Packet Transmission with Turbo Codes over Fading Channels

Bartosz Mielczarek[1] and Arne Svensson[2]

[1]Chalmers University of Technology, SE-412 96 Göteborg, Sweden. E-mail: bartosz@s2.chalmers.se.
[2]Chalmers University of Technology, SE-412 96 Göteborg, Sweden. E-mail: arne.svensson@s2.chalmers.se.

Abstract

This paper presents the main issues of using turbo coding schemes in wireless packet transmission networks, including rate compatibility, channel estimation and header encapsulation. We claim that incorporating payload information bits, pilot bits and header bits into the code structure yields performance improvements in terms of bit error probability and easiness of implementation. We propose a novel idea of code construction using frequent trellis termination which has interesting properties making it particularly suitable for packet transmission.

1 Introduction

Turbo codes [1, 2] are one of the most promising coding techniques for very noisy channels. Their exceptional performance allows transmitting the signal with very low power which is one of the most crucial demands in future mobile systems. There exist, however, a number of problems with the practical use of turbo codes in systems operating over fading channels.

Wireless packet transmission applications usually employ different block lengths and code structures to adapt to changing parameters of a channel. For example, if the fading channel quality decreases, the problem can be counteracted by decreasing the rate of the code which presents a problem with turbo codes. The natural rate for the typical parallel turbo code is 1/3 and reducing it demands additional hardware blocks. Even if the rate can be decreased, using a fixed interleaver size means sending additional blocks of data over the channel, which causes additional delays and requires buffering at the receiver.

Another inherent feature of turbo codes is the iterative decoding, i.e., the quality of the decoded bits becomes better after each iteration. While this allows for significant reduction of the decoder complexity, it creates problems for certain applications demanding fast, reliable decoding with low bit error rate (BER). One example is multiuser packet transmission. In such systems, each packet is marked with a so-called header which identifies the sender and the receiver of a packet. If the header is turbo encoded, in most cases more than one iteration is needed to decode it reliably. If there are many users in the system, a lot of packets will not be intended for the particular receiver and their decoding will take a lot of power and cause significant delays.

Finally, when the detection of the signal is coherent, the problem of channel estimation using pilot symbols appears. Inserting external pilot bits into the packet reduces the payload and shortens the interleaver length of a turbo code which decreases its performance.

In this paper we address all these issues. We will show that frequent termination of the first turbo code trellis can be used to alleviate the above problems. The rate compatibility can be provided by changing the termination patterns of the trellis and the interleaver length can be kept constant which simplifies the hardware design. Moreover, channel estimation can be achieved by using fixed code sequences instead of external symbols. Finally, by proper design of the code termination, header bits can be decoded with high reliability already after the first iteration. The additional benefit lies in simplification of the turbo decoder hardware.

2 System model

The system analyzed in this paper consists of four blocks: turbo encoder, Rayleigh fading channel, channel estimator and turbo decoder. A brief description of these blocks follows.

2.1 Turbo encoder

The typical parallel turbo encoder consists of two identical recursive systematic convolutional encoders (RSC), connected in a parallel way by a pseudo-random interleaver. The stream of N input bits x_k^s is fed to the first encoder without any modifications and is randomly interleaved for the second encoder. The streams of systematic bits x_k^s and parity bits x_k^p are then transformed into the antipodal format $\pm\sqrt{E_c}$ (where E_c is the signal energy), punctured, if necessary, to rate r_0 and BPSK-modulated. The N/r_0 code symbols are then channel-interleaved and P pilot symbols used for channel estimation are added. The total length of the frame is therefore $N/r_0 + P$.

2.2 Rayleigh fading channel

The encoded signal is transmitted over a flat Rayleigh fading channel with normalized Doppler frequency $f_D T$ (where f_D is the Doppler frequency and T is the symbol period) and AWGN. After down-converting, matched-filtering and normalizing to unit energy, the received complex baseband signal can be expressed as

$$z_k = a_k x_k + n_k \tag{1}$$

where a_k is the complex fading amplitude and n_k is white, complex Gaussian noise process with variance $E[n_k n_k^*] = N_0/2E_c$ per dimension. The fading amplitude is Rayleigh distributed and the phase error follows the uniform distribution. The correlation between the amplitudes a_k is

$$R_c(n) = E[a_k a_{k+n}^*] = J_0(2\pi f_D n T) \tag{2}$$

where J_0 is the zero-order modified Bessel function of the first kind (note that the fading process is normalized to unit power). The power spectrum of the fading process is the well known U-shaped Jakes spectrum resulting from isotropic scattering.

2.3 Channel estimation

After receiving the signal, the first task of the receiver is the channel estimation where the complex fading amplitudes a_k are estimated as $\hat{a}_k$ prior to decoding. Note, that the channel estimation is performed prior to channel deinterleaving to keep the correlation between samples.

In this paper, we use a coherent, Pilot Symbol Assisted Modulation (PSAM) algorithm which periodically inserts known symbols into the encoded data stream. The classical linear approach to the channel estimation in the presence of noise is the Wiener filtering which minimizes the minimum square error of the estimation defined as

$$\mathbf{w}(k) = \arg \min_{\mathbf{w}(k)\in\mathbb{R}^K} E\left[|a_k - \hat{a}_k|^2\right], \hat{a}_k = \mathbf{w}^T(k)\mathbf{z} \tag{3}$$

where $\mathbf{w}(k)$ are the filter coefficients calculated for every bit position $k \in 1, 2, ...N/r_0 + P$ and $\mathbf{z}$ is the vector of length $N/r_0 + P$ containing the received pilot symbols at P positions and zeros in the positions corresponding to the encoded bits.

To solve (3), the well known Wiener-Hopf equations can be used to calculate the filter coefficients $\mathbf{w}(k)$ as solution to the equation

$$\mathbf{R}\mathbf{w}(k) = \mathbf{r}(k) \tag{4}$$

where autocorrelation matrix $\mathbf{R}$ can be calculated for a given fading rate and noise level as

$$\mathbf{R}_{n,k} = \begin{cases} R_c(0) + N_0/E_c & n = k \\ R_c(n-k) & \text{otherwise} \end{cases}. \tag{5}$$

The vector $\mathbf{r}(k)$ must be calculated for every symbol position k as

$$\mathbf{r}_n(k) = R_c(n-k). \tag{6}$$

After channel estimation, the received complex channel symbols are demodulated and fed to the turbo decoder as

$$y_k = \frac{\Re\{z_k \hat{a}_k^*\}}{|\hat{a}_k|} \tag{7}$$

In the following sections we will use y_k^s and y_k^p for demodulated noisy systematic bits and parity bits, respectively.

2.4 Turbo decoder

In this paper we implement the BCJR algorithm as the component code decoder. The principle of the turbo decoding (see [1]) can be formulated as

$$L(\hat{u}_k) = L_C y_k^s + L_e(\hat{u}_k) + \\ + \log\left(\frac{\sum_{S^+} \alpha_{k-1}(s')\tilde{\gamma}_k^{(e)}(s',s)\beta_k(s)}{\sum_{S^-} \alpha_{k-1}(s')\tilde{\gamma}_k^{(e)}(s',s)\beta_k(s)}\right) \tag{8}$$

where $\tilde{\gamma}_k^{(e)}(s',s)$ is given as

$$\tilde{\gamma}_k^{(e)}(s',s) = \exp\left(\frac{1}{2}L_C y_k^p x_k^p\right) \tag{9}$$

and the $\alpha_{k-1}(s')$ values are the probabilities of arriving at trellis state s' with forward transitions and the $\beta_k(s)$ values are the probabilities of arriving at trellis state s with backward transitions. The summations are performed over S^+ state transitions corresponding to data bit $x_k^s = 1$ and S^- state transitions corresponding to data bit $x_k^s = -1$, respectively.

The first term $L_C y_k^s$ where $L_C = 4E_c/N_0$ is the channel reliability factor, is called the *channel value.* The second term, $L_e(\hat{u}_k)$, is the *a priori* information provided by the second BCJR decoder. Finally, the third term, the *extrinsic* information, is calculated by the first BCJR decoder based on the whole received sequence and the code trellis. This value is then passed to the second decoder in the next iteration.

In this paper, we assume that the turbo decoder does not incorporate the channel model into its structure, i.e., the classical turbo decoder is used for all the simulations.

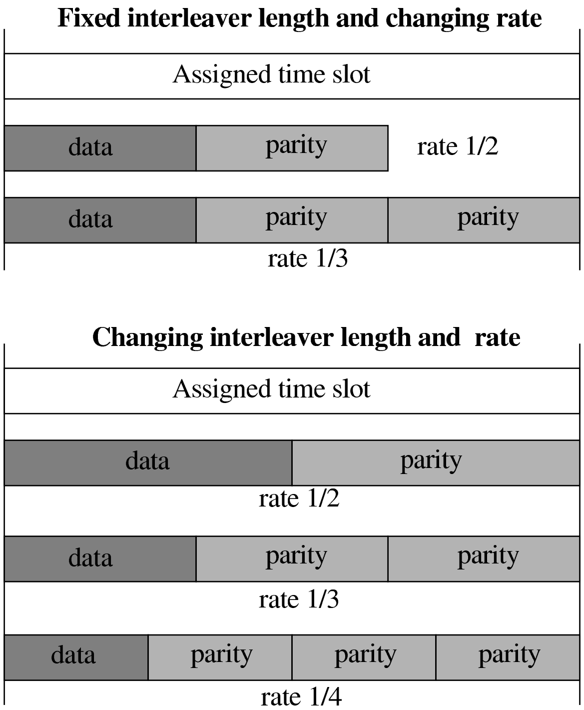

Figure 1: Possible packet organizations

3 Turbo codes in packet systems

3.1 Packet division

Two possible solutions for using turbo codes in packet systems are shown in Figure 1. The simplest implementation employs a constant interleaver length which does not vary with changing rates. This solution, while easy to implement, limits the available rates. For example, if the assigned time slot is equal to the codeword length of rate 1/3 code, decreasing the rate will mean that additional parity bits must be transmitted in a separate packet. This is not always possible due to the delay requirements of the services and/or buffer storage capabilities of the receiver. The second problem arises when the rate is increased. If the codeword gets shorter, a fraction of the time slot will not be used which reduces the bandwidth efficiency.

Another solution is to change the interleaver length for each rate change. If the rate decreases, the interleaver length decreases as well and vice versa. This way the whole packet is always optimally used, i.e., there are no empty spaces for high rates and there is no need to transfer the codeword over more than one packet. Unfortunately, such a solution involves implementing an interleaver for each possible rate and shortens the available interleaver impairing the BER performance.

3.2 Pilot symbols and header bits

In a wireless system, not all the bits in a frame can be used to transmit data. Some of the available bandwidth must be used for pilot symbols and header bits. If the pilot symbols which do not belong to the code structure are inserted into the packet, the effective payload length decreases. This means that the length of the interleaver must be decreased (so the codeword will fit into the remaining slot space) and the resulting bit error rate of the code will decrease as well.

Moreover, in realistic applications, there appears frequently a need for unequal error protection (UEP) which is used to increase reliability of decoding of a selected group of bits (in multiuser scenarios, they can include header or packet identification bits). For the same reasons as above, it is preferable that those bits are encoded using the same turbo code as the other data bits. However, without any modifications to the code, unequal error protection is difficult to achieve for the arbitrary number of header bits.

4 Frequent trellis termination

A possible solution to the above problems is a frequent trellis termination and insertion of the pilot and header symbols into the code structure (see [4]).

Figure 2 shows the principle of the algorithm operation. The data sequence is divided into n sub-blocks of fixed length. Some of the sub-blocks begin with a fixed pilot sequence which is used by a channel estimator. The header and payload data bits follow and the first code trellis is terminated by inserting tail bit sequences. This splits one long trellis of the first encoder into a set of shorter blocks, carrying pilot bits, header bits, data bits and tail bits. Since the second encoder is left unchanged, the principle of a turbo code operation is the same, i.e,. the second trellis is not terminated at all due to the presence of a pseudo-random interleaver (this means that all the classes of bits will be spread randomly in the whole packet).

Employing the above method gives the system two additional levels of adaptivity. Firstly, the number of pilot symbols can be adaptively changed without the need to reconstruct the frame structure and secondly, by changing the number of sub-blocks, it is possible to almost arbitrarily change the rate of the code. This stems from the fact that inserting tail bits which carry no useful information is equivalent to reducing the code rate. The effective rate of such a code is given as

$$r = r_0 \left(\frac{N - nK}{N + r_0 P} \right) \qquad (10)$$

where K is the memory of the code.

In general, if the channel quality decreases, more pilot symbols can be inserted in the beginning of each

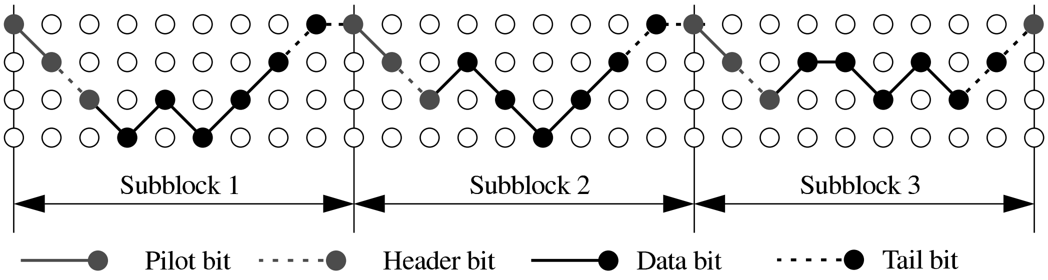

Figure 2: Frequent termination of the first turbo code trellis

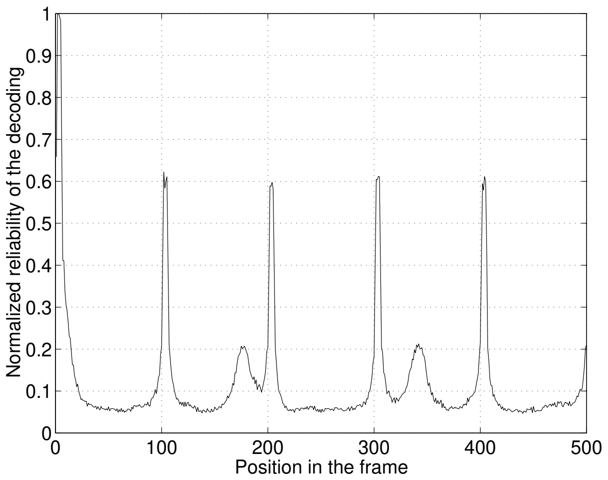

Figure 3: Absolute extrinsic information passed from the first to the second decoder as a function of the position in the code block (N=500, with 10 header bits placed in pairs every 100 bits $E_b/N_0 = 7$ dB).

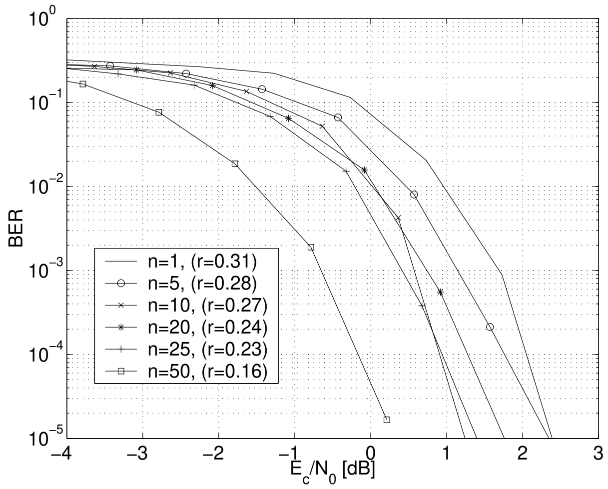

Figure 4: Bit error rates for different trellis termination patterns

sub-block and the termination of the trellis can be done more frequently which reduces the code rate and provides better protection for fewer transmitted bits.

5 Header decoding

Since the reliability of bit detection increases close to the trellis termination points, a straightforward way to decrease the BER of header bits is to insert them between two trellis terminations (see [3]). In this way, the header bits are 'wedged' between the well-defined, all-zero state on the left and the all-zero state on the right. This will lead to a better reliability of decoding as Figure 3 shows. In this figure, we present the average reliability of the extrinsic information generated by the first APP decoder as a function of the position in the code block. One can clearly see that the reliability of bit decoding is much higher for the header bits. Additional high reliability peaks result from uneven spread of channel estimation errors.

6 Simulations

The proposed code modifications were tested by Monte Carlo simulation. We used a parallel rate 1/3 turbo encoder with two RSC (037,033) encoders ($K = 4$) connected by a pseudo-random interleaver of size $N = 500$. Different trellis termination patterns were used, starting with the non-modified code ($n = 1$) and with increasing number of trellis terminations n. Within each frame $P = 150$ symbols were used as pilot symbols. 10000 random data packets were transmitted independently and decoded using our method.

The results of the simulations are shown in Figure 4 with bit error rates for different number of terminations and E_c/N_0, and in Figure 5 where the plots are drawn against normalized SNR defined as $E_b/N_0 = E_c/rN_0$. The first figure shows that by using the frequent termination, it is possible to adjust the code rate according to the channel quality, without changing the length of the transmission. This way the constant bit error rate can be maintained regardless the noise level in the channel. Figure 5 shows the

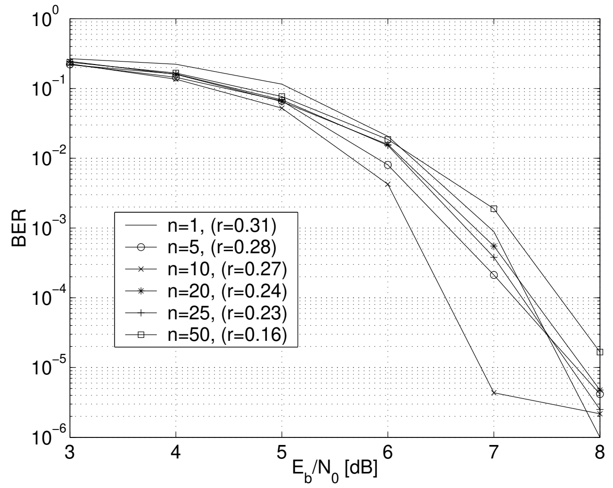

Figure 5: Bit error rates for different trellis termination patterns and normalized SNR=E_b/N_0

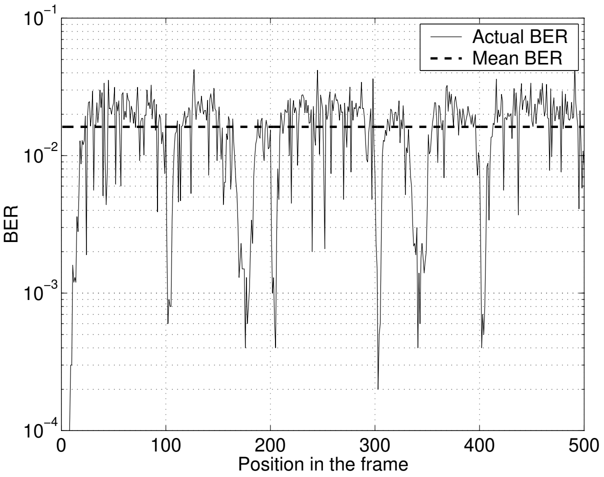

Figure 6: Bit error rates for different bit positions (N=500, with 10 header bits distributed in pairs every 100 bits $E_b/N_0 = 7$ dB).

performance of the scheme from the information theory point of view. An interesting observation is that, while our codes have worse Hamming distance properties (the frequent trellis termination increases the minimum free distance weight), the suboptimal decoding algorithm actually benefits from faster decoding of certain bits. This is particularly visible when comparing the curve corresponding to 10 terminations with the non-modified code. The gain in the waterfall region is around 1 dB, even though the flattening of the BER curve occurs faster in the first case.

Finally, the header decoding performance of the proposed algorithm is shown in Figure 6, where after only 1 iteration, 10 header bits are decoded with BER $< 10^{-3}$ while the mean BER of other bits is almost two orders of magnitude higher.

7 Conclusions

The practical applications of turbo codes are still few and a lot of research is necessary to gain full insight into the specific problems they present. It is, however, quite clear that a lot can be gained by modifying turbo codes to accommodate specific tasks such as, for example, channel estimation.

We have shown that by modifying the structure of the uninterleaved turbo code component, we can easily adapt the code properties to the changing channel parameters since the rate and the number of pilot symbols do not have to be fixed beforehand. We have also shown that our relatively simple modification of the code can provide very effective and fast header decoding, reducing the delay and power consumption at the decoder. The proposed algorithm can be easily tailored to any protection level of the header bits.

The future work will concentrate on the analysis of the modified turbo decoder properties. The performance loss due to the early trellis termination should be examined in greater detail as well as the convergence properties of the turbo decoder in waterfall region.

8 Acknowledgments

This work was supported by the Swedish Foundation for Strategic Research under the Personal Computing and Communication grant.

References

[1] S. A. Barbulescu. *Iterative decoding of turbo codes and other concatenated codes.* PhD thesis, School of Electrical Engineering, Faculty of Engineering, University of South Australia, Feb. 1996.

[2] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon limit error control-correcting coding and decoding: Turbo-codes (1). In *Proc. IEEE International Conference on Communications*, pages 1064–1070, Geneva, Switzerland, 1993.

[3] B. Mielczarek and A. Svensson. Fast header decoding using turbo codes on Rayleigh fading channels. In *Proc. International Conference on Information, Communication, and Signal Processing*, Singapore, 2001.

[4] B. Mielczarek and A. Svensson. Joint adaptive rate turbo decoding and synchronization on Rayleigh fading channels. In *Proc. IEEE Vehicular Technology Conference*, pages 1342–1346, Rhodes, Greece, 2001.