MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Replica Shuffled Iterative Decoding

Zhang, J.; Wang, Y.; Fossorier, M.; Yedidia, J.S.

TR2005-063 September 2005

Abstract

Replica shuffled versions of interactive decoders of turbo codes, low-density parity-check codes and turbo product codes are presented. The proposed schemes converge faster than standard and previously proposed shuffled approaches. Simulations show that the new schedules offer good performance versus complexity/latency trade-offs.

IEEE International Symposium on Information Theory (ISIT)

This work may not be copied or reproduced in whole or in part for any commercial purpose. Permission to copy in whole or in part without payment of fee is granted for nonprofit educational and research purposes provided that all such whole or partial copies include the following: a notice that such copying is by permission of Mitsubishi Electric Research Laboratories, Inc.; an acknowledgment of the authors and individual contributions to the work; and all applicable portions of the copyright notice. Copying, reproduction, or republishing for any other purpose shall require a license with payment of fee to Mitsubishi Electric Research Laboratories, Inc. All rights reserved.

Copyright © Mitsubishi Electric Research Laboratories, Inc., 2005
201 Broadway, Cambridge, Massachusetts 02139

MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Replica Shuffled Iterative Decoding

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

TR-2005-063 February 2005

Abstract

Replica shuffled versions of iterative decoders of turbo codes, low-density parity-check codes and turbo product codes are presented. The proposed schemes converge faster than standard and previously proposed "shuffled" approaches. Simulations show that the new schedules offer good performance versus complexity/latency trade-offs.

This work may not be copied or reproduced in whole or in part for any commercial purpose. Permission to copy in whole or in part without payment of fee is granted for nonprofit educational and research purposes provided that all such whole or partial copies include the following: a notice that such copying is by permission of Mitsubishi Electric Research Laboratories, Inc.; an acknowledgment of the authors and individual contributions to the work; and all applicable portions of the copyright notice. Copying, reproduction, or republishing for any other purpose shall require a license with payment of fee to Mitsubishi Electric Research Laboratories, Inc. All rights reserved.

Copyright © Mitsubishi Electric Research Laboratories, Inc., 2005
201 Broadway, Cambridge, Massachusetts 02139

To appear in the Proceedings of the 2005 International Symposium on Information Theory.

Replica Shuffled Iterative Decoding

Juntan Zhang, Yige Wang and Marc Fossorier
Department of Electrical Engineering
University of Hawaii at Manoa
Honolulu, HI 96822
Email: juntan, yige, marc@spectra.eng.hawaii.edu

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

***Abstract*— Replica shuffled versions of iterative decoders of turbo codes, low-density parity-check codes and turbo product codes are presented. The proposed schemes converge faster than standard and previously proposed "shuffled" approaches. Simulations show that the new schedules offer good performance versus complexity/latency trade-offs.**

I. Introduction

Iterative decoding based on belief propagation (BP) [1] has received significant attention recently, mostly due to its near-Shannon-limit error performance for the decoding of low-density parity-check (LDPC) codes [2] and turbo codes [3]. Like maximum a posterior probability (MAP) decoding [4], it is a symbol-by-symbol soft-in/soft-out decoding algorithm. It processes the received symbols recursively to improve the reliability of each symbol based on constraints that specify the code. In the first iteration, the decoder only uses the channel output, and generates 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 the decoded symbols at the last decoding iteration. The standard BP decoder of LDPC codes often needs several tens or hundreds of iterations for the decoding process to converge, which is not desirable because of high decoding delay. Furthermore, LDPC codes of interest can have large codeword length and it can be difficult to implement the decoding in hardware in a fully parallel way. Because of the serial property of the BCJR algorithm, decoders of turbo codes cause high delay as well. In [5], [6] and [7], "shuffled" methods were presented to reduce the required number of iterations for decoding LDPC and turbo codes, respectively. A speedup factor of 2 for LDPC codes and the saving of one iteration for turbo codes were reported. The aim of this paper is to introduce a "replica shuffled" scheme which further accelerate the decoding process for turbo codes, LDPC codes, turbo product codes and other iterative decodable codes.

II. Iterative decoding of turbo code

A turbo code [3] encoder is constructed using a 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 to each other. 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 $\mathbf{c} = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K)$ be the corresponding coded sequence, where $\mathbf{c}_k = (c_{k,1}, c_{k,2}, \ldots, c_{k,n})$, for $k = 1, 2, \ldots, K$, is the output code block at time k. Suppose 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 [4], 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 [8].

A. Standard serial and parallel turbo decoding

The decoding approach proposed in [3] operates in serial mode, i.e., the component decoders take turns generating the extrinsic values of the estimated information symbols, and each component decoder uses the extrinsic messages delivered by the last component decoder as the a priori values of the information symbols. The disadvantage of this scheme is high decoding delay. In the parallel turbo decoding algorithm [9], 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.

B. Plain shuffled turbo decoding

Although the parallel turbo decoding reduces the decoding delay of serial decoding by half, the extrinsic messages are not taken advantage of as soon as they become available, because the extrinsic messages are delivered to component decoders only after each iteration is completed. The aim of the shuffled turbo decoding is to use the more reliable extrinsic messages at each time. Let $\tilde{\mathbf{u}} = (\tilde{u}_1, \tilde{u}_2, \ldots, \tilde{u}_K)$ be the sequence permuted by the interleaver corresponding to the original information sequence $\mathbf{u} = (u_1, u_2, \ldots, u_K)$, 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 $k \neq \pi^-(k), \forall k$. In shuffled turbo decoding, the two component decoders operate simultaneously as in the parallel turbo decoding scheme, but the messages are updated during each iteration based on $\pi(k)$ and $\pi^-(k)$ [7]. Correspondingly it provides a faster decoding convergence.

C. Replica shuffled turbo decoding

In the plain shuffled turbo decoding summarized in Section II-B, we assume all the component decoders process the backward recursion followed by the forward recursion. Let us refer to the two component decoders as $\overrightarrow{D}_1$ and $\overrightarrow{D}_2$. Naturally another possible scheme is to operate in the reverse order, i.e, all the component decoders process the forward recursion followed by the backward recursion and we refer to them as $\overleftarrow{D}_1$ and $\overleftarrow{D}_2$. In terms of error performance, there is no difference between these two approaches. However, the reliabilities of the extrinsic messages concerning a certain information bit delivered by these two shuffled turbo decoders are not the same. In general, the more independent information is used, the more reliable the delivered messages become. Therefore for the extrinsic messages delivered by component decoder $\overrightarrow{D}_1$, which are denoted by $\overrightarrow{L}_{e1}^{(i)}(\hat{u}_k)$, the larger k is, the more reliable this message is. Following a similar analysis, for the extrinsic message $\overleftarrow{L}_{e1}^{(i)}(\hat{u}_k)$ delivered by $\overleftarrow{D}_1$, the smaller k is, the more reliable this message is. It is natural to expect a faster decoding convergence if these two shuffled turbo decoders operate cooperatively instead of independently. Because in this approach two sets of shuffled component decoders are used to decode the same sequence of information bits, we refer to it as replica shuffled turbo decoding. In replica shuffled turbo decoding, two plain shuffled turbo decoders (processing recursions in opposite directions) $\overrightarrow{D}_1$, $\overrightarrow{D}_2$ and $\overleftarrow{D}_1$, $\overleftarrow{D}_2$ operate simultaneously and exchange more reliable extrinsic messages. We assume that the component decoders deliver extrinsic messages synchronously, 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 k-th ($(K+1-k)$-th) estimated symbol of the original information sequence $\mathbf{u}$ and of the interleaved sequence $\tilde{\mathbf{u}}$, respectively.

Let us first consider the forward recursion stage at the i-th 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 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 last iteration. In replica shuffled turbo decoding, however, there are two further 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 $\overrightarrow{L}_{e2}^{(i-1)}(\hat{u}_k)$ 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 $(i-1)$-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 $\hat{\mathbf{u}} = (\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_K)$ as the decoded codeword, 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 [3] and plain shuffled turbo decoding. It is straightforward to generalize the replica shuffled turbo decoding to multiple turbo codes which consist of more than two component codes. Also group of bits can be updated periodically only to reduce information exchanges between replicas. Based on the above descriptions with two replicas, 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.

D. Simulation results

Fig. 1 depicts the bit error performance of a turbo code with two component codes (rate-1/3) and interleaver size 16384, with standard parallel decoding, plain shuffled and replica shuffled decoding. We observe that, to obtain the same error performance, the replica turbo decoding requires about half the number of iterations of that of the standard parallel turbo decoding. Hence latency is greatly reduced (by half if information exchanges are ignored).

III. Iterative decoding of LDPC codes

Let $\mathbf{H} = [H_{mn}]$ be the parity check matrix which defines an LDPC code. We denote the set of bits that participate in check m by $\mathcal{N}(m) = \{n : H_{mn} = 1\}$ and the set of checks in which bit n participates as $\mathcal{M}(n) = \{m : H_{mn} = 1\}$. Assume a codeword $\mathbf{w} = (w_1, w_2, \ldots, w_N)$ is transmitted over an AWGN channel with zero mean and variance $N_0/2$ using BPSK signaling and let $\mathbf{y} = (y_1, y_2, \ldots, y_N)$ be the corresponding received sequence.

A. Standard BP for iterative decoding of LDPC codes

Based on [1], let F_n be the log-likelihood ratio (LLR) of bit n and initially set $F_n = \frac{4}{N_0} y_n$. Let $\varepsilon_{mn}^{(i)}$ and $z_{mn}^{(i)}$ be the LLR of bit n which is sent from check node m to bit node n and sent from the bit node n to check node m, respectively. Let $z_n^{(i)}$ denote the *a posteriori* LLR of bit n. The standard BP algorithm [1] is carried out as follows:

Initialization Set $i = 1$, maximum number of iterations to I_{Max}. For each m,n, set $z_{mn}^{(0)} = F_n$.

Step 1

(i) Horizontal Step, for $1 \leq n \leq N$ and each $m \in \mathcal{M}(n)$, process:

$$\tau_{mn}^{(i)} = \prod_{n' \in \mathcal{N}(m)\backslash n} \tanh(z_{mn'}^{(i-1)}/2) \quad (1)$$

$$\varepsilon_{mn}^{(i)} = \log \frac{1+\tau_{mn}^{(i)}}{1-\tau_{mn}^{(i)}} \quad (2)$$

(ii) Vertical Step, for $1 \leq n \leq N$ and each $m \in \mathcal{M}(n)$, process:

$$z_{mn}^{(i)} = F_n + \sum_{m' \in \mathcal{M}(n)\backslash m} \varepsilon_{m'n}^{(i)} \quad (3)$$

$$z_{n}^{(i)} = F_n + \sum_{m \in \mathcal{M}(n)} \varepsilon_{mn}^{(i)} \quad (4)$$

Step 2 Hard decision and stopping criterion test:

(i) Create $\hat{\boldsymbol{w}}^{(i)} = [\hat{w}_n^{(i)}]$ such that $\hat{w}_n^{(i)} = 0$ if $z_n^{(i)} > 0$, and $\hat{w}_n^{(i)} = 1$ if $z_n^{(i)} < 0$.

(ii) If $\boldsymbol{H}\hat{\boldsymbol{w}}^{(i)} = \boldsymbol{0}$ or I_{Max} is reached, stop the decoding and go to Step 3. Otherwise set $i := i + 1$ and go to Step 1.

Step 3 Output $\hat{\boldsymbol{w}}^{(i)}$ as the decoded codeword.

B. Plain shuffled BP for iterative decoding of LDPC codes

At the i-th iteration of the standard BP algorithm, first all values of the check-to-bit messages are updated by using the values of the bit-to-check messages obtained at the $(i-1)$-th iteration, i.e., each $\varepsilon_{mn}^{(i)}$ is updated by using $\{z_{mn'}^{(i-1)} : n' \in \mathcal{N}(m)\backslash n\}$. Then, all values of the bit-to-check messages are updated by using the values of the check-to-bit messages newly obtained at the i-th iteration, i.e., each $z_{mn}^{(i)}$ is updated from $\{\varepsilon_{m'n}^{(i)} : m' \in \mathcal{M}(n)\backslash m\}$.

We observe that certain values $z_{mn'}^{(i)}$ could already be computed in (3) based on a partial computation of the values $\varepsilon_{mn}^{(i)}$ obtained from (2), and then could be used instead of $z_{mn'}^{(i-1)}$ in (1) to compute the remaining values $\varepsilon_{mn}^{(i)}$. Hence Step 1 of the shuffled BP algorithm is performed as: for $1 \leq n \leq N$ and each $m \in \mathcal{M}(n)$, process the horizontal step and vertical step **jointly**, with (1) modified as:

$$\tau_{mn}^{(i)} = \prod_{\substack{n' \in \mathcal{N}_{(m)\backslash n} \\ n'<n}} \tanh(z_{mn'}^{(i)}/2) \prod_{\substack{n' \in \mathcal{N}_{(m)\backslash n} \\ n'>n}} \tanh(z_{mn'}^{(i-1)}/2)$$

Since this algorithm becomes totally serial, bits can be grouped to maintain a sufficient level of parallelism with the same error performance [7].

C. Replica shuffled BP decoding for LDPC codes

Plain shuffled BP decoding is a bit-based sequential approach and the method described in Section III-B is based on a natural increasing order, i.e, belief messages concerning bit nodes are updated according to order $i = 1, 2, \ldots, N$. The larger the value of i, the more independent informations are used to update the beliefs of bit note i and the more reliable these belief messages are. Therefore the reliability of bit nodes increases and the error rate decreases as i increases. Following a similar analysis, in plain shuffled BP decoding based on a natural decreasing order, after each iteration, the reliability of bit nodes decreases as i increases. As in replica shuffled turbo decoding, in replica shuffled BP decoding, two replica shuffled subdecoders based on different updating orders operate simultaneously and cooperatively. After each iteration, each subdecoder receives more reliable messages from and sends more reliable messages to another subdecoder. Based on these more reliable messages, both replica subdecoders begin the next iteration of decoding. Let $\overrightarrow{D}$ and $\overleftarrow{D}$ denote the replica subdecoder with natural increasing and decreasing updating order, respectively. Let $\overrightarrow{\varepsilon}_{mn}^{(i)}$ and $\overrightarrow{z}_{mn}^{i}$ be the variables associated with $\overrightarrow{D}$ at iteration i. Variables associated with $\overleftarrow{D}$ are defined in a similar way. The replica shuffled BP decoding with two replica subdecoders is carried out as followings:

Initialization Set $i = 1$, maximum number of iteration to I_{Max}. For each m,n, set $\overrightarrow{z}_{mn}^{(0)} = \overleftarrow{z}_{mn}^{(0)} = F_n$.

Step 1 Each replica subdecoder process as the following two steps simultaneously. For $0 \leq n \leq N-1$ and each $m \in \mathcal{M}(n)$, process

(i) Horizontal Step

$$\overrightarrow{\tau}_{mn}^{(i)} = \prod_{\substack{n' \in \mathcal{N}_{(m)\backslash n} \\ n'<n}} \tanh(\overrightarrow{z}_{mn'}^{(i)}/2) \cdot \prod_{\substack{n' \in \mathcal{N}_{(m)\backslash n} \\ n'>n}} \tanh(\overrightarrow{z}_{mn'}^{(i-1)}/2)$$

$$\overrightarrow{\varepsilon}_{mn}^{(i)} = \log \frac{1+\overrightarrow{\tau}_{mn}^{(i)}}{1-\overrightarrow{\tau}_{mn}^{(i)}}$$

$$\overleftarrow{\tau}_{mn}^{(i)} = \prod_{\substack{n' \in \mathcal{N}_{(m)\backslash n} \\ n'>n}} \tanh(\overleftarrow{z}_{mn'}^{(i)}/2) \cdot \prod_{\substack{n' \in \mathcal{N}_{(m)\backslash n} \\ n'<n}} \tanh(\overleftarrow{z}_{mn'}^{(i-1)}/2)$$

$$\overleftarrow{\varepsilon}_{mn}^{(i)} = \log \frac{1+\overleftarrow{\tau}_{mn}^{(i)}}{1-\overleftarrow{\tau}_{mn}^{(i)}}$$

(ii) Vertical Step

$$\overrightarrow{z}_{mn}^{(i)} = F_n + \sum_{m' \in \mathcal{M}(n)\backslash m} \overrightarrow{\varepsilon}_{m'n}^{(i)}$$

$$\overleftarrow{z}_{mn}^{(i)} = F_n + \sum_{m' \in \mathcal{M}(n)\backslash m} \overleftarrow{\varepsilon}_{m'n}^{(i)}$$

Step 2 Exchange of the more reliable messages. Set $\overrightarrow{z}_{mn}^{(i)} = \overleftarrow{z}_{mn}^{(i)}$ for $0 \leq n < N/2$ and $\overleftarrow{z}_{mn}^{(i)} = \overrightarrow{z}_{mn}^{(i)}$ for $N/2 \leq n \leq N-1$.

Step 3 Hard decision and stopping criterion test:

(i) Create $\hat{\boldsymbol{w}}^{(i)} = [\hat{w}_n^{(i)}]$ such that for $0 \le n < N/2$, $\hat{w}_n^{(i)} = 1$ if $F_n + \sum_{m \in \mathcal{M}(n)} \overleftarrow{\varepsilon}_{mn}^{(i)} < 0$, and $\hat{w}_n^{(i)} = 0$ otherwise; for $N/2 < n \le N-1$, $\hat{w}_n^{(i)} = 1$ if $F_n + \sum_{m \in \mathcal{M}(n)} \overrightarrow{\varepsilon}_{mn}^{(i)} < 0$, and $\hat{w}_n^{(i)} = 0$ otherwise.

(ii) If $\boldsymbol{H}\hat{\boldsymbol{w}}^{(i)} = \boldsymbol{0}$ or I_{Max} is reached, stop the decoding and go to Step 4. Otherwise set $i := i + 1$ and go to Step 1.

Step 4 Output $\hat{\boldsymbol{w}}^{(i)}$ as the decoded codeword.

An alternative approach that can be used is to make the two subdecoders exchange the more reliable messages after updating the reliability message of each bit (or a group of bits). We observe that this "simultaneous replica" approach provides a faster convergence, especially for Gallager-type LDPC codes. Thus all simulation results of LDPC codes are based on simultaneous updating of each bit (or a group of bits). It is also straightforward to extend the replica shuffled BP decoding to the cases in which more than two replica subdecoders are used. In order to decrease decoding delay of plain shuffled BP decoding, a parallel version of shuffled BP named group shuffled BP was developed in [7]. In a similar way, group replica shuffled BP can also preserve the parallelism advantage of the standard BP algorithm. It is also clear this idea can be extended to other other grouping scheme (e.g, [11]) and other iterative decoding algorithms, such as bit flipping and weighted bit flipping decoding.

D. Simulation results

Fig. 2 depicts the word error rate (WER) of iterative decoding of a $(8000, 4000)(3, 6)$ LDPC code, with the standard BP, plain shuffled and group replica shuffled BP algorithm, for $G = 2$, 4, 8, 16 and 8000 and with four replica subdecoders. The maximum number of iterations for plain and group replica shuffled BP was set to be 10. We observe that the WER performance of replica shuffled BP decoding with four subdecoders, I_{max}=10, and a group number larger or equal to four, are approximately the same as that of the standard BP with I_{max}=60. Fig. 3 depicts the WER of the standard and replica shuffled BP decoding of the $(16200, 7200)$ irregular LDPC code which is constructed in a semi-random matter [12]. The variable node degree distribution is $\lambda(x) = 0.00006x + 0.57772x^2 + 0.3111x^3 + 0.11111x^8$. The number of replica subdecoders was four. We observe that the replica shuffled BP with I_{max}=10 and $G = 32$ provides a similar performance as that of the standard BP with I_{max}=70.

IV. Iterative decoding of turbo product codes

A two-dimension turbo product code (TPC) can be denoted as $C_1 \bigotimes C_2$, where C_1 and C_2 are two linear block codes. Place $k_1 \times k_2$ information symbols in an array of k_1 rows and k_2 columns, and then encode the k_1 rows using code C_2. Afterwards, the resulting n_2 columns are encoded using code C_1. Usually, we choose C_1 the same as C_2.

A. Conventional decoding methods of TPC

The conventional TPC decoder performs row and column decoding in a serial fashion. A soft input soft output (SISO) decoder, such as MAP, is used to decode each row or column. A low complexity decoding approach is provided in [13]. It applies the Chase algorithm iteratively on the row and column decoding, but still in a serial fashion. In order to halve the decoding latency, a parallel TPC decoder has been proposed in [10]. As opposed to the conventional serial TPC decoder, the row and column decoders in this method operate in parallel and send each other the updated extrinsic information immediately after a row or column has been decoded. The simulation results reveal that this parallel decoder can reduce decoding latency by half of that of the original decoder.

B. Replica decoding of TPC

Based on the parallel decoder, we propose a replica parallel decoder as shown in Fig. 4, which can further reduce decoding latency. Both row and column decoders are duplicated, but work from opposite extremes, which means the two row decoders process rows from the top and bottom, respectively and the two column decoders process columns from the left and right, respectively. Row (Column) decoders send to both column (row) decoders immediately the latest extrinsic matrices, $[\boldsymbol{W}_T^{col}]$ and $[\boldsymbol{W}_B^{col}]$ ($[\boldsymbol{W}_L^{row}]$ and $[\boldsymbol{W}_R^{row}]$), after a row (column) has been decoded.

The procedure of the replica parallel decoding is shown in Fig. 5. The circles denote bit positions that were already updated by other decoders. Their number is much larger than that in the parallel decoder, which greatly benefits the decoding because most bits have more accurate priori information. The arrows with letters a_T, b_T, etc., represent the processing order of different decoders. After both row (column) decoders finish decoding all the rows (columns), the most reliable parts of them are combined and the resulting extrinsic matrix for the next iteration is transmitted to both column (row) decoders.

C. Simulation results

Fig. 6 depicts the performance of TPC $(64, 57, 4)^2$. The weighting factors with the iteration number are $[0.1, 0.2, 0.3, 0.5, 0.7, 0.9, 1.0, 1.0]$. The MAP algorithm is used to decode the two linear block codes. As shown in Fig. 6, the performance at the third iteration in the replica decoder is better than that at the fourth iteration in the parallel decoder.

References

[1] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399-431, Mar. 1999.

[2] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA: M.I.T. Press, 1963.

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

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

[5] H. Kfir and I. Kanter, "Parallel versus sequential updating for belief propagation decoding," submitted to *Physical Review E*, July 2002.

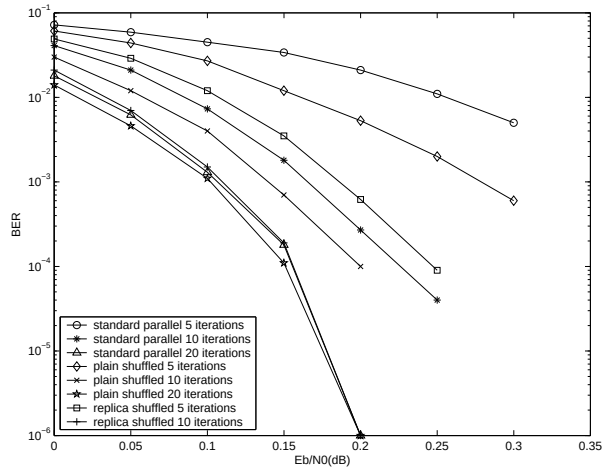

Fig. 1. Bit error performance of 2-component turbo code with interleaver size 16384, for standard parallel, plain shuffled and replica shuffled decodings.

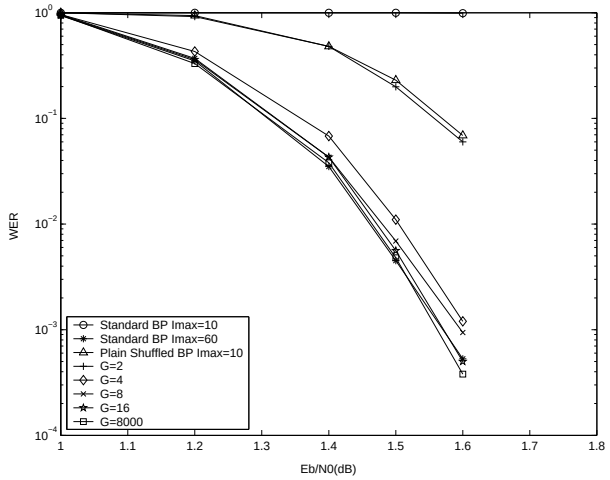

Fig. 2. Error performance for iterative decoding of the $(8000, 4000)(3, 6)$ LDPC code with group shuffled BP algorithm, for $G = 1, 2, 8, 100, 8000$ and at most 10 iterations.

[6] J. Zhang and M. Fossorier, "Shuffled Belief Propagation Decoding," *Proceedings of 36th Annual Asilomar Conference on Signals, Systems and Computers*, pp. 8-15, Nov. 2002.

[7] J. Zhang and M. Fossorier, "Shuffled Belief Propagation Decoding," *IEEE Trans. Commun.*, Feb. 2005.

[8] J. Hagenauer, E. Offer and L. Papke, "Iterative decoding of block and convolutional codes," *IEEE Trans. on Inform.*, vol. 42, pp. 429-445, March 1996.

[9] D. Divsalar and F. Pollara, "Multiple turbo codes for deep-space communications," *JPL TDA Progress Report*, 71-78, May 1995.

[10] C. Argon and S. McLaughlin, "A parallel decoder for low latency decoding of turbo product codes," *IEEE Commun. Lett.*, vol. 6, pp. 70-72, Feb. 2002.

[11] E. Yeo, P. Pakzad, B. Nikolic and V. Anantharam, "High throughput low-density parity-check decoder architectures," *Proceedings of Global Telecommunications Conference*, vol. 5, pp. 3019-3024, Nov. 2001.

[12] "Draft DVB-S2 Standard," availible at *http://www.dvb.org*.

[13] R. M. Pyndiah, "Near-optimum decoding of product codes: Block turbo codes", *IEEE Trans. Commun.*, vol. 46, pp. 1003-1010, Aug. 1998.

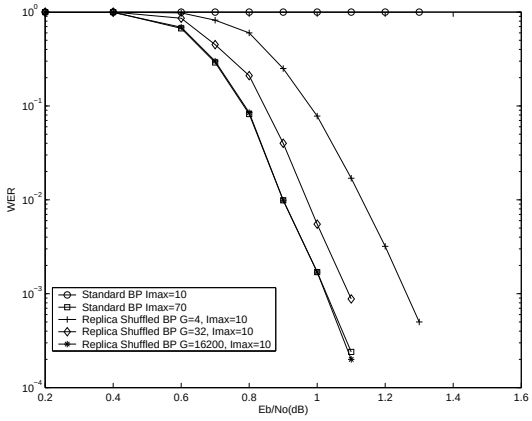

Fig. 3. Error performance for iterative decoding of the $(16200, 7200)$ irregular LDPC code.

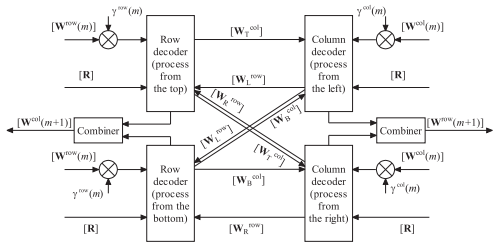

Fig. 4. Replica parallel decoder (one iteration)

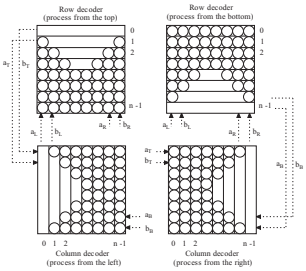

Fig. 5. Procedure of replica parallel decoder

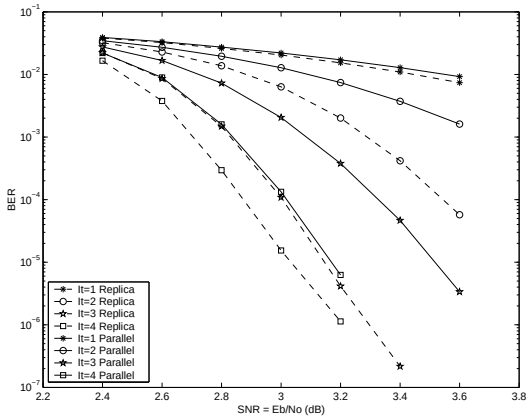

Fig. 6. Performance of TPC $(64, 57, 4)^2$ with parallel and replica parallel decoders.