Adaptive Low Complexity MAP Decoding for Turbo Equalization

Jan Geldmacher, Klaus Hueske and Juergen Goetze
TU Dortmund University, Information Processing Lab
Otto-Hahn-Str. 4, 44227 Dortmund, Germany.
jan.geldmacher@tu-dortmund.de

Sascha Bialas
Federal Office for Information Security
Mobile Communication Security
Mainzer Str. 84, 53179 Bonn, Germany.

***Abstract*—Turbo equalization is a powerful method to iteratively detect and decode convolutionally encoded data that is corrupted by inter symbol interference (ISI) and Gaussian noise. It is based on the exchange of reliability information between the equalizer and the decoder, which is typically some sort of maximum a posteriori (MAP) decoder. While the number of remaining errors in the received sequence decreases during the iteration process, the computational effort for decoding remains unchanged in each iteration. In this paper a syndrome based MAP decoder is proposed, that is capable of reducing the computational decoding effort during the iteration process without significantly influencing the convergence behavior.**

I. Introduction

Soft input soft output (SISO) decoding is a crucial prerequisite of iterative receiver structures following the Turbo principle as applied in Turbo decoding [1] and Turbo equalization [2]. In case of convolutional codes, the SISO decoder is typically based on the BCJR [3] algorithm, which uses the trellis structure of the encoder for the computation of *a posteriori* information.

The BCJR algorithm can also be applied to the syndrome former trellis of the used code, as described in [4]. The structure of this trellis depends on the syndrome sequence, which is a function of the errors in the received sequence. Hence, error-free parts in the received sequence lead to zero sequences in the syndrome. Considering this, error-free parts in the received sequence can be estimated by analyzing the syndrome sequence before the decoding process starts. This property of syndrome decoding allows a significant reduction of decoding complexity by decoding only erroneous blocks while not processing blocks that are error-free anyway [5]. In [6] this approach has been applied to reduce the decoding complexity of a DVB-T channel decoder.

The reduction of decoding complexity depends on the number of errors in the sequence to be decoded. In conventional systems this is a function of the SNR: A higher SNR leads to lower decoding complexity. However, when considering a system applying Turbo equalization, the number of remaining errors in the sequence to be decoded not only depends on the SNR but also on the current iteration step: For higher iteration steps a reduced number of errors is expected, which leads to a lower decoding complexity.

The paper is organized as follows: In Section II the basic principles of Turbo equalization and the used equalizer are explained. The topic of Section III is the log-MAP decoder with adaptive complexity. After describing the syndrome based preprocessing (III-A), the main principle of syndrome based MAP decoding will be clarified (III-B). Simulation results are presented in Section IV. Conclusions are drawn in Section V.

II. Turbo Equalization

A. Basic Principles

We assume a transmission as depicted in Fig. 1: A binary information sequence represented by a $K \times 1$ vector $\mathbf{u} = \{u_k\}$ is encoded using a rate R convolutional code. The resulting code sequence $\mathbf{v} = \{v_k\}$ of length $L = K/R$ is interleaved, $\mathbf{v}' = \prod(\mathbf{v})$, mapped to modulation symbols and transmitted over a frequency selective channel. The received sequence can then be described as

$$\mathbf{y} = \mathbf{H}_c \mathbf{v}' + \mathbf{n},$$

where $\mathbf{H}_c$ is the $(L + M - 1) \times L$ channel convolution matrix corresponding to the length M channel, and $\mathbf{n}$ represents additive white Gaussian noise (AWGN) with noise power σ_n^2.

The task of the receiver is to compute a suitable estimate $\hat{\mathbf{u}}$ of the original information sequence. Following the turbo principle, this task is solved by iterating between a SISO equalizer and a SISO channel decoder. Both components exchange their beliefs about the probabilities of each bit v_k, which leads to an iterative improvement of the estimation up to a certain limit. The probabilities are expressed as Log-Likelihood Ratios (LLRs), denoted by $L(\cdot)$. The important underlying principle is that both components generate independent, extrinsic, LLRs $L_E(\cdot)$.

Fig. 2 shows the structure of the Turbo detector: The equalizer generates *a posteriori* estimates using the distorted sequence $\mathbf{y}$ and, after the first iteration, additionally the *a priori* information $L_E^d(v_k')$ about the code bits v_k'. Note that mapping/demapping is required to transform between bits and modulation symbols. Extrinsic information is generated by

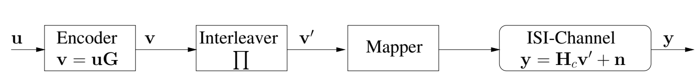

Fig. 1. Convolutional encoded transmission over ISI channel.

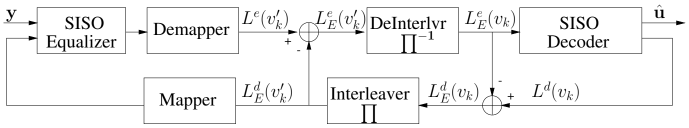

Fig. 2. Turbo detector.

subtracting the *a priori* LLRs from the *a posteriori* values $L^e(v'_k)$,

$$L^e_E(v'_k) = L^e(v'_k) - L^d_E(v'_k). \quad (1)$$

The decoder gets the deinterleaved sequence $L^e_E(v_k)$ as input and generates the *a posteriori* estimate $L^d(v_k)$. Extrinsic information $L^d_E(v_k)$ of the code bits v_k is generated by calculating

$$L^d_E(v_k) = L^d(v_k) - L^e_E(v_k) \quad (2)$$

and the interleaved extrinsic LLRs $L^d_E(v'_k)$ are passed to the equalizer at the next iteration. In the final iteration, the decoder will also output an estimate $\hat{\mathbf{u}}$ of the information sequence.

Maximum a posteriori probability (MAP) versions of equalizer and decoder can be realized based on the trellis representation of the channel and the code, respectively. Alternatively, for the equalizer a linear realization (minimum mean squared error linear equalizer, MMSE LE) has been proposed in [7]. For the simulations presented in this work the MMSE LE is used. Therefore it will be summarized in the following section for the case of BPSK.

B. MMSE Linear Equalizer

The equalizer calculates the extrinsic LLRs $L^e_E(v'_k)$ by applying a time varying filter $\mathbf{c}_k$ to the normalized residuum of the received symbols $\mathbf{y}_k$ and a prediction $\mathbf{H}_b\overline{\mathbf{v}}'_k$ of the received symbols,

$$L^e_E(v'_k) = 2\mathbf{c}_k^H(\mathbf{y}_k - \mathbf{H}_b\overline{\mathbf{v}}'_k)/(1 - \mathbf{s}^H\mathbf{c}_k), \quad (3)$$

where

- $\mathbf{c}_k$ is the $(N \times 1)$ filter vector, with $N = N_1 + N_2 + 1$.
- $\mathbf{y}_k$ is a vector consisting of the received symbols y_{k-N_1} to y_{k+N_2}.
- $\mathbf{H}_b$ is an $N \times (N + M - 1)$ block of $\mathbf{H}_c$.
- $\overline{\mathbf{v}}'_k$ is an $(N+M-1)\times 1$ vector consisting of the expectations $\overline{v}'_{k-M-N_2+1}$ to $\overline{v}'_{k+N_1}$, which can be computed from the LLRs following $\overline{v}'_n = \tanh\left(L^d_E(v'_n)/2\right)$ with $n = k - M - N_2 + 1, ..., k + N_1$. The element $\overline{v}'_k$ is set to zero.
- $\mathbf{s}$ is the $(N_2 + M)$-th column of $\mathbf{H}_b$.

The filter vector is computed as a weighted MMSE estimator,

$$\mathbf{c}_k = (\sigma_n^2\mathbf{I}_N + \mathbf{H}_b\mathbf{V}_k\mathbf{H}_b^H)^{-1}\mathbf{s}, \quad (4)$$

where $\mathbf{V}_k$ is a diagonal matrix,

$$\mathbf{V}_k = \mathrm{diag}(1 - |\overline{v}'_{k-M-N_2+1}|^2, \ldots, 1 - |\overline{v}'_{k+N_1}|^2). \quad (5)$$

Similar to setting $\overline{v}'_k = 0$, the $(N_2 + M)$-th diagonal element of $\mathbf{V}_k$ is set to 1. This guarantees that the filter generates an estimate $L^e_E(v'_k)$, which is independent of $L^d_E(v'_k)$. Considering that equalizer and decoder are separated by an interleaver, it follows that $L^e_E(v'_k)$ is the extrinsic LLR.

III. Low complexity adaptive log-MAP Decoder

The basic idea of the proposed adaptive decoding concept is to preprocess the hard decision of the decoder's input sequence and decompose it into blocks that are considered to be error-free and blocks that are erroneous. Only the erroneous blocks are actually processed by the decoder, while the error-free blocks are just assigned with a high confidence and directly passed to the consecutive stage. The proposed concept is based on the syndrome decoding (SD) principle of convolutional codes [8], which means that the decoding algorithm operates on the trellis of the syndrome former instead of the trellis of the encoder. In the SD approach presented by Schalkwijk et.al. [9] this trellis is searched for the maximum likelihood error sequence, which can then be used to correct the transmission errors in the received sequence (cf. also [6]).

In this work, a MAP decoder based on the syndrome former trellis is proposed. Note that compared to Turbo Codes, the decoder generates LLRs for the code bits, not for the information bits. This makes a syndrome trellis based implementation more straight-forward, compared to [4]. The following Section III-A describes the syndrome based preprocessing, while Section III-B introduces the syndrome based log-MAP decoder.

A. Syndrome Based Preprocessing

Given a convolutional code $\mathcal{C}$ represented by an encoder $\mathbf{G}(D)$, then the syndrome former $\mathbf{H}^T(D)$ is the transpose of an encoder $\mathbf{H}(D)$ of the dual code $\mathcal{C}^\perp$. It holds that

$$\mathbf{G}(D)\mathbf{H}^T(D) = \mathbf{0}, \quad (6)$$

i.e. $\mathbf{H}^T(D)$ is orthogonal to all code sequences of $\mathcal{C}$.

Let $\mathbf{r} = \{r_k\}$ be the hard decision of $L^e_E(v_k)$ such that

$$r_k = \begin{cases} 1 & \text{for } L^e_E(v_k) \geq 0 \\ 0 & \text{for } L^e_E(v_k) < 0. \end{cases} \quad (7)$$

Note that $\mathbf{r}$ changes in each iteration. Then the length L sequence $\mathbf{r}$ can be expressed in time-domain as

$$\mathbf{r} = \mathbf{uG} \oplus \mathbf{e} = \mathbf{v} \oplus \mathbf{e}, \quad (8)$$

where $\mathbf{e}$ represents the remaining channel error. Applying the time-domain representation $\mathbf{H}^T$ of the syndrome former to $\mathbf{r}$ yields the syndrome sequence $\mathbf{b}$ of length $L(1 - R)$:

$$\mathbf{b} = \mathbf{rH}^T = \mathbf{uGH}^T \oplus \mathbf{eH}^T = \mathbf{eH}^T \quad (9)$$

From (9) it can be noticed that due to the orthogonality of $\mathbf{H}^T$ to the code, $\mathbf{b}$ only depends on the channel error. In the case that $\mathbf{r}$ is completely error-free, it is entirely orthogonal to $\mathbf{H}^T$, and thus the syndrome sequence $\mathbf{b}$ is equal to zero. However, in the general case, where $\mathbf{r}$ contains errors, it can only be partially orthogonal to $\mathbf{H}^T$: This means, that error-free subsequences in $\mathbf{r}$ of a certain minimum length will propagate to sequences of consecutive zeros in $\mathbf{b}$. Thus it is possible to estimate error-free blocks in $\mathbf{r}$ by analyzing $\mathbf{b}$.

Based on this observation, adaptive reduced complexity decoding can be realized as follows:

1) Compute the harddecision sequence $\mathbf{r}$ following (7).

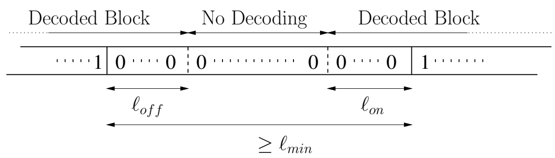

Fig. 3. Preprocessing Parameters.

2) Calculate the syndrome sequence $\mathbf{b} = \mathbf{r}\mathbf{H}^T$.
3) Decompose $\mathbf{r}$ into erroneous and error-free blocks by analyzing $\mathbf{b}$: For blocks of $\geq \ell_{min}$ consecutive zero symbols in $\mathbf{b}$, the corresponding blocks in $\mathbf{r}$ are considered to be error-free. Two additional parameters ℓ_{on} and ℓ_{off} are used to define an extension of the erroneous blocks at the beginning and end, respectively. The meaning of the parameters is illustrated in Fig. 3. It holds $\ell_{on} + \ell_{off} \leq \ell_{min}$.
4) The individual blocks are processed as follows:
   a) The erroneous blocks are fed into the decoder. These blocks are independent from each other and considered to be terminated and initialized in the zero-state[1]. The number of erroneous blocks per frame is denoted by J, and the number of trellis stages of each block j by T_j.
   The syndrome based MAP decoder computes the LLRs $L^d(v_k)$ for each of the J blocks (see Section III-B).
   b) The remaining blocks are considered to be error-free. Hence, they are not processed by the decoder and the corresponding bits are assigned with infinite LLR:
   $$L^d(v_k) = \infty \cdot (2r_k - 1) \tag{10}$$
   Note that for the equalizer this just means that the expectation of the corresponding interleaved bits is set to
   $$\overline{v}'_k = 2r'_k - 1. \tag{11}$$
5) During the iterations the LLRs $L^d(v_k)$ are transformed into extrinsic LLRs using (2) and fed back to the equalizer. After the last iteration a hard-decision $\hat{\mathbf{v}}$ of $L^d(v_k)$ is taken and transformed back to the estimate of the information sequence $\hat{\mathbf{u}}$ using the right inverse of the generator matrix, $\hat{\mathbf{u}} = \hat{\mathbf{v}}\mathbf{G}^{-1}$.

This concept reduces the decoding effort, because only the erroneous parts are processed by the MAP decoder. The reduction depends on the input BER of the decoder: The lower the input BER, the more and the longer error-free blocks can be identified and hence the higher the reduction of decoding effort. On the other hand, in the worst case, when no error-free blocks can be identified ($J = 1$), the whole sequence needs to be processed and the effort is identical to that of the conventional decoder. For turbo equalization, the input BER of the decoder improves from one iteration to the next, in case of convergence. Hence, this results in increased savings of decoding effort during the iteration process.

Processing only parts of the input sequence, and assigning high confidence to the remaining supposedly error-free blocks, may obviously degrade the BER performance of the Turbo Equalization system. Sources of performance degradation may be that

- the LLRs for the erroneous blocks are calculated based only on the individual blocks, not the whole sequence,
- individual blocks may wrongly be assumed to be zero-state initialized or zero-state terminated, and
- error-free blocks may wrongly be considered error-free and thus individual erroneous bits may be assigned with a high confidence.

The choice of the design parameters $(\ell_{min}, \ell_{on}, \ell_{off})$ is therefore critical to maintain a BER performance equivalent to the optimum decoder. Generally, higher values for ℓ_{min} will reduce the impact of the mentioned degrading factors. On the other hand, smaller values will increase the reduction of effort. Therefore, the design parameters have to be carefully selected for a given system and its underlying code. In fact, it is possible to choose parameter values, such that the BER performance does not suffer, and the reduction of effort is still significant. This aspect will become clear from the simulation results presented in Section IV.

Compared to the actual MAP decoding, we consider the preprocessing steps to be of insignificant complexity: Syndrome calculation and application of the right inverse are realized by XOR-operations. The decomposition into blocks can be implemented by just counting consecutive zeros in $\mathbf{b}$ [10].

B. Syndrome based log-MAP Decoder

The structure of the syndrome based log-MAP decoding algorithm (log-MAP SD) is identical to the well-known log-MAP [3], [11]. However, while the latter operates on the encoder trellis, the log-MAP SD is based on the trellis of the syndrome former. More specific, it operates on the trellis defined by $\mathbf{b} = \mathbf{r}\mathbf{H}^T$, where at each time step only the transitions corresponding to the current syndrome symbol are considered. This ensures that all trellis paths are admissible error sequences, i.e. result in a valid code sequence when applied to $\mathbf{r}$. Note also that the trellis complexities of encoder and syndrome former under this constraint are generally identical.

The decoder estimates the LLRs
$$L^d(v_k) = \log \frac{P(v_k = 1|\tilde{\mathbf{r}})}{P(v_k = 0|\tilde{\mathbf{r}})} \tag{12}$$
of the code bits v_k based on the observed input sequence $\tilde{\mathbf{r}} = \{L_E^e(v_k)\}$. Using the trellis representation of the code, (12) can be rewritten as
$$L^d(v_k) = \log \frac{\sum_{(p,q)\in\mathcal{S}_1} \alpha_t(p)\gamma_t(p,q)\beta_{t+1}(q)}{\sum_{(p,q)\in\mathcal{S}_0} \alpha_t(p)\gamma_t(p,q)\beta_{t+1}(q)}, \tag{13}$$
where p and q are trellis states at time instants t and $(t+1)$, respectively, and $\mathcal{S}_0$ and $\mathcal{S}_1$ represent transitions that lead to

[1]For suitable parameter settings this is the case because the decoder operates on the trellis of the syndrome former, where all paths represent error sequences. Thus at the beginning of an erroneous block, the error sequence deviates from the zero-state and at the end it merges back into the zero-state.

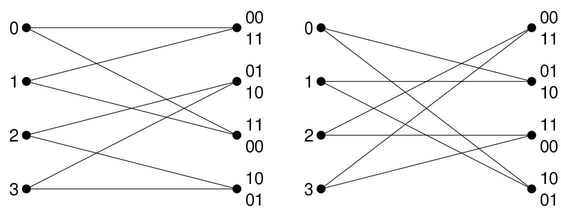

Fig. 4. Trellis of $\mathbf{H}^T(D)$ for $b = 0$ (left) and $b = 1$ (right). The error $\mathbf{e}^{(p,q)}$ corresponding to each transition is shown on the right, while the state numbers are on the left of each trellis module.

$v_k = 0$ and $v_k = 1$, respectively. The forward and backward state metrics $\alpha_t(p)$ and $\beta_t(q)$ are computed recursively by

$$\alpha_t(q) = \sum_p \alpha_{t-1}(p)\gamma_t(p,q), \quad \beta_t(p) = \sum_q \gamma_t(p,q)\beta_{t+1}(q),$$

and the computation of the transition metric $\gamma_t(p,q)$ is formulated in Log domain as

$$\log \gamma_t(p,q) = \frac{1}{2}\left\langle \tilde{\mathbf{r}}^{(t)}, \left(2(\mathbf{e}^{(p,q)} \oplus \mathbf{r}^{(t)}) - 1\right)\right\rangle, \qquad (14)$$

where $\tilde{\mathbf{r}}^{(t)}$ and $\mathbf{r}^{(t)}$ represent n soft values and n corresponding harddecision bits at time instant t, respectively. The vector $\mathbf{e}^{(p,q)}$ contains the n error bits associated with the transition from state p to state q and $\langle \cdot, \cdot \rangle$ denotes the inner product.

Applying the syndrome former trellis based log-MAP SD to the complete sequence $\tilde{\mathbf{r}}$ delivers identical LLRs as the conventional log-MAP, which operates on the encoder trellis: This is readily seen, because the transitions $\mathbf{v}^{(p,q)}$ of the encoder trellis are linked to the transitions $\mathbf{e}^{(p,q)}$ of the syndrome former trellis by

$$\mathbf{r}^{(t)} = \mathbf{v}^{(p,q)} \oplus \mathbf{e}^{(p,q)}. \qquad (15)$$

For the proposed reduced complexity decoding method, the log-MAP SD is separately applied to the J erroneous blocks, which have been identified in the preprocessing stage (Section III-A). This approach is therefore referred to as log-MAP BSD (Block SD) in the following. These blocks are terminated and initialized to the zero-state, thus the forward and backward metrics have to be suitably initialized for each block j as

$$\alpha_0(p) = \begin{cases} 1 & \text{for } p = 0 \\ 0 & \text{else,} \end{cases} \quad \text{and} \quad \beta_{T_j-1}(q) = \begin{cases} 1 & \text{for } q = 0 \\ 0 & \text{else.} \end{cases}$$

IV. Simulations

A. Parameters settings

The proposed reduced complexity decoder, log-MAP BSD, is evaluated in a Turbo equalization system with the following parameters:

- ISI-Channel of length $M = 5$ and channel impulse response $\mathbf{h} = [0.227, 0.46, 0.688, 0.46, 0.227]$,
- Non-recursive $R = 1/2$-rate terminated convolutional code with $\mathbf{G}(D) = \left[D^2 + 1,\ D^2 + D + 1\right]$ and corresponding syndrome former $\mathbf{H}^T(D) = \left[D^2 + D + 1,\ D^2 + 1\right]^T$ with trellis shown in Fig. 4,
- 2^{15} data bits per frame, which results for the half rate code in $T = 2^{15}$ trellis sections for one frame,
- 14 iterations and BPSK modulation.

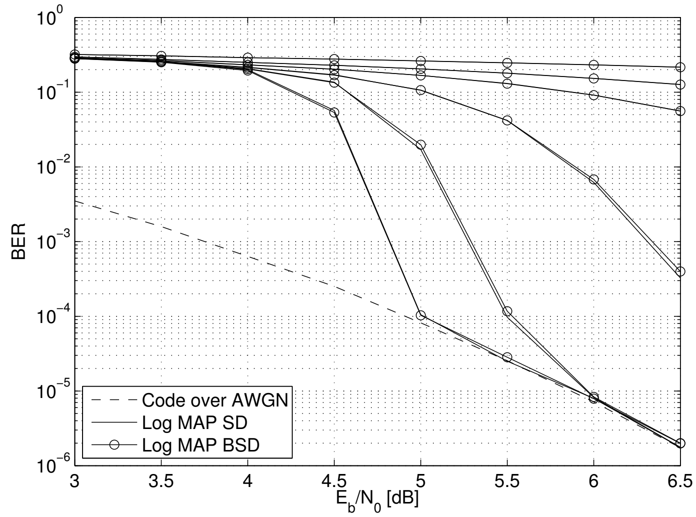

Fig. 5. BER for log-MAP SD and log-MAP BSD for iterations 1,2,3,5,8,14.

In the following we use the log-MAP SD, which processes the whole sequence without any preprocessing, as the reference. It is important to emphasize, that the performance and the complexity of log-MAP SD are identical to the conventional log-MAP decoder.

The design parameter triplet $(\ell_{min}, \ell_{on}, \ell_{off})$ is chosen heuristically using simulation results for achievable BER and reduction of decoding effort. The parameters can be precomputed and are independent of the equalizer or current channel conditions. They only depend on the used code. Here we choose the parameters as

$$(\ell_{min}, \ell_{on}, \ell_{off}) = (9,\ 4,\ 4), \qquad (16)$$

which guarantees optimum BER performance but significant reduction of effort.

B. Results

Fig. 5 shows the BER of the Turbo Equalizer after $1, 2, 3, 5, 8$ and 14 iterations. A system with reduced complexity decoder (log-MAP BSD) is compared to a reference implementation with log-MAP SD decoding. It is easy to see that the log-MAP BSD system performs as well as the reference system.

The interesting point is to analyze the reduction of decoding effort. This reduction can be measured as the ratio of the accumulated size of erroneous (decoded) blocks to the length T of the entire received sequence. If we denote the number of identified erroneous blocks in iteration i as $J(i)$ and the length of block j as $T_j(i)$, then this ratio can be expressed as

$$R(i) = \frac{1}{T}\sum_{j=1}^{J(i)} T_j(i), \qquad (17)$$

for iteration i. Fig. 6 shows $R(i)$ in percent for different E_b/N_0. One can see, that the decoder processes nearly the

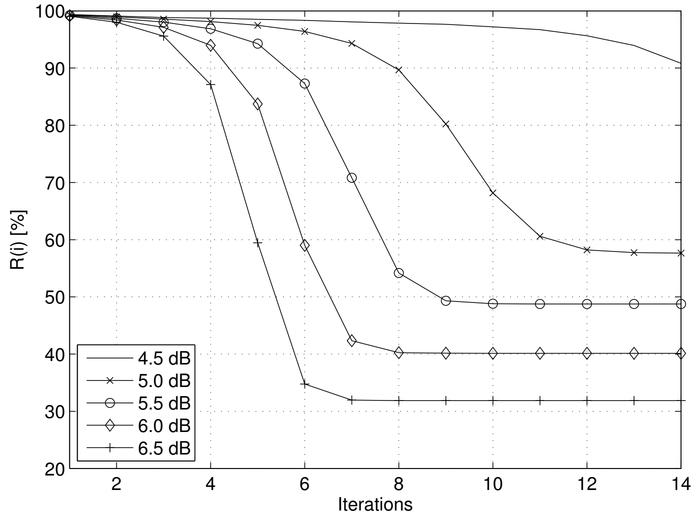

Fig. 6. Ratio $R(i)$ in percent for different Noise levels E_b/N_0.

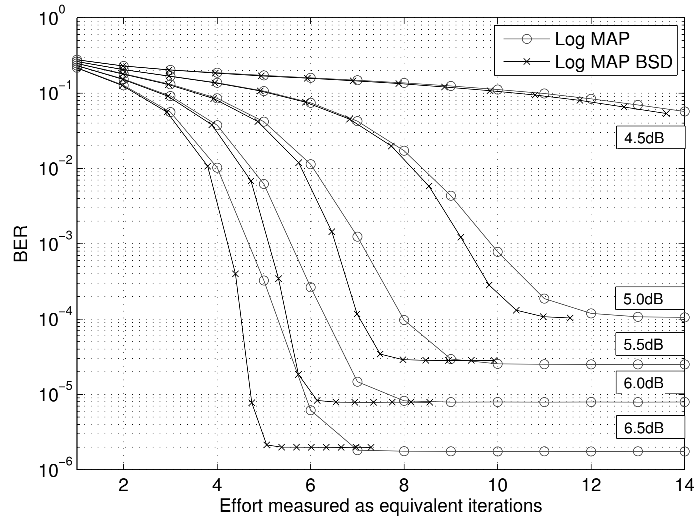

Fig. 7. Convergence vs. equivalent iterations for different Noise levels..

whole sequence for low iteration numbers ($R(i) \approx 1$), which corresponds to the effort of the conventional decoder. But for increasing iteration number, $R(i)$ decreases towards a lower bound. This lower bound decreases with increasing SNR. For example, at $6dB$ the decoder only processes about 59% and 42% of the received sequence in the 6-th and 7-th iterations.

To make the complexity analysis independent of the equalizer complexity, the parameters were chosen to achieve an identical convergence behavior, i.e. the reduced complexity decoder reaches the BER floor with the same number of iterations as the reference. In Fig. 7 the convergence in terms of BER is plotted over the number of equivalent iterations. The number of equivalent iterations is computed as $\sum_{i=1}^{I} R(i)$ for iteration I and indicates the computational effort measured as iterations of the reference decoder.

From Fig. 7 it can be seen that the log-MAP BSD reaches the BER floor in the same number of iterations as the reference decoder. For example, at $5dB$ both decoders reach the floor after about 12 iterations. For the log-MAP BSD this can be seen by counting the line markers ($\times$). The computational effort to reach the BER limit, however, is only about 10.5. For the other noise levels, we can find a saving of about 1.8 to 2 iterations compared to the reference decoder, without impact on the BER performance.

In summary, for the considered system the log-MAP BSD delivers the same BER as the reference decoder, but requires only 71% to 87% of the operations to reach the error floor.

V. Conclusions

This paper introduces a syndrome based MAP decoding concept with reduced, adaptive complexity and shows its application to Turbo Equalization. The basic idea is to avoid decoding of supposedly error-free blocks in the decoder's input sequence. If a proper set of design parameters is selected, this does not affect the BER performance of the system. It can, however, reduce the decoding effort by up to 30% compared to the reference decoder in the considered system.

Besides reduction of decoding effort, the proposed concept can also be used to improve the speedup of a parallel MAP decoder implemented on a multi-core platform: Identification of error-free blocks is equivalent to state identification and can thus be used to avoid overlapping between parallel decoded subblocks [5], [10].

References

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes," in *IEEE International Conference on Communications (ICC 93)*, vol. 2, Geneva, Switzerland, May 1993, pp. 1064 –1070 vol.2.

[2] C. Douillard, M. Jezequel, C. Berrou, A. Picart, P. Didier, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo-equalization," *European Transactions on Telecommunications*, vol. 6, no. 5, pp. 507–511, 1995.

[3] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284 – 287, March 1974.

[4] M. Tajima, K. Shibata, and Z. Kawasaki, "Relation between encoder and syndrome former variables and symbol reliability estimation using a syndrome trellis," *IEEE Transactions on Communications*, vol. 51, no. 9, pp. 1474–1484, Sept. 2003.

[5] 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.

[6] ——, "An adaptive and complexity reduced decoding algorithm for convolutional codes and its application to digital broadcasting systems," in *Proceedings of the International Conference on Ultra Modern Telecommunications (ICUMT2009)*, St. Petersburg, Russia, Oct. 2009.

[7] M. Tuchler, R. Koetter, and A. Singer, "Turbo equalization: principles and new results," *IEEE Transactions on Communications*, vol. 50, no. 5, pp. 754–767, May 2002.

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

[9] 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.

[10] K. Hueske, J. Geldmacher, and J. Goetze, "Multi core performance of a block syndrome decoder for convolutional codes," in *6th Karlsruhe Workshop on Software Radios (WSR10)*, Karlsruhe, Germany, 2010.

[11] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal map decoding algorithms operating in the log domain," in *Proceedings of the IEEE International Conference on Communications (ICC'95)*, vol. 2, Seattle, June 1995, pp. 1009 –1013.