolutionally Encoded QAM," *ETT*, Vol. 4, No. 2, March/April 1993, pp. 25-31.

[7] ten Brink, S., "Convergence Behavior of Iteratively Decoded Parallel Concatenated Codes," *IEEE Trans. on Communications*, Vol. 49, No. 10, pp. 1727-1737, Oct. 2001.

[8] Johannesson, R., Wan, Z.-X., "A Linear Algebra Approach to Minimal Convolutional Encoders," *IEEE Trans. on Information Theory*, Vol. 39, Issue 4, pp. 1219 - 1233, July. 1993.