An Adaptive and Complexity Reduced Decoding Algorithm for Convolutional Codes and its Application to Digital Broadcasting Systems

Jan Geldmacher, Klaus Hueske and Jürgen Götze
Information Processing Lab,
Department of Electrical Engineering and Information Technology,
TU Dortmund University, Otto-Hahn-Strasse 4, 44227 Dortmund, Germany.
Correspondence: jan.geldmacher@tu-dortmund.de

***Abstract*—Convolutional codes are still prevalent in digital broadcasting systems, like DVB-T, DVB-H, DAB, DMB and DRM. This paper presents an adaptive and complexity reduced decoding algorithm for CCs. The proposed algorithm is based on the earlier proposed block syndrome decoder (BSD) and features a significant reduction of decoding effort depending on the current channel conditions. Additionally it is capable of adapting its output bit error rate to a given threshold to achieve a further reduction of decoding complexity. For this purpose, an SNR estimation is incorporated to achieve an adaption of decoding performance to a given bit error rate threshold in dynamic SNR environments. This is especially useful for the widely used serially concatenated coding schemes, which is demonstrated by means of the terrestrial DVB system.**

***Index Terms*—Convolutional Codes, Syndrome Decoding, Serial Concatenation, Adaptive Decoding, DVB**

I. Introduction

Convolutional codes (CC) are still widely used as forward error correction (FEC) codes in today's broadcasting systems, e.g. DVB-T, DVB-H, DAB, DMB and DRM. Recently receivers for these broadcasting systems are also integrated into battery powered devices, where the power consumption of the signal processing algorithms is a governing factor. Considering this, complexity reduction in baseband signal processing algorithms is an important issue.

One of the tasks with highest computational complexity in baseband processing is FEC decoding using the Viterbi Decoder (VD) [1]. The error correction capabilities are improved for a longer memory length of the CC, however, the decoding complexity in terms of trellis states grows exponentially with the memory length: Better codes lead to higher decoding effort. Furthermore, the VD cannot benefit from good channel conditions, i.e. its complexity does not depend on the number of errors actually corrected.

A modification of the Viterbi algorithm (VA) that tackles these problems is the T-algorithm [2]. The T-algorithm reduces the number of trellis paths by only keeping those paths, which are better than a predefinded threshold regarding their metrics. This does not only reduce the number of required add-compare-select (ACS) operations, but also leads to an adaption of decoding complexity to channel conditions: A bad channel with a large number of errors to be corrected requires a higher number of trellis paths to be considered, while good channel conditions result in larger metric deviations and a smaller number of paths.

In practical systems, the question arises, which bit-error-rate (BER) is actually required after the VD: In case of serially concatenated coding schemes with the CC being used as the inner code, as found in DVB-T, DVB-H and DMB, this question can be answered by analyzing the performance of the outer decoder. The latter will deliver "quasi-error-free" (QEF) outputs if its input BER is smaller than a certain lower threshold and it will not improve its output BER, but even increase it, if its input BER is greater than a certain upper threshold. These lower and upper BER thresholds are termed LBER and UBER in the following. The consequence of this observation is, that the decoding effort of the inner decoder can be reduced in two cases: Firstly, if the output BER is worse than the UBER. In this case the inner decoder can be switched off, because the outer decoder's output is useless anyway. This can reduce energy consumption for temporary bad reception conditions, e.g. when passing a tunnel. Secondly, if the output BER is better than the LBER. In this case the decoder can reduce its decoding effort and ideally adapt its output BER to the LBER. This can reduce energy consumption under good reception conditions. A decoder, which implements this idea, needs a method both to determine the current SNR (and with it its current output BER) and to adaptively increase its output BER up to a given threshold in favor of the decoding effort.

The T-algorithm can be used to implement this behaviour: In [3] the authors propose the adaption of the threshold and the truncation length, depending on SNR estimates. This leads to trade-off between decoding effort and error correction capability.

In this work another approach to adaptive decoding, based on the block syndrome decoder (BSD), which was previously considered for efficient parallel decoding [4], is presented. Compared to the T-algorithm, this approach can reduce its effort to almost zero in the best case and can use a standard implementation of the VA.

This paper is organized as follows: In Section II the basic

principles of the BSD for convolutional codes are explained. This includes the background on syndrome decoding, its extension to punctured convolutional codes and the segmentation of the received sequence using the syndrome sequence. If used in transmission systems that require a specific BER and do not benefit from further BER reduction, the complexity of the block syndrome decoder can be further reduced by introducing an adaptive behavior. This adaption to a specific BER is described in Section III. The achievable BER performance and complexity reduction of the proposed method are investigated using simulations. The simulation results are presented in Section IV. Conclusions are drawn in Section V.

In the following matrices and vectors are polynomial in D, where D is a delay operator. They are denoted by bold upper (e.g. $\mathbf{G}$) and lower case (e.g. $\mathbf{b}$) letters, respectively. Their corresponding time domain matrices and vectors are underlined, e.g. $\underline{\mathbf{b}}$. The operator $\oplus$ denotes addition in $GF(2)$.

II. Block Syndrome Decoder for Convolutional Codes

A. Syndrome Decoding of Convolutional Codes

The syndrome decoder has been proposed by Schalkwijk et al. as an alternative maximum likelihood (ML) decoder for CCs [5]. In contrast to the VD, which estimates the ML *code* sequence for a given received sequence, the syndrome decoder computes the most likely *error* sequence. This error sequence is then used to correct the transmission errors.

Let us first briefly review the syndrome decoding algorithm: Given a k/n-rate convolutional code with $k \times n$-generator matrix $\mathbf{G}$, the encoding and transmission of an information sequence represented by a $1 \times k$ vector $\mathbf{u}$ can be written as

$$\mathbf{r} = \mathbf{uG} \oplus \mathbf{e} = \mathbf{v} \oplus \mathbf{e}, \tag{1}$$

where the $1 \times n$ vectors $\mathbf{v}$, $\mathbf{e}$ and $\mathbf{r}$ represent encoded sequence, transmission error and received sequence, respectively. Defining an $n \times (n-k)$ matrix $\mathbf{H}^T$, which is orthogonal to the generator matrix, $\mathbf{GH}^T = \mathbf{0}$, a $1 \times (n-k)$ vector $\mathbf{b}$ can be calculated as follows:

$$\mathbf{b} = \mathbf{rH}^T = \mathbf{uGH}^T \oplus \mathbf{eH}^T = \mathbf{eH}^T \tag{2}$$

The vector $\mathbf{b}$ is denoted as the syndrome of the transmission error, as it solely depends on the transmission error $\mathbf{e}$. The matrix $\mathbf{H}^T$ is therefore called the syndrome former. In ML decoding the code sequence with minimum distance to the received sequence has to be computed. This is equivalent to computing the error sequence $\hat{\mathbf{e}}$ with minimum Hamming weight, that at the same time yields that $\mathbf{r} \oplus \hat{\mathbf{e}}$ is a valid codeword. The latter condition can be written as

$$(\mathbf{r} \oplus \hat{\mathbf{e}})\,\mathbf{H}^T = \mathbf{0} \Leftrightarrow \mathbf{rH}^T = \hat{\mathbf{e}}\mathbf{H}^T = \mathbf{b}. \tag{3}$$

Thus the decoding problem can be formulated as

$$\hat{\mathbf{e}} = \arg\min_{\mathbf{e}} \left\{ \| \mathbf{e} \| \;\middle|\; \mathbf{eH}^T = \mathbf{b} \right\}, \tag{4}$$

where the operator $\| \cdot \|$ denotes a suitable metric, e.g. the Hamming metric for hard decision decoding. The optimization problem (4) can be solved by using the Viterbi algorithm (VA) to search the trellis of the syndrome former corresponding to $\mathbf{b}$ for the path with minimum weight. For soft decision decoding the VA's branch metrics at time instant t can be computed as

$$\sum_{i=0}^{n-1} \underline{e}_i^{(t)} |\underline{r}_i^{(t)}|, \tag{5}$$

where $\underline{e}_i^{(t)}$ and $\underline{r}_i^{(t)}$ denote i-th error bit and received soft bit in time domain, respectively. In terms of ACS operations solving (4) is of the same complexity as the Viterbi decoding problem, because the trellis complexities of $\mathbf{G}$ and $\mathbf{H}^T$ are identical for minimal realizations [6]. After solving (4), the ML information sequence is computed as

$$\hat{\mathbf{u}} = (\mathbf{r} \oplus \hat{\mathbf{e}})\,\mathbf{G}^{-1}, \tag{6}$$

where $\mathbf{G}^{-1}$ is the right inverse of $\mathbf{G}$.

For a given code, the matrices $\mathbf{G}^{-1}$ and $\mathbf{H}^T$ can be calculated from the invariant factor decomposition (IFD)[7] of the generator matrix. The IFD of $\mathbf{G}$ is given as

$$\mathbf{G} = \mathbf{A\Gamma B}, \tag{7}$$

where $\mathbf{A}$ and $\mathbf{B}$ are invertible scrambler matrices with unit determinant. The inverse of $\mathbf{B}$ can be partitioned as

$$\mathbf{B}^{-1} = \left(\mathbf{G}_b^{-1}\ \mathbf{H}^T\right). \tag{8}$$

Thus the right inverse ${\mathbf{G}_\mathbf{b}}^{-1}$ of the equivalent generator matrix $\mathbf{G_b}$ of $\mathbf{G}$ and the syndrome former $\mathbf{H}^T$ can be identified as the first k columns and the last $n-k$ columns of $\mathbf{B}^{-1}$, respectively. The right inverse of $\mathbf{G}$ can be computed from $\mathbf{G}_b^{-1}$ following

$$\mathbf{G}^{-1} = \mathbf{G}_b^{-1}\mathbf{A}^{-1}, \tag{9}$$

where a basic encoder is assumed, i.e. $\mathbf{\Gamma} = (\mathbf{I}_k\ \mathbf{0})$. For $k = 1$ the special case $\mathbf{G}^{-1} = \mathbf{G}_b^{-1}$ holds.

In most digital broadcasting scenarios, like in the DVB-T or DVB-H systems considered in this work, punctured convolutional codes (PCCs) are used. By puncturing codes with higher code rate can be constructed from lower rate (usually rate $1/n$) mother codes using certain puncturing patterns. This is realized by deleting specific bits from the encoded data sequence prior to transmission according to the chosen puncturing pattern. PCCs can be decoded by the VD with the same complexity as the mother code, by simply inserting zeros into the received sequence according to the puncturing pattern. However, the insertion of zeros is not directly applicable to SD: Syndrome decoding depends on the orthogonality of the transmitted code sequence to the syndrome former $\mathbf{H}^T$, which does not hold for a punctured code sequence and the syndrome former of the mother code.

Nevertheless, the SD is capable of decoding the PCCs used in this work with same complexity as the VD: This is achieved by constructing a suitable trellis representation for each PCC. The first step is to include the puncturing pattern of rate $(n-1)/n$ into the generator matrix (cf. [8]), which results in the $(n-1) \times n$ generator matrix $\mathbf{G}_p$ of the PCC. Using the IFD the corresponding $n \times 1$ syndrome former $\mathbf{H}_p^T$ for this

PCC can be computed. In the next step the trellis for $\mathbf{H}_p^T$ is constructed, which has 2^{n-1} edges merging into each trellis state. To reduce the complexity of the trellis one section of the syndrome former trellis is divided into $n-1$ sections with only 2 edges merging into each trellis state. Using this trellis together with the syndrome former $\mathbf{H}_p^T$, the SD can decode the PCCs used in this paper with the same complexity as the VD. A detailed description of this procedure is beyond the scope of this paper; a similar approach for trellis reduction is described in [9].

B. Block Syndrome Decoder

The syndrome decoding algorithm offers an interesting property for reduction of decoding effort and for parallel decoding [4]: As mentioned earlier the syndrome sequence $\mathbf{b}$ only depends on the the channel error and not on the transmitted code sequence (cf. (2)). This means that error-free parts in $\underline{\mathbf{r}}$ will lead to zero sequences in $\underline{\mathbf{b}}$, due to the finite influence length of $\mathbf{H}^T$. By analyzing $\underline{\mathbf{b}}$ and identifying parts of zeros with sufficient length, error-free parts in the received sequence $\underline{\mathbf{r}}$ can be identified. Knowing error-free parts allows to separate $\underline{\mathbf{r}}$ into blocks that need to be decoded and blocks that are error-free and hence do not need any decoding. Additionally, for the blocks to be decoded, the initial and the final trellis states are known to be the "zero-state" of the trellis [4], which means the blocks are terminated and can be decoded independently.

To specify how exactly a sequence is partitioned, three design parameters are defined: Firstly a parameter named ℓ_{min} specifies how many consecutive zeros in the syndrome sequence are required to consider a part of $\underline{\mathbf{r}}$ to be error-free. Additionally two parameters ℓ_{on} and ℓ_{off} are defined to specify how many zeros are included in the block to be decoded. In other words, for each block in $\underline{\mathbf{b}}$ with $i \geq \ell_{min}$ consecutive zeros, the corresponding

$$(i - (\ell_{on} + \ell_{off}))\frac{n}{n-k} \tag{10}$$

bits in the received sequence are directly sent to the output and not treated by the Viterbi algorithm. Fig. 1 illustrates the meaning of the parameters.

The selection of the design parameters depends on the used code and is a trade-off between decoding effort and decoding performance. The longer ℓ_{off} is chosen, the higher the probability that the ML path has merged with the zero path of the trellis. The longer ℓ_{on} is chosen, the higher the probability that the ML path of the considered block starts in the zero state. In general, the smaller the value for ℓ_{min} is chosen, the higher the reduction of decoding effort and the lower the decoding performance in terms of bit error rate (BER). On the other hand, for larger values for ℓ_{min}, both decoding effort and performance of the BSD converge towards the standard VD. Generally the choice of parameters also depends on the underlying code, where better codes in terms of constraint length and free distance require larger values. Suitable parameter settings with reasonable decoding performance and effort can be determined by carrying out BER simulations for different $(\ell_{on}, \ell_{off}, \ell_{min})$ values for the used code.

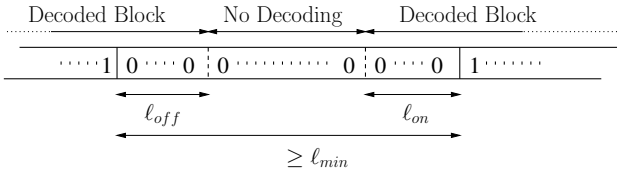

Fig. 1. Syndrome sequence and meaning of BSD parameters.

In summary the BSD algorithm consists of the following steps:

1. Compute the syndrome sequence $\mathbf{b} = \mathbf{r}\mathbf{H}^T$, where $\mathbf{H}^T$ is a syndrome former of the code.
2. Identify error-free blocks in $\underline{\mathbf{r}}$ by analyzing $\underline{\mathbf{b}}$: For every block in $\underline{\mathbf{b}}$ with $\geq \ell_{min}$ consecutive zeros the corresponding block in $\underline{\mathbf{r}}$ excluding the first ℓ_{off} and the last ℓ_{on} stages is considered error-free, see Fig. 1.
3. Apply the VA to the remaining erroneous blocks to solve (4) for every block, resulting in a correction sequence $\underline{\hat{\mathbf{e}}}^i$ for every block. Construct the complete correction sequence $\underline{\hat{\mathbf{e}}}$ by putting together the blocks $\underline{\hat{\mathbf{e}}}^i$, padded with zeros corresponding to each error-free block.
4. Compute the corrected ML information sequence $\hat{\mathbf{u}}$ by $\hat{\mathbf{u}} = (\mathbf{r} \oplus \hat{\mathbf{e}})\,\mathbf{G}^{-1}$.

The highest computational effort is required in step 3, because it involves the VA to search the trellis of $\mathbf{H}^T$ for $\hat{\mathbf{e}}$. In worst case, when no error-free segments are identified (i.e. bad SNR), the complexity is identical to the VD. However, in typical scenarios it is much lower as demonstrated in Section IV. Compared to step 3 the computational effort of steps 1, 2 and 4 is negligible: steps 1 and 4 can easily be realized by XOR-operations and step 2 requires a simple counter and a decision logic.

III. Adaptive Decoder

A. Basic Idea

Considering a serially concatenated coding scheme with inner convolutional code, as found for example in DVB physical layers, it is interesting to analyze which BER is actually required after the inner decoder. As already mentioned in Section I this can be done by analyzing the performance of the outer decoder: Firstly, the outer decoder will deliver QEF outputs if its input BER, which is identical to the inner decoder's output BER, is smaller than a certain lower threshold, termed LBER in the following. Secondly, the outer decoder will not improve its output BER, but even increase it, if its input BER is greater than a certain upper threshold. This upper output BER threshold of the inner decoder is termed UBER in this work.

From this observation we conclude, that a reduction of decoding effort of the inner decoder is feasible in two cases: Firstly, if the output BER is worse than the UBER. In this case the inner decoder could be switched off, because the outer

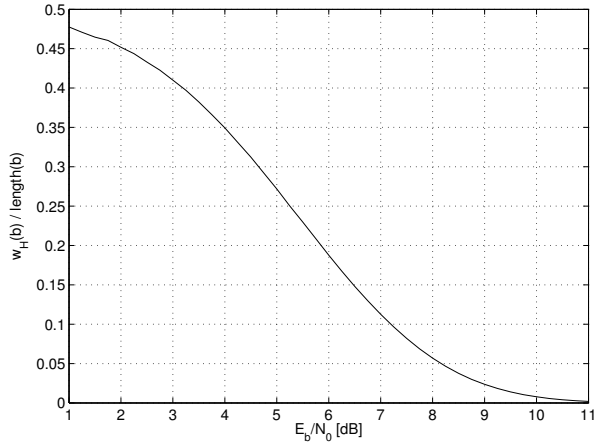

Fig. 2. Dependency between SNR and relative syndrome weight for the $1/2$-rate code with $\mathbf{G} = (171_8, 133_8)$.

decoder's output is useless anyway. Secondly, if the output BER is better than the LBER. In this case the decoder could reduce its decoding effort and ideally adapt its output BER to the LBER. A decoder, which implements this idea, needs a method both to determine the current SNR (current output BER) and to adjust its output BER in favor of the decoding effort.

The BSD's decoding effort adapts to the channel conditions while keeping the impact on the BER almost negligible by only decoding erroneous blocks of a received frame. To achieve an adaption of the output BER and with it an even higher reduction of the decoding effort, the design parameters can be adjusted depending on the current SNR. This requires that the decoder can determine its current output BER at any point. In context of syndrome decoding, this can be realized with minimum effort: Keeping in mind that the syndrome sequence reflects the errors in the received frame, it is easy to see that its Hamming weight $w_H(\underline{\mathbf{b}})$ can be used to estimate the current channel conditions in terms of number of transmission errors or SNR. The dependency of $w_H(\underline{\mathbf{b}})$ and SNR for a transmission over an AWGN channel is illustrated in Fig. 2.

The adaptive BSD can then be implemented by storing a set $(\ell_{min}, \ell_{on}, \ell_{off})$ for different values of $w_H(\mathbf{b})$ in a look-up-table (LUT). In step 2 of the BSD algorithm - instead of using fixed parameters - a parameter triplet $(\ell_{min}, \ell_{on}, \ell_{off})$ is selected depending on $w_H(\mathbf{b})$.

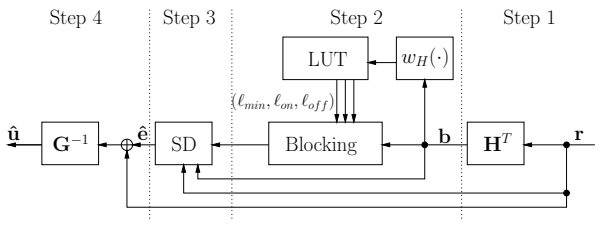

Fig. 3. Adaptive Block Syndrome Decoder.

Fig. 3 provides a schematic overview of the adaptive BSD. After $\mathbf{b}$ has been computed, in the second step the weight $w_H(\underline{\mathbf{b}})$ is calculated and the parameter triplet $(\ell_{min}, \ell_{on}, \ell_{off})$ is chosen accordingly. Using these parameters the erroneous and error-free blocks are determined, so that in step 3 only the erroneous blocks are decoded. Finally in step 4 the received sequence is corrected using the estimated error sequence and the resulting code sequence is mapped back to the according information sequence.

B. Application to DVB-T system

The efficiency of the proposed adaptive decoder is demonstrated for the terrestrial digital video broadcasting system (DVB-T) [10]. In this system the forward error correction is realized by the serial concatenation of an outer Reed-Solomon block code (RS code) and an inner CC. The RS code is a shortened (204,188) code and the CC is a $1/2$-rate code with constraint length $\nu = 6$ and $\mathbf{G} = (171_8, 133_8)$ or one of its PCCs with rate $R = 2/3$, $R = 3/4$, $R = 5/6$ or $R = 7/8$. A simplified scheme of the transmission system is depicted in Fig. 4.

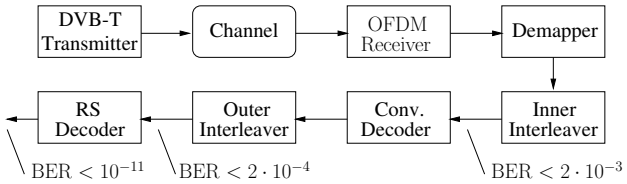

Fig. 4. Simplified scheme of DVB-T receiver and required BERs for quasi error-free reception.

The threshold LBER of the convolutional decoder output for a DVB-T system is given as $2 \cdot 10^{-4}$, because this corresponds to an output BER of the RS decoder of $< 10^{-11}$, which is defined as QEF output in the DVB-T standard [10]. The input/output BER of the (204,188) shortened RS code and its original (255,239) code is plotted in Fig. 5. Regarding to this BER plot the upper threshold UBER could be chosen as $\approx 4 \cdot 10^{-3}$ because for BERs higher or equal, the outer $(204, 188)$ RS decoder will only increase the overall BER. Further investigation on the MPEG-2 transport stream in [11] suggested that a transmission is not possible for a RS output BER $> 10^{-4}$, which corresponds to a convolutional decoder output BER of $2 \cdot 10^{-3}$. Thus UBER will be set to $2 \cdot 10^{-3}$.

Once the values for LBER and UBER are known, an LUT of the required parameters $(\ell_{min}, \ell_{on}, \ell_{off})$ for each SNR value (i.e. each Hamming weight $w_h(\underline{\mathbf{b}})$) can be constructed for each code rate. The parameters are chosen to achieve a BER smaller than LBER with maximum reduction of decoding effort. For the simulations presented in the following section LUTs with 20 to 30 entries have been used depending on the code rate. The spacing between the stored $w_H(\underline{\mathbf{b}})$ values corresponds to $0.25dB$ steps of the input E_b/N_0 of the convolutional decoder.

IV. Simulation Results

A. AWGN channel transmission

In this section simulation results for the proposed adaptive decoding algorithm are presented. The parameter values for

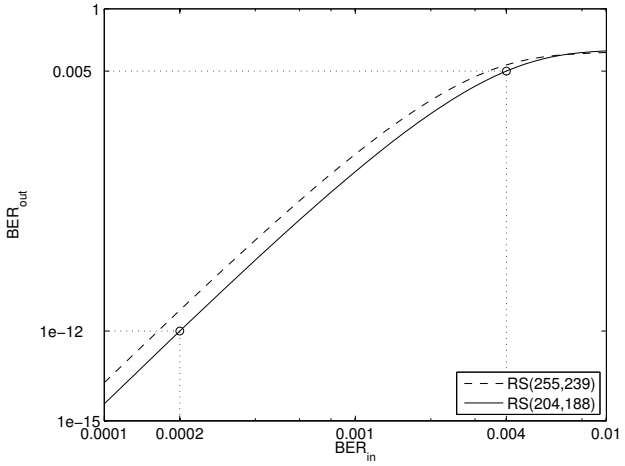

Fig. 5. BER of (255,239) and (204,188) RS code.

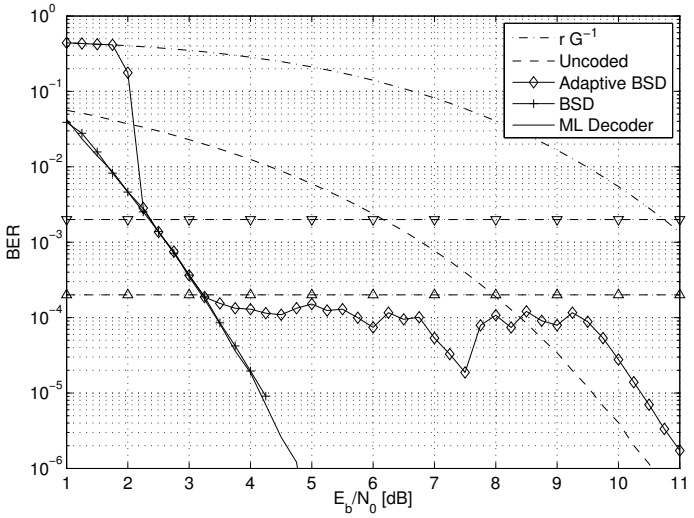

Fig. 6. Bit-Error-Rate versus E_b/N_0 for AWGN channel, $R = 1/2$.

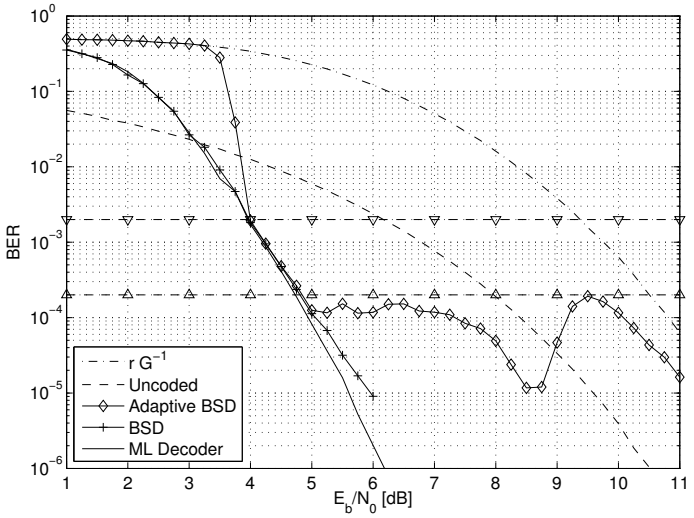

Fig. 7. Bit-Error-Rate versus E_b/N_0 for AWGN channel, $R = 5/6$.

the BSD with fixed parameters are given in Table I. The parameters have been selected such that the resulting BER of the BSD is identical to that of the ML decoder until the LBER is reached.

For the adaptive BSD, the parameters are identical to the fixed BSD until the LBER is reached. From the LBER on, the parameters values are successively decreased depending on the current SNR.

TABLE I
PARAMETER SETTINGS OF BSD FOR CODE WITH $\mathbf{G} = (171_8,\ 133_8)$ AND ITS PCCS.

Code rate R	$(\ell_{min},\ \ell_{on},\ \ell_{off})$
$1/2$	$(16,\ 7,\ 2)$
$2/3$	$(15,\ 6,\ 1)$
$3/4$	$(13,\ 6,\ 2)$
$5/6$	$(12,\ 6,\ 1)$
$7/8$	$(11,\ 6,\ 1)$

Before considering the DVB-T system, Fig. 6 and 7 show exemplary the BER of the CC with rate $R = 1/2$ and its PCC with $R = 5/6$ for a BPSK (binary phase shift keying) transmission over an AWGN (additive white gaussian noise) channel. Both figures present the BERs of the VD (denoted as "ML decoder"), the BSD with fixed parameter settings ("BSD") and the BSD with adaptive parameter settings ("Adaptive BSD"). The dashed line marks uncoded transmission, the dash-dotted marks coded transmission without decoding, i.e. the BER when directly feeding the received sequence into step 4 of the BSD algorithm. By modifying the design parameters $(\ell_{min}, \ell_{on}, \ell_{off})$ in step 3, the BER can be adjusted between the BER without decoding and the BER of the ML decoder. Additionally LBER and UBER are marked by horizontal lines with upright and reciprocal triangles, respectively.

From these figures one can see that the BER of the BSD with fixed settings is identical to that of the ML decoder up to an SNR that corresponds to the LBER. For higher SNR, the BER slightly deviates from the ML decoder, which, however, has no impact on the system performance as the QEF BER is already reached.

The BER of the adaptive BSD is identical to that of the BSD until LBER is reached. For higher SNR it strongly deviates from the BER of the ML decoder and approximates the LBER. This is achieved by successively decreasing the design parameters, which on the one hand approximates LBER and on the other hand reduces the decoding complexity as more zero sequences in the syndrome can be exploited. Again, the system performance is not influenced, as the BER is always below LBER, i.e. a QEF system output is guaranteed for this decoder.

From the figures one can notice that the approximation of the LBER is not perfect, i.e. the BER of the adaptive decoder does not always show a perfect convergence towards the LBER, but performs significantly better for some SNRs.

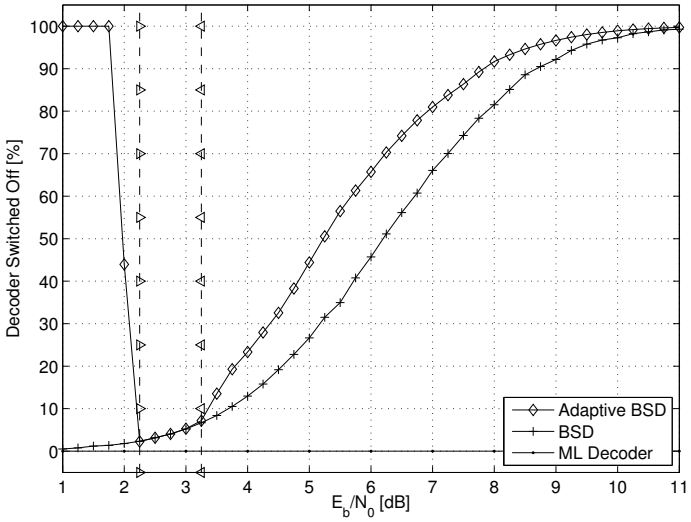

Fig. 8. Percentage the decoder is switched-off, $R = 1/2$.

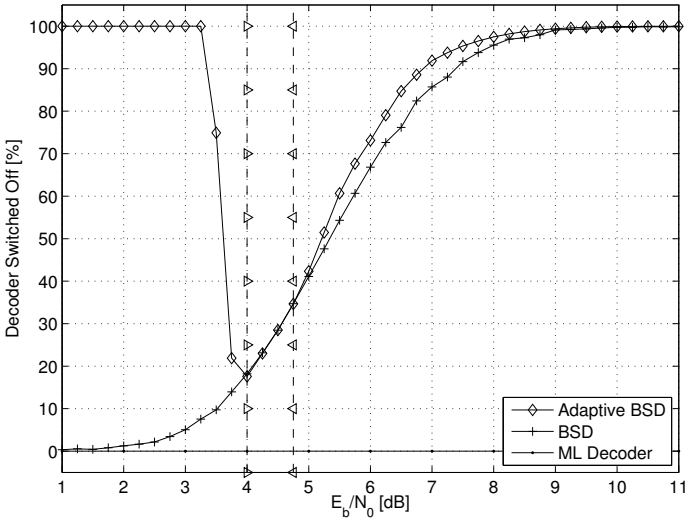

Fig. 9. Percentage the decoder is switched-off, $R = 5/6$.

This is the case around 7.5dB for the $1/2$-rate code and around 8.5dB for the $5/6$-rate code. The reason for this is that not for each SNR value an optimal parameter setting can be found, that results in a BER close to the LBER. However, as long as the resulting BER is below the LBER, this clearly has no impact on the performance.

As discussed in Section III, the decoder can be switched off, if its output BER is worse than the minimum required input BER of the block code, the UBER. This behavior can be seen in both figures, i.e. the BER converges towards that of $\mathbf{r}\mathbf{G}^{-1}$ if the SNR is worse than 2.25dB or 4dB for the $1/2$-rate and the $5/6$-rate code respectively. In this range step 3 is completely skipped and thus the computational most expensive part of the BSD algorithm, the Viterbi Algorithm, is not executed. Consequently, if we measure the decoding effort in terms of operations on the trellis[1], in this low SNR range the reduction of decoding effort is 100% compared to the VD.

Fig. 8 and 9 illustrate this reduction of decoding effort for both codes. It is obvious that the implementation of the BSD with fixed parameters already results in an adaptive decoder: Depending on the SNR, the BSD achieves a reduction of decoding effort of greater than 7% and greater than 34% for the $1/2$- and $5/6$-rate codes, respectively, if the system is in the QEF SNR range, marked by the right vertical line in these figures. The higher the SNR the higher the reduction of decoding effort, because the less errors occur the more and longer error-free blocks are identified in step 2. Additionally the figures show, that the reduction differs for both codes, because both codes use different design parameters (cf. Table I) and smaller parameter values allow for a greater reduction of decoding effort.

An implementation of the BSD with adaptive parameters decreases the decoding effort even more. The adaptive reduction of the design parameters results in an additional reduction of up to 20% for the $1/2$-rate code and 8% for the $5/6$-rate code depending on the SNR.

[1]The reduction is measured here as the percentage of received symbols which are not decoded, i.e. which do not require any operations of the VA in step 3 of the algorithm but are directly forwarded into step 4.

B. DVB-T physical layer transmission

Both decoders, the BSD and the adaptive BSD, have been implemented as inner decoders for a DVB-T physical layer as a replacement for the VD. The DVB-T system was simulated with QAM64 modulation and transmission over an AWGN channel. The receiver operated in 2k mode, i.e. the number of OFDM subcarriers was 2048. However, the results given below would be similar for 8k mode or the DVB-H 4k mode. The following discussion only covers the adaption to the LBER, switching off the decoder for BER>UBER is not considered for the sake of clarity of the presentation.

Table II lists the simulation results: For all code rates specified in the DVB-T physical layer [10], the BER of the BSD ("BER BSD") and the BSD with adaptive parameters ("BER A-BSD") and the reduction of decoding effort ("Dec. Off BSD" and "Dec. Off A-BSD") are given for four SNR values. The lowest SNR value for each code rate is the minimum SNR required for QEF system output as specified in [10]. Three additional values are given in 1dB steps. It can be seen, that both decoders ensure QEF system output, as their BER is always better than or equal to the required minimum BER. Considering the reduction of decoding effort, it is obvious that there is a considerable benefit: Depending on the used code, the decoder can save between 7% ($1/2$-rate) and 43% ($7/8$-rate) of the VA's trellis operations for the minimum SNR. Keeping in mind that this is the minimum SNR, which is only found in regions with bad reception conditions, even more significant savings can be excepted for the average user (cf. Table II).

For the adaptive BSD the same conclusions as in the previous section hold: The switched-off-time of the decoder can be increased even more by implementing adaptive parameter

settings. Depending on the code rate an additional amount of up to 20% can be saved.

A comparison of different code rates shows that the higher the coder rate, the more reduction of decoding effort is possible. The reason for this is, that a higher code rate will operate at higher SNR values, which means that less errors occur and thus more and longer error free blocks are detected by the algorithm. Additionally, for higher code rates, smaller values can be selected for the design parameters, because of worse error correcting capabilities, as stated in Section II-B.

V. Conclusions

Syndrome decoding is a powerful approach to reduce the decoding complexity of convolutional codes, which are used in many digital broadcasting systems. In this paper it was shown that the previously presented BSD already offers significant reductions of decoding effort under high SNR reception conditions. An extended BSD adapts its output BER to a BER threshold where QEF reception is possible, which results in an even higher decrease of decoding effort. Both the BSD and the adaptive BSD were implemented in a DVB-T receiver simulation model and showed tremendous savings in the decoding effort under typical reception conditions. Considering that more and more digital broadcasting receivers are included in wireless, battery-powered devices, these decoding approaches can help to increase the operating times of these devices significantly.

TABLE II
Results for DVB-T, 2K-mode, QAM64, AWGN channel.

$R = 1/2$				
SNR [dB]	14.4	15.0	16.0	17.0
BER BSD	$1 \cdot 10^{-4}$	$4 \cdot 10^{-5}$	$5 \cdot 10^{-6}$	$1 \cdot 10^{-6}$
BER A-BSD	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
Dec. Off BSD	7%	11%	20%	31%
Dec. Off A-BSD	7%	19%	33%	51%
$R = 2/3$				
SNR [dB]	16.5	17.0	18.0	19.0
BER BSD	$2 \cdot 10^{-4}$	$8 \cdot 10^{-5}$	$5 \cdot 10^{-6}$	$1 \cdot 10^{-7}$
BER A-BSD	$2 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$9 \cdot 10^{-5}$
Dec. Off BSD	17%	20%	35%	53%
Dec. Off A-BSD	17%	22%	48%	66%
$R = 3/4$				
SNR [dB]	18.0	19.0	20.0	21.0
BER BSD	$2 \cdot 10^{-4}$	$1 \cdot 10^{-5}$	$1 \cdot 10^{-6}$	$3 \cdot 10^{-7}$
BER A-BSD	$2 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$9 \cdot 10^{-5}$
Dec. Off BSD	25%	41%	60%	75%
Dec. Off A-BSD	25%	50%	70%	85%
$R = 5/6$				
SNR [dB]	19.3	20.0	21.0	22.0
BER BSD	$2 \cdot 10^{-4}$	$3 \cdot 10^{-5}$	$1 \cdot 10^{-6}$	$< 1 \cdot 10^{-7}$
BER A-BSD	$2 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$2 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
Dec. Off BSD	34%	47%	72%	85%
Dec. Off A-BSD	34%	51%	79%	92%
$R = 7/8$				
SNR [dB]	20.1	21.0	22.0	23.0
BER BSD	$2 \cdot 10^{-4}$	$1 \cdot 10^{-5}$	$2 \cdot 10^{-7}$	$< 1 \cdot 10^{-7}$
BER A-BSD	$2 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$5 \cdot 10^{-5}$
Dec. Off BSD	43%	65%	86%	94%
Dec. Off A-BSD	43%	67%	91%	96%

References

[1] I. Kang and A. Willson Jr., "Low-power viterbi decoder for CDMA mobile terminals," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 3, pp. 473–482, Mar 1998.

[2] F. Chan and D. Haccoun, "Adaptive viterbi decoding of convolutional codes over memoryless channels," *IEEE Transactions on Communications*, vol. 45, no. 11, pp. 1389–1400, Nov. 1997.

[3] R. Henning and C. Chakrabarti, "An approach for adaptively approximating the viterbi algorithm to reduce power consumption while decoding convolutional codes," *IEEE Transactions on Signal Processing*, vol. 52, no. 5, pp. 1443–1451, May 2004.

[4] J. Geldmacher, K. Hueske, and J. Goetze, "Syndrome based block decoding of convolutional codes," in *Proceedings of the IEEE International Symposium on Wireless Communication Systems (ISWCS '08)*, Reykjavik, Iceland, Oct. 2008, pp. 542–546.

[5] J. Schalkwijk and A. Vinck, "Syndrome decoding of binary rate-1/2 convolutional codes," *IEEE Transactions on Communications*, vol. 24, no. 9, pp. 977–985, Sept. 1976.

[6] G. Forney Jr., "Convolutional codes I: Algebraic structure," *IEEE Transactions on Information Theory*, vol. 16, no. 6, pp. 720–738, Nov. 1970.

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

[8] R. McEliece, "The algebraic theory of convolutional codes," *Handbook of Coding Theory*, vol. I, pp. 1065–1138, 1998.

[9] M. Tajima, K. Shibata, and Z. Kawasaki, "Modification of syndrome trellises for high-rate convolutional codes," The Institute of Electronics, Information and Communication Engineers (IEICE), Tech. Rep., 2002.

[10] *DVB; Framing structure, channel coding and modulation for digital terrestrial television (ETSI EN 300 744)*, ETSI Std., Rev. 1.5.1, 2004.

[11] J. Mei and S. Cheng, "Analysis for error code performance of MPEG-2 transport system," in *Proceedings of the IEEE International Symposium on Consumer Electronics (ISCE'97)*, Dec 1997, pp. 111–113.