Reduced Latency Turbo Decoding

Yige Wang, Juntan Zhang, Marc Fossorier, Jonathan S. Yedidia

TR2005-089 June 2005

Abstract

Reduced latency versions of iterative decoders for turbo codes are presented and analyzed. The proposed schemes converge faster than standard and shuffled decoders. EXIT charts are used to analyze the performance of the proposed algorithms. Both theoretical analysis and simulation results show that the new schedules offer good performance / complexity trade-offs.

International Workshop on Signal Processing Advances in Wireless Communication

Appeared in the Proceedings of SPAWC 2005, the sixth International Workshop in Signal Processing Advances for Wireless Communications, April 2005.

REDUCED LATENCY TURBO DECODING

Yige Wang, Juntan Zhang, and Marc Fossorier

Department of Electrical Engineering
University of Hawaii at Manoa, Honolulu, HI 96822
Email: yige, juntan, marc@spectra.eng.hawaii.edu

Jonathan S. Yedidia

Mitsubishi Electric Research Laboratories
201 Broadway, Cambridge, MA 02139
Email: yedidia@merl.com

ABSTRACT

Reduced latency versions of iterative decoders for turbo codes are presented and analyzed. The proposed schemes converge faster than standard and "shuffled" decoders. EXIT charts are used to analyze the performance of the proposed algorithms. Both theoretical analysis and simulation results show that the new schedules offer good performance / complexity trade-offs.

1. INTRODUCTION

Turbo codes [1] have received significant attention recently, mostly due to their near-Shannon-limit performance. They are normally decoded using a "turbo-decoder" [2] that is a symbol-by-symbol soft-input/soft-output decoding algorithm. Turbo-decoders process the received symbols recursively to improve the reliability of each symbol using the constraints that specify the code. A "shuffled" decoding algorithm was proposed in [3], that aims to take better account of the different relative reliabilities of extrinsic messages in the turbo decoding algorithm. Compared with the standard turbo decoding approach, the shuffled turbo decoding algorithm converges faster, and with almost the same computational complexity. In [4]-[6], a "replica shuffled" decoding scheme has been presented, which was shown to improve the convergence of iterative decoders for codes like LDPC (low-density parity check) codes. It uses replicas of sub-decoders working in parallel and provides a faster convergence than plain shuffled decoders at the expense of higher complexity. In this paper, replica shuffled turbo decoding algorithms are presented, and compared with standard and plain shuffled turbo decoders using EXIT charts [7]-[9]. The EXIT chart analysis agrees with simulations that show faster convergence of the replica-shuffled decoders.

2. ITERATIVE DECODING OF TURBO CODE

A turbo code [1] encoder comprises the concatenation of two (or more) convolutional encoders and its decoder consists of two (or more) soft-in/soft-out convolutional decoders which feed reliability information back and forth to each other. Initially, the decoders only use the channel output as input, and generate soft output for each symbol. Subsequently, the output reliability measures of the decoded symbols at the end of each decoding iteration are used as inputs for the next iteration. The decoding iteration process continues until a certain stopping condition is satisfied. Then hard decisions are made based on the output reliability measures of decoded symbols from the last decoding iteration.

For simplicity, we consider a turbo code that consists of two rate-$1/n$ systematic convolutional codes with encoders in feedback form. Let $\mathbf{u} = (u_1, u_2, \ldots, u_K)$ be an information block of length K and $\tilde{\mathbf{u}} = (\tilde{u}_1, \tilde{u}_2, \ldots, \tilde{u}_K)$ be the sequence permuted by the interleaver, according to the mapping $\tilde{u}_k = u_{\pi(k)}$, for $k = 1, 2, \ldots, K$. We assume that $k \neq \pi(k), \forall k$. There is a unique corresponding reverse mapping $u_k = \tilde{u}_{\pi^-(k)}$, for $k = 1, 2, \ldots, K$ and $k \neq \pi^-(k), \forall k$. Denote $\mathbf{c} = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K)$ as the coded sequence for $\mathbf{u}$, where $\mathbf{c}_k = (c_{k,1}, \ldots, c_{k,n})$, for $k = 1, 2, \ldots, K$, is the output code block at time k. Assume BPSK transmission over an AWGN channel, with u_k and $c_{k,j}$ all taking values in {+1, -1} for $k = 1, 2, \ldots, K$ and $j = 1, 2, \ldots, n$. Let $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K)$ be the received sequence, where $\mathbf{y}_k = (y_{k,1}, y_{k,2}, \ldots, y_{k,n})$ is the received block at time k. Let $\hat{\mathbf{u}} = \{\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_K\}$ denote the estimate of $\mathbf{u}$. Let s_k denote the encoder state at time k. Following [2], define: $\alpha_k(s) = p(s_k = s, \mathbf{y}_1^k)$, $\gamma_k(s', s) = p(s_k = s, y_k | s_{k-1} = s')$, $\beta_k(s) = p(\mathbf{y}_{k+1}^K | s_k = s)$, where $\mathbf{y}_a^b = (\mathbf{y}_a, \mathbf{y}_{a+1}, \ldots, \mathbf{y}_b)$, and let $\alpha_k^{(m)}(s)$, $\gamma_k^{(m)}(s', s)$, $\beta_k^{(m)}(s)$ be the corresponding values computed in component decoder m, with $m = 1, 2$. Let $L_{em}^{(i)}(\hat{u}_k)$ denote the extrinsic value of the estimated information bit $\hat{u}_k$ delivered by component decoder m at the ith iteration [10].

2.1. Standard serial and parallel turbo decoding

The decoding approach proposed in [1] operates in a serial mode. The disadvantage of this scheme is high decoding delay. A parallel turbo decoding algorithm was presented in [11], in which all component decoders operate in parallel at any given time. After each iteration, each component decoder delivers extrinsic messages to other decoder(s) which use these messages as a priori values at the next iteration. These serial and parallel approaches are illustrated in Fig. 1 (a) and Fig. 1 (b), respectively.

2.2. Plain shuffled turbo decoding

In shuffled turbo decoding [3], the two component decoders operate simultaneously as in parallel turbo decoding, but the ways of updating and delivering messages are different. We assume that the two component decoders deliver extrinsic messages synchronously, i.e., $T_k^1 = T_k^2$, where the T_k^1 and T_k^2 denote the times at which decoder-1 and decoder-2 deliver the extrinsic values of the kth estimated symbol of the original information sequence $\mathbf{u}$ and of the interleaved sequence $\tilde{\mathbf{u}}$, respectively. The shuffled turbo decoding scheme processes the backward recursion followed by the forward recursion. Let us first consider the forward recursion stage at the ith iteration of component decoder-1. After time T_{k-1}^1, the values of $\alpha_k^{(1)}(s)$ should be updated and the values of $\gamma_k^{(1)}(s)$ are needed. There are two possible cases. The first case is $k < \pi^-(k)$, which means the extrinsic value $L_{e2}^{(i)}(\hat{u}_k)$ of the information bit $\hat{u}_k$ is not available yet. Then the values $\gamma_k^{(1)}(s)$, which are stored in the backward recursion stage of the current iteration, are used to update the values $\alpha_k^{(1)}(s)$ and $L_{e1}^{(i)}(\hat{u}_k)$. The second case is $k > \pi^-(k)$, which means the extrinsic value $L_{e2}^{(i)}(\hat{u}_k)$ of the information bit $\hat{u}_k$ has already been delivered by decoder-2. Then this newly available $L_{e2}^{(i)}(\hat{u}_k)$ is used to compute the values $\gamma_k^{(1)}(s)$ (then stored), $\alpha_k^{(1)}(s)$, and $L_{e1}^{(i)}(\hat{u}_k)$. The backward recursion in decoder-1 as well as both recursions in decoder-2 are realized based on the same principle. After I_{max} iterations, the shuffled turbo decoding algorithm outputs $\hat{\mathbf{u}} = (\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_K)$ as the decoded codeword, where $\hat{u}_k =$sgn$[L_{e1}^{(i)}(\hat{u}_k) + L_{e2}^{(i)}(\hat{u}_k) + \frac{4}{N_0} y_{k,1}]$.

2.3. Replica shuffled turbo decoding

In the plain shuffled turbo decoding, we assume all the component decoders process the backward recursion followed by the forward recursion and refer to them as $\overrightarrow{D}1$ and $\overrightarrow{D}2$, respectively. In order to speed up the convergence, the replica shuffled turbo decoding employs two more component decoders, say $\overleftarrow{D}1$ and $\overleftarrow{D}2$, which operate in the reverse order [4]-[6]. The four decoders exchange extrinsic messages simultaneously, i.e., $\overrightarrow{T}_k^1 = \overrightarrow{T}_k^2 = \overleftarrow{T}_k^1 = \overleftarrow{T}_k^2$, where the $\overrightarrow{T}_k^1$ ($\overleftarrow{T}_k^1$) and $\overrightarrow{T}_k^2$ ($\overleftarrow{T}_k^2$) denote the times at which $\overrightarrow{D}1$ ($\overleftarrow{D}1$) and $\overrightarrow{D}2$ ($\overleftarrow{D}2$) deliver the extrinsic values of the kth ($(K+1-k)$th) estimated symbol of the original information sequence $\mathbf{u}$ and of the interleaved sequence $\tilde{\mathbf{u}}$, respectively. Let the extrinsic messages delivered by component decoders $\overrightarrow{D}1$ ($\overrightarrow{D}2$) and $\overleftarrow{D}1$ ($\overleftarrow{D}2$) be $\overrightarrow{L}_{e1}^{(i)}(\hat{u}_k)$ $\left(\overrightarrow{L}_{e2}^{(i)}(\hat{u}_k)\right)$ and $\overleftarrow{L}_{e1}^{(i)}(\hat{u}_k)$ $\left(\overleftarrow{L}_{e2}^{(i)}(\hat{u}_k)\right)$, respectively. For the forward recursion stage at the ith iteration of component decoder $\overrightarrow{D}1$, after time $\overrightarrow{T}_{k-1}^1$, the values of $\overrightarrow{\alpha}_k^{(1)}(s)$ should be updated and the values of $\overrightarrow{\gamma}_k^{(1)}(s)$ are needed. There are two possible cases. The first case is $k > \pi^-(k)$, which means the extrinsic value $\overrightarrow{L}_{e2}^{(i)}(\hat{u}_k)$ of the information bit $\hat{u}_k$ has already been delivered by decoder $\overrightarrow{D}2$. As in plain shuffled turbo decoding, this newly available $\overrightarrow{L}_{e2}^{(i)}(\hat{u}_k)$ is used to compute the values $\overrightarrow{\gamma}_k^{(1)}(s)$, $\overrightarrow{\alpha}_k^{(1)}(s)$, and $\overrightarrow{L}_{e1}^{(i)}(\hat{u}_k)$. The second case is $k < \pi^-(k)$, which means the extrinsic value $\overrightarrow{L}_{e2}^{(i)}(\hat{u}_k)$ of the information bit $\hat{u}_k$ has not been delivered yet by $\overrightarrow{D}2$. Then in the plain shuffled turbo decoding, the values $\alpha_k^{(1)}(s)$ and $L_{e1}^{(i)}(\hat{u}_k)$ are updated based on the extrinsic messages delivered at the last iteration. In replica shuffled turbo decoding, however, there are further two subcases. The first subcase is $K+1-k < \pi^-(k)$, which means the extrinsic value $\overleftarrow{L}_{e2}^{(i)}(\hat{u}_k)$ of the information bit $\hat{u}_k$ has already been delivered by decoder $\overleftarrow{D}2$. Then this newly available $\overleftarrow{L}_{e2}^{(i)}(\hat{u}_k)$, instead of the old $\overrightarrow{L}_{e2}^{(i)}(\hat{u}_k)$ is used to compute the values $\overrightarrow{\gamma}_k^{(1)}(s)$, $\overrightarrow{\alpha}_k^{(1)}(s)$, and $\overrightarrow{L}_{e1}^{(i)}(\hat{u}_k)$. The second subcase is $K+1-k < \pi^-(k)$, which means both extrinsic messages of the information bit $\hat{u}_k$, i.e, $\overleftarrow{L}_{e2}^{(i)}(\hat{u}_k)$ and $\overrightarrow{L}_{e2}^{(i)}(\hat{u}_k)$ are not available yet. In this subcase, the values of $\overrightarrow{\alpha}_k^{(1)}(s)$ and $\overrightarrow{L}_{e1}^{(i)}(\hat{u}_k)$ are updated based on the extrinsic messages delivered at the $(i-1)$th iteration. The recursions of component decoders $\overrightarrow{D}2$, $\overleftarrow{D}1$ and $\overleftarrow{D}2$ are realized based on the same principle. After I_{max} iterations, the shuffled turbo decoding algorithm outputs $\hat{\mathbf{u}} = (\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_K)$, where $\hat{u}_k =$sgn$[(\overrightarrow{L}_{e1}^{(i)}(\hat{u}_k) + \overleftarrow{L}_{e1}^{(i)}(\hat{u}_k))/2 + (\overrightarrow{L}_{e2}^{(i)}(\hat{u}_k) + \overleftarrow{L}_{e2}^{(i)}(\hat{u}_k))/2 + \frac{4}{N_0} y_{k,1}]$, which is different from that in the standard turbo decoding and plain shuffled turbo decoding. The decoding processes of the standard serial, parallel, plain shuffled and replica shuffled turbo decoding are illustrated in Fig. 1. It is straightforward to generalize the replica shuffled turbo decoding to multiple turbo codes which consist of more than two component codes. Based on the above descriptions, the total computational complexity of the replica shuffled turbo decoding for multiple turbo codes at each decoding iteration is about twice that of the parallel turbo decoding. Generalization to more than two replicas is also straightforward.

3. ANALYSIS OF PLAIN SHUFFLED AND REPLICA SHUFFLED TURBO DECODING

The EXIT chart technique [7]-[9] is an effective way to analyze the convergence behavior of iterative decoding. It illustrates the input/output mutual information relationship of a SISO decoder and can be used to analyze both turbo codes and LDPC codes. In [12], closed forms of EXIT functions of plain shuffled BP and replica shuffled BP decoding of LDPC codes have been presented. In this paper, plain shuffled and replica shuffled turbo decoding are analyzed using EXIT charts.

3.1. EXIT charts for parallel concatenated codes

Both channel observations and a priori knowledge can be modeled as conditional Gaussian random variables [7]. Denote L_o, L_a, and L_e the log-likelihood ratios (LLRs) of channel observation, a priori and extrinsic messages, respectively. Since we assume an AWGN channel, the received signal $y = c + n$ with $n \sim \mathcal{N}(0, \sigma_n^2)$. Then $L_o = \ln \frac{p(y|c=+1)}{p(y|c=-1)} = \frac{2}{\sigma_n^2}(c+n)$. Therefore

$$L_o|c \sim \mathcal{N}(\mu_o, \sigma_o^2) \tag{1}$$

where $\sigma_o^2 = 4/\sigma_n^2$ and $\mu_o = c\sigma_o^2/2$. Hence the consistency condition [14] is satisfied.

Assume the a priori input $A = \mu_A \cdot u + n_A$. Using a similar analysis, we get

$$L_a|u \sim \mathcal{N}(u\sigma_a^2/2, \sigma_a^2) \tag{2}$$

and the consistency condition is also satisfied. Denote I_a as the mutual information exchanged between L_a and u, and I_e as that exchanged between L_e and u. Since L_a is conditionally Gaussian and the consistency condition is satisfied, I_a is independent of the value of u. Therefore I_a can be written as a function of σ_a, say $J(\sigma_a)$ where

$$J(\sigma_a) = 1 - \int_{-\infty}^{\infty} \frac{\mathrm{e}^{-[(\xi-\sigma_a^2/2)^2/2\sigma_a^2]}}{\sqrt{2\pi}\sigma_a} \log_2(1+\mathrm{e}^{-\xi})d\xi. \tag{3}$$

Since we do not impose a Gaussian assumption on L_e, I_e is approximated based on the observation of N samples of L_e, so that

$$I_e \approx 1 - \frac{1}{N}\sum_{i=1}^{N} \log_2[1+\mathrm{e}^{-u_i L_{ei}}]. \tag{4}$$

The transfer function is defined as $I_e = T(I_a, E_b/N_0)$ and for a fixed E_b/N_0, it is just $I_e = T(I_a)$. The transfer functions of both decoders are plotted on a single chart. Since the extrinsic messages of the first decoder serve as the a priori messages of the second decoder, the axes are swapped for the transfer function of decoder-2.

3.2. Monte Carlo models for computing transfer functions

In [13, Chapter 9], a Monte Carlo model is used to derive the EXIT chart for a given turbo code. Its structure is shown in Fig. 2, with two Gaussian random noise generator outputs L_o and L_a whose distributions satisfy (1) and (2), respectively. Then L_o and L_a are sent to the SISO decoder, which outputs L_e. Through (3) and (4) I_a and I_e can be calculated. The transfer functions are obtained accordingly.

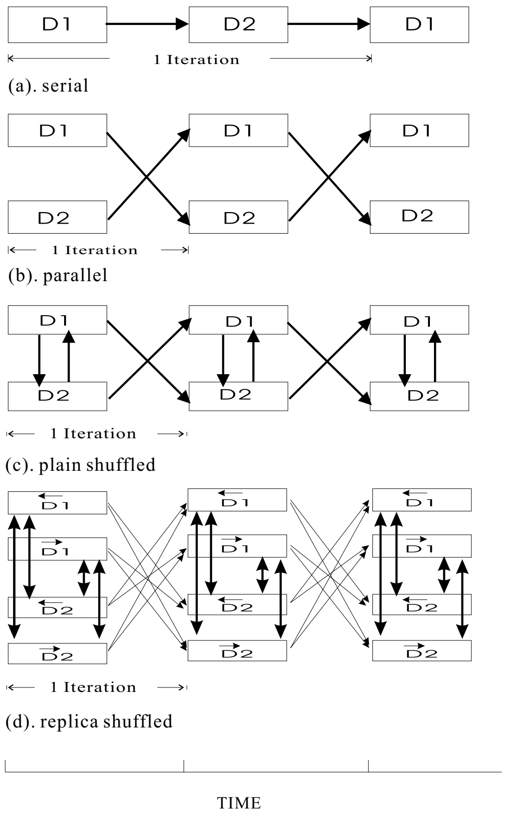
In plain shuffled turbo decoding, each decoder sends the newly updated extrinsic messages to the other decoder immediately after the updating. Hence we adopt three Gaussian random noise generators in the transfer function computing model, as shown in Fig. 3. The first two generators are identical to those in Fig. 2, while the third one takes the interleaved sequence $\tilde{\mathbf{u}}$ as input. The outputs of all these generators, L_o, L_{a1} and L_{a2}, are sent to the plain shuffled turbo decoders, where L_{a1} and L_{a2} are used as the a priori messages of decoder-1 and decoder-2, respectively. Then L_{e1} and L_{e2} are obtained and both of them are used to calculate I_e in (4).

For replica shuffled turbo decoding, the transfer function computing model is shown in Fig. 4. Since the four decoders, $\overrightarrow{D}1$, $\overrightarrow{D}2$, $\overleftarrow{D}1$ and $\overleftarrow{D}2$, exchange information synchronously, the newly updated a priori messages of $\overrightarrow{D}1$ and $\overleftarrow{D}1$ are the same after each iteration and so are those of $\overrightarrow{D}2$ and $\overleftarrow{D}2$. Therefore we still use three Gaussian random noise generators, but send L_{a1} to $\overrightarrow{D}1$ and $\overleftarrow{D}1$, and L_{a2} to $\overrightarrow{D}2$ and $\overleftarrow{D}2$, respectively. Since each decoder takes the extrinsic messages from two other decoders as its a priori messages, only the most recently updated extrinsic messages serve as the a priori messages in the next iteration. Hence it is more convenient to use the a priori LLRs for the next iteration, say L'_{a1} and L'_{a2}, to calculate I_e. Therefore in Fig. 4, we have the replica shuffled turbo decoder output L'_{a1} and L'_{a2} instead of $\overleftarrow{L}_{e1}$, $\overleftarrow{L}_{e2}$, $\overrightarrow{L}_{e1}$ and $\overrightarrow{L}_{e2}$. The values I_a and I_e are then calculated using the same formulas as before and the transfer functions are obtained afterward.

4. SIMULATION RESULTS

Fig. 5 depicts the EXIT charts of a turbo code with two component codes (rate-1/3) and interleaver size 16384, for standard parallel, plain shuffled, and replica shuffled turbo decoding at 0.15dB. We observe that the replica shuffled turbo decoding converges faster than both the parallel and plain shuffled turbo decoding.

Fig. 6 depicts the bit error performance of the same turbo code, with standard parallel, plain shuffled and replica shuffled decoding. After five iterations, the replica shuffled turbo decoder outperforms its parallel and plain counterparts by several tenths of a dB. Furthermore, at the SNR value 0.15dB, the BER of replica shuffled turbo decoding after five iterations is slightly worse than that of standard parallel turbo decoding after ten iterations, as predicted from the EXIT charts in Fig. 5.



(a). serial

(b). parallel

(c). plain shuffled

(d). replica shuffled

Fig. 1. The processes of serial, parallel and shuffled turbo decoding.

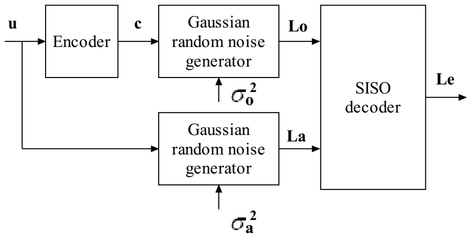

u
Encoder
c
Gaussian random noise generator
Lo
σ_o^2
Gaussian random noise generator
La
σ_a^2
SISO decoder
Le

Fig. 2. Monte Carlo model for computing the EXIT chart of a given turbo code with conventional turbo decoding.

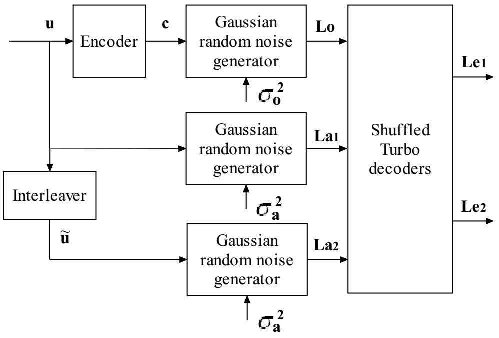


Fig. 3. Monte Carlo model for computing the EXIT chart of plain shuffled turbo decoding.

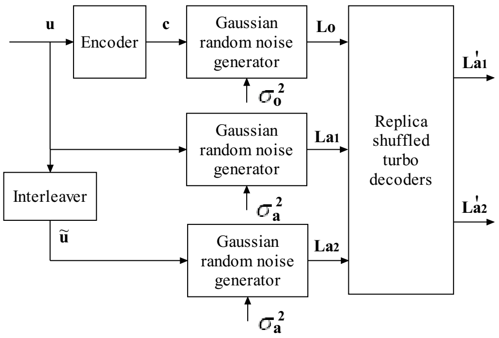

u
Encoder
c
Gaussian random noise generator
Lo
σ_o^2
Interleaver
Gaussian random noise generator
La1
σ_a^2
$\tilde{u}$
Gaussian random noise generator
La2
σ_a^2
Replica shuffled turbo decoders
La'1
La'2

Fig. 4. Monte Carlo model for computing the EXIT chart of replica shuffled turbo decoding.

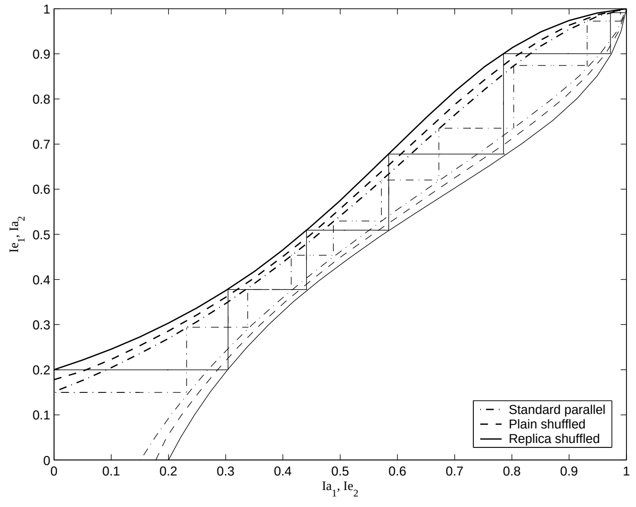


Fig. 5. EXIT charts of 2-component turbo code with interleaver size 16384, for standard parallel, plain shuffled, and replica shuffled turbo decoding, E_b/N_0=0.15dB.

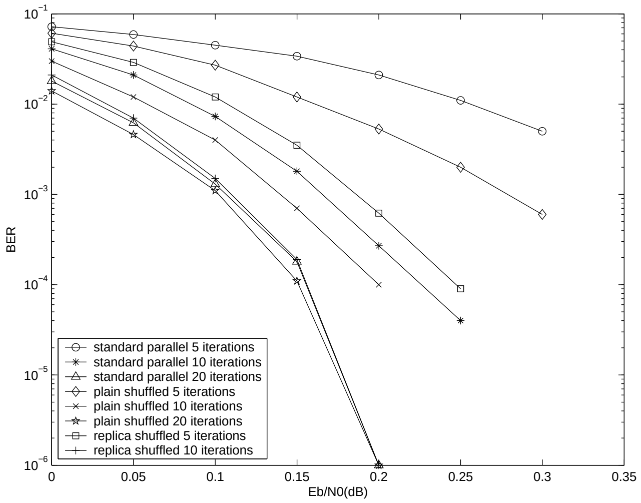


Fig. 6. Bit error performance of 2-component turbo code with interleaver size 16384, for standard parallel decoding, plain shuffled and replica shuffled decoding.

5. REFERENCES

[1] C. Berrou and A. Glavieux, “Near-optimum error-correcting coding and decoding: Turbo-codes,” *IEEE Trans. Commun.*, vol. 44, pp. 1261-1271, Oct. 1996.

[2] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, “Optimal decoding of linear codes for minimizing symbol error rate,” *IEEE Trans. Inform. Theory*, pp. 284-287, Mar. 1974.

[3] J. Zhang and M. Fossorier, “Shuffled iterative decoding,” *IEEE Trans. Commun.*, vol. 53, pp. 209-213, Feb. 2005.

[4] J. Zhang, Y. Wang, M. Fossorier, and J. S. Yedidia, “Replica shuffled iterative decoding of LDPC codes,” *Proc. 2005 Conf. Information Sciences and Systems*, to appear.

[5] J. Zhang, Y. Wang, M. Fossorier, and J. S. Yedidia, “Replica shuffled iterative decoding,” submitted to *2005 IEEE Int. Symp. Inform. Theory*.

[6] J. Zhang, Y. Wang, M. Fossorier, and J. S. Yedidia, “Replica shuffled iterative decoding,” *in preparation*.

[7] S. ten Brink, “Convergence behavior of iteratively decoded parallel concatenated codes,” *IEEE Trans. Inform. Theory*, vol. 49, pp. 1727-1737, Oct. 2001.

[8] M. Tüchler, S. ten Brink, and J. Hagenauer, “Measures for tracing convergence of iterative decoding algorithms,” in *Proc. 4th IEEE/ITG Conf. on Source and Channel Coding*, Berlin, Germany, pp. 53-60, Jan. 2002.

[9] M. Tüchler and J. Hagenauer, “EXIT charts of irregular codes,” in *Proc. 2002 Conf. Information Sciences and Systems*, Princeton, NJ, pp. 748-753, Mar. 2002.

[10] J. Hagenauer, E. Offer, and L. Papke, “Iterative decoding of binary block and convolutional codes,” *IEEE Trans. Inform. Theory*, vol. 42, pp. 429-445, Mar. 1996.

[11] D. Divsalar and F. Pollara, “Multiple turbo codes for deep-space communications,” *JPL TDA Progress Report*, pp. 66-77, May 1995.

[12] Y. Wang, J. Zhang, M. Fossorier, and J. S. Yedidia, “Reduced latency iterative decoding of LDPC codes,” submitted to *IEEE Global Telecommun. Conf. 2005*.

[13] E. Biglieri, *Coding of Wireless Channels*. Springer, preprint.

[14] T. Richardson, A. Shokrollahi, and R. Urbanke, “Design of provably good low-density parity-check codes,” in *Proc. 2000 IEEE Int. Symp. Inform. Theory*, Sorrento, Italy, p. 199, Jun. 2000.