# Improvements In SOVA-Based Decoding For Turbo Codes

*Lang Lin, Roger S. Cheng*[*]
*Dept. of Electrical and Electronic Engineering*
*Hong Kong University of Science & Technology,*
*Clear Water Bay, Kowloon, HONG KONG*

## ABSTRACT

*In this paper, we propose two modifications to soft output Viterbi algorithm (SOVA) for Turbo Code decoding. One is to limit the reliability values to a small range to compensate for the defect brought by overestimating those values in the original SOVA. The other is to employ a new block interleaver to combat the tail effect of SOVA-Based Turbo Code decoding. The simulation results show that the new SOVA with both modifications is able to obtain the similar result achieved by a maximum a posteriori (MAP) algorithm with a random interleaver. In this paper, we also provide the results of the SOVA with the Battail's updating rule and compare them to those of the SOVA with the updating rule proposed by Hagenauer.*

## 1 INTRODUCTION

Applying soft output Viterbi algorithm (SOVA) to decode Turbo Codes was first detailedly described in [1]. Comparing with maximum a posteriori (MAP) algorithm, the less complex SOVA is more suitable for hardware implementation. A Turbo Code codec chip using SOVA has already been reported in [2]. Unfortunately, the SOVA-based decoding is generally inferior to the MAP-based decoding in terms of BER performance [1, 3]. Therefore, improving the performance of the SOVA-based decoding is very important to the future application of Turbo Codes.

In fact, some good results in this area have already been available. In [3], the authors showed that the original SOVA introduces two types of distortion in its output. Its performance can be improved by normalizing its output and eliminating the correlation between the intrinsic and the extrinsic information. However, the normalization method in [3] requires the knowledge of the signal-to-noise ratio (SNR) and is not very simple.

The difference between SOVA and conventional VA is that during decoding, the former one produces 'soft' bit reliability values according to an specific updating rule. In the literature, there are two updating rules. One was proposed by Battail [4] and the other by Hagenauer [1]. In this paper, we denote the Battail's updating rule by BR and the Hagenauer's by HR. Consequently, we denote the SOVA with BR by BR-SOVA and the one with HR by HR-SOVA. Our results show that comparing with BR-SOVA, HR-SOVA gives larger reliability values and worse performance. This is mainly because HR-SOVA omits some updates for ease of implementation, leading to overestimation of the reliability values. However, it is important to note that HR-SOVA has the advantage that it can be implemented readily with some VA structures [8, 9].

In this paper, motivated by reducing the reliability values in the HR-SOVA, we propose to put a range limitation on the reliability values. The simulation results show that the SOVA with this scheme is able to bring a 0.5 dB gain over the original one at BER of $10^{-4}$ for a rate 1/3 16-state Turbo Code with a 400-bit frame and the traditional block interleaver. Meanwhile, there is no requirement of any knowledge about the SNR and the implementation is very straightforward.

When decoding only one frame of data at a time, the bits near the end of a frame are very likely to be associated with inaccurate reliability values. We refer to this effect as the tail effect which can degrade the performance of the SOVA decoding seriously. The second modification we propose is to utilize a rotated block interleaver in Turbo Codes to combat the tail effect.

The simulation results show that our modified SOVA produce performance similar to that achieved by a maximum a posteriori (MAP) algorithm with a random interleaver reported recently [11].

In section 2, we offer a short review of the SOVA for Turbo Code decoding. In section 3, the performance of two

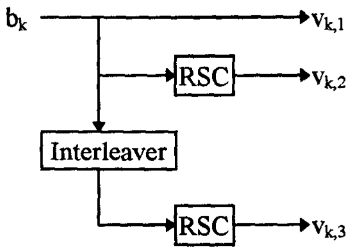

Figure 1: 1/3 Turbo Code encoder

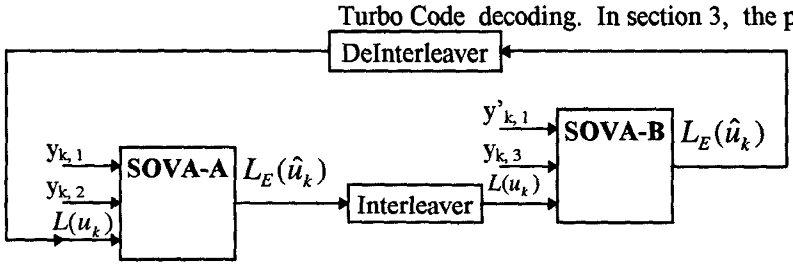

Figure 2: Turbo Code decoder structure

[*]Corresponding Author: Roger S. Cheng, Department of Electrical and Electronic Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong. Tel: (852) 2358 7072. Fax: (852) 2358 1485. Email: eecheng@ee.ust.hk.

different updating rules, BR and HR, are compared. Our results show that comparing with BR-SOVA, HR-SOVA gives a larger average reliability value and worse performance. In section 4, we give the simulation results for different range limitations. In section 5, the detail of our rotated block interleaver is given. At the end of this paper, we provide the simulation results of the SOVA with both modifications.

## 2 TURBO CODES CODEC AND SOVA

### 2.1 Turbo Codes Codec Structure

Figure 1 shows the structure of a rate 1/3 2-consituent-code Turbo Code encoder. In this paper, we consider two Turbo Codes and each of them has two identical constituent codes which are recursive systematic convolutional (RSC) codes. The constituent codes used in these two Turbo Codes are the 4-state code with generators (5, 7) and the 16-state with generators (23, 35), respectively. These codes provide the largest effective distance [5]. In the encoder, we denote by $b_k$ the kth information bit and $v_k=(v_{k,1}, v_{k,2}, v_{k,3})$ the kth 3-bit output vector.

Figure 2 shows the decoder structure. We denote by $y_k=(y_{k,1}, y_{k,2}, y_{k,3})$ the kth received vector and $y'_{k,1}$ is the interleaved version of $y_{k,1}$. The output of a SOVA decoder is the extrinsic information, $L_E(\hat{u}_k)$ and the corresponding decoded bit is $\hat{u}_k$ which is not output at every decoding stage. The extrinsic information, after interleaving or deinterleaving, will be interpreted as the a priori information, $L(u_k)$, in the next decoding stage.

### 2.2 Soft Output Viterbi Algorithm

SOVA is a modified Viterbi algorithm with additional output values associated with the original decoded bit sequence. It was formerly used in serial concatenated coding scheme [6]. When used in Turbo Codes decoding, it was further revised according to [1, 7].

The decoding procedure of SOVA can be divided into 3 steps:

#### 2.2.1 APRI-VA

In this step, the soft decision Viterbi algorithm is performed. The probability of the path m at time k is defined as

$$p_k^m = Ce^{M_k^m/2}, \qquad (1)$$

where $C$ is a constant and $M_k^m$ is the path metric. In an AWGN channel[1], the accumulative path metric of a rate 1/N code is updated according to

$$M_k^m = M_{k-1}^m + \sum_{n=1}^{N} x_{k,n}^m L_c y_{k,n} + x_{k,1}^m L(u_k), \qquad (2)$$

where x and y are the corresponding code word sequence and the received sequence respectively, and

$$L_c = 4\frac{E_s}{N_0}. \qquad (3)$$

Therefore, the probability of choosing a wrong path is

$$\Phi_k^m = \frac{p_k^{m'}}{p_k^{m'} + p_k^m} = \frac{1}{1+e^{(M_k^m - M_k^{m'})/2}}, \qquad (4)$$

where $M_k^{m'}$ and $p_k^{m'}$ indicate the path metric and the probability of the concurrent path. The log likelihood ratio or "soft" value of this path decision is

$$\Delta_k^m = \log\frac{1-\Phi_k^m}{\Phi_k^m} = (M_k^m - M_k^{m'})/2. \qquad (5)$$

#### 2.2.2 UPDATING LOG-LIKELIHOOD RATIO

The APRI-VA provides not only a hard output sequence, but also "soft" path reliability values. In order to get the reliability value or log-likelihood ratio for each output bit, $L(\hat{u}_k)$, an updating procedure is required. The detailed description and discussion of the updating procedure are postponed until the next section.

#### 2.2.3 SUBTRACTING INTRINSIC INFORMATION

The $L(\hat{u}_k)$ is made of two parts. The first part, the *intrinsic* information $L_I(\hat{u}_k)$, is composed of a channel value $L_c y$ and the a priori value $L(u_k)$ which is the contribution of the last decoding stage. The second part is the contribution of this decoding stage called *extrinsic* information, $L_E(\hat{u}_k)$, which is fed into the next decoding stage as the new a priori estimate. Therefore, a subtraction,

$$L_E(\hat{u}_k) = L(\hat{u}_k) - L_I(\hat{u}_k), \qquad (6)$$

is required in the end of a decoding stage.

In fact, after carefully examining the 3 steps listed above, we find that SOVA is independent of $L_c$, the only parameter that requires the knowledge of SNR. At the first iteration where $L(u_k)$=0, (2) can be rewritten as

$$M_k^m = M_{k-1}^m + L_c\sum_{n=1}^{N} x_{k,n}^m y_{k,n}. \qquad (7)$$

Hence, $L_c$ is a scaling factor of $M_k^m$. Since $\Delta_k^m$ in (5) is obtained from these path metrics, it also contains the scaling factor $L_c$. Moreover, as we will see in the next section, the bit reliability values $L(\hat{u}_k)$ will still keep that factor after updating. Finally, the subtraction in (6) will create the extrinsic information $L_E(\hat{u}_k)$ with the same factor. It means that as the iterative decoding goes on, the branch metrics which are composed of the second and the third terms in (2) will hold the

[1] In our simulation, we assume an additive white Gaussian noise (AWGN) channel with BPSK modulation and coherent demodulation. The amplitude of the modulated symbol is normalized to 1.

factor $L_c$ in all decoding stages. Thus, we set $L_c$ equal to 1 in our simulation.

## 3 UPDATING RULE

The updating procedure is to produce the bit reliability $L(\hat{u}_k)$ from the path reliability $\Delta_k^m$. We are going to use $L_k^m$ to denote the intermediate reliability value of $u_k^m$. The initial value of $L_k^m$ is $\Delta_k^m$ and $L_k^m$ is kept being modified according to a certain updating rule while decoding. Finally, there is only one survivor sequence $\hat{u}_k$ and

$$L(\hat{u}_k) = \hat{u}_k L_k^m . \qquad (8)$$

There are two updating rules for SOVA. One was proposed by G. Battail in [4] and the other by Hagenauer in [1]. To our knowledge, only HR-SOVA was applied to decode Turbo Codes.

The Battail's rule (BR) could be explained as follows. If at time k, the jth (j<k) bit in the survivor sequence $u_j^s$ is not the same as the corresponding bit in the concurrent sequence $u_j^c$, the new log-likelihood value $L_j^s$ should be the minimum value between itself and $\Delta_k^m$,

$$u_j^s \neq u_j^c \Rightarrow L_j^s = \min(L_j^s, \Delta_k^m) . \qquad (9)$$

Otherwise, the new log-likelihood value $L_j^s$ should be the minimum value between itself and the sum of $\Delta_k^m$ and $L_j^c$,

$$u_j^s = u_j^c \Rightarrow L_j^s = \min(L_j^s, \Delta_k^m + L_j^c) . \qquad (10)$$

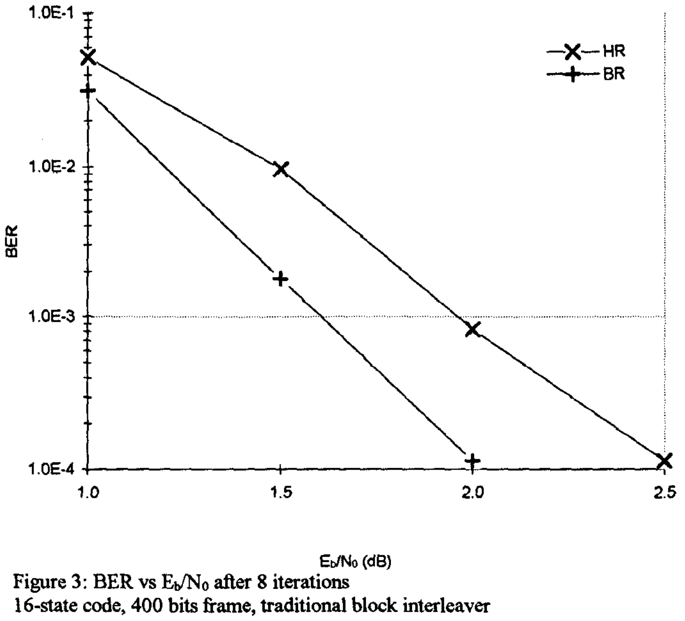

Figure 3: BER vs $E_b/N_0$ after 8 iterations
16-state code, 400 bits frame, traditional block interleaver

In comparison with BR, the Hagenauer's (HR) is simpler by only updating when $u_j^s \neq u_j^c$ according to (9).[2] Hence, it is no need to keep updating $L_j^c$, which is the key idea leading to the low complexity SOVA implementations in [8] and [9].

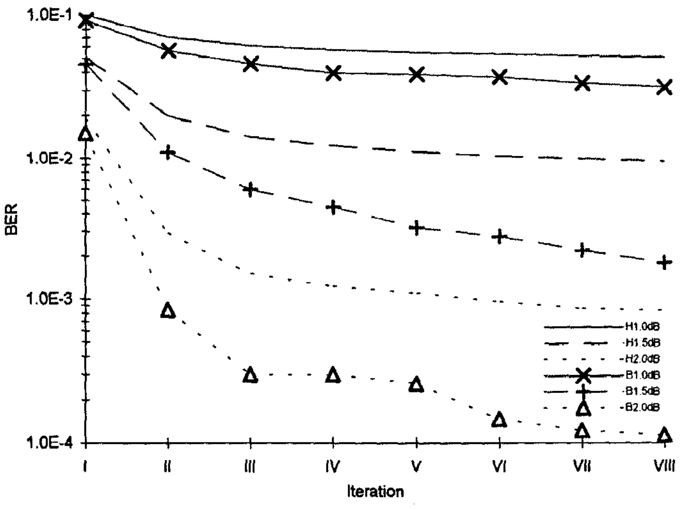

Figure 4: BER vs Iteration
16-state code, 400 bits frame, traditional block interleaver
H1.0dB, H1.5dB, H2.0dB: HR
B1.0dB, B1.5dB, B2.0dB: BR

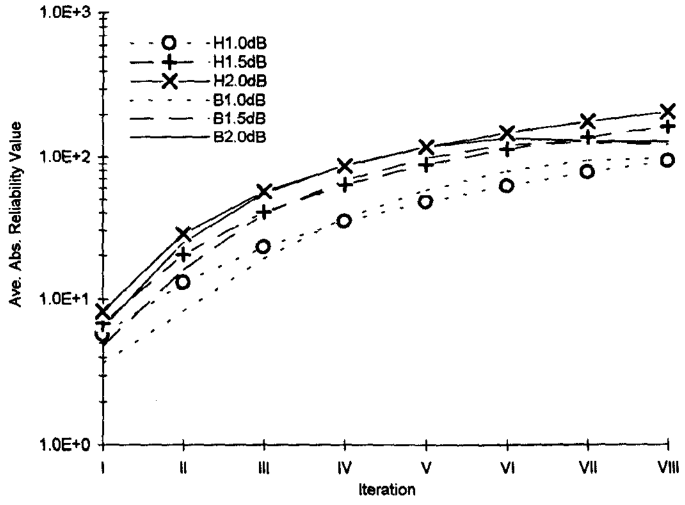

Figure 5: average absolute reliability value from iteration I to V
16-state code, 400 bits frame, traditional block interleaver
H1.0dB, H1.5dB, H2.0dB: HR
B1.0dB, B1.5dB, B2.0dB: BR

In serial concatenated coding systems, the difference between the performance of BR-SOVA and HR-SOVA was reported to be negligible [8]. However, in Turbo Codes, our simulation in Figure 3 shows that the performance of BR-SOVA is 0.5dB better than that of HR-SOVA at BER=$10^{-4}$ with a 400-bit frame and the traditional block interleaver after 8

[2] From (8), (9) and (10), it is clear that $L(\hat{u}_k)$ will have the same scaling factor $L_c$ as $\Delta_k^m$.

iterations. Moreover, Figure 4 shows that the performance of BR-SOVA is better than HR-SOVA at every iteration and that most of the gain comes in the first few iterations.

In Figure 5 and Figure 6, the average absolute reliability value $\overline{|L(\hat{u}_k)|}$ and the average absolute extrinsic information $\overline{|L_E(\hat{u}_k)|}$ after every decoding iteration are given. It is worthwhile to point out that

i) BR-SOVA gives small values for both $\overline{|L(\hat{u}_k)|}$ and $\overline{|L_E(\hat{u}_k)|}$ in the first 3 iterations, indicating that HR overestimates the reliability values, by not updating according to (10).

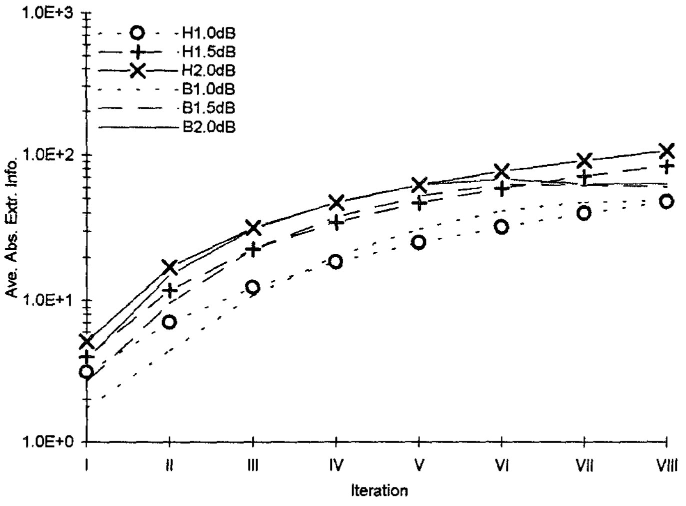

Figure 6: average absolute extrinsic information from iteration I to V
16-state code, 400 bits frame, traditional block interleaver
H1.0dB, H1.5dB, H2.0dB: HR
B1.0dB, B1.5dB, B2.0dB: BR

ii) After the first 3 iterations, $\overline{|L_E(\hat{u}_k)|}$ are more than 10 times as large as the average |y|. Hence, the path metric $M_k^m$ in (2) will rely mostly on the a priori values $L(u_k)$. Consequently, as the iteration continues, the received sequence y plays a lesser crucial role in each decoding step.

## 4 EFFECT OF RELIABILITY THRESHOLD

Using the results and analysis in the last session, we can expect better performance by keeping $\overline{|L(\hat{u}_k)|}$ and $\overline{|L_E(\hat{u}_k)|}$ small during the first few iterations. Motivated partly by this and partly by ease of hardware implementation, we propose to simply limit the range of $\Delta_k^m$, the value used for updating, by a threshold value $\Delta_{TH}$. Since $\Delta_k^m$ is always non-negative, we simply limit it by

$$\Delta_k^m > \Delta_{TH} \Rightarrow \Delta_k^m = \Delta_{TH}. \qquad (11)$$

Consequently, $L(\hat{u}_k)$ which is the final value of updating will be also limited.

Theoretically, because $\Delta_k^m$ is a real value, we can choose any real value to be $\Delta_{TH}$. However, considering quantization in implementation, it is convenient to select values equal to $2^h$ (h>0). If the received sequence y has been linearly quantized to Q levels, the $\Delta_k^m$ will have $2^{h-1}Q$ levels or $(h-1)\log_2 Q$-bit long. Thus, different h represents different bit-width for $\Delta_k^m$.

Selecting the threshold value is a very tricky process. If $\Delta_{TH}$ is too small, most reliability values will hit the boundary and the algorithm is unable to distinguish the relative reliability of various bits. However, if $\Delta_{TH}$ is too large, most reliability values will not hit $\Delta_{TH}$ in the first few iterations and it will have no effect in compensating for the overestimation in HR-SOVA.

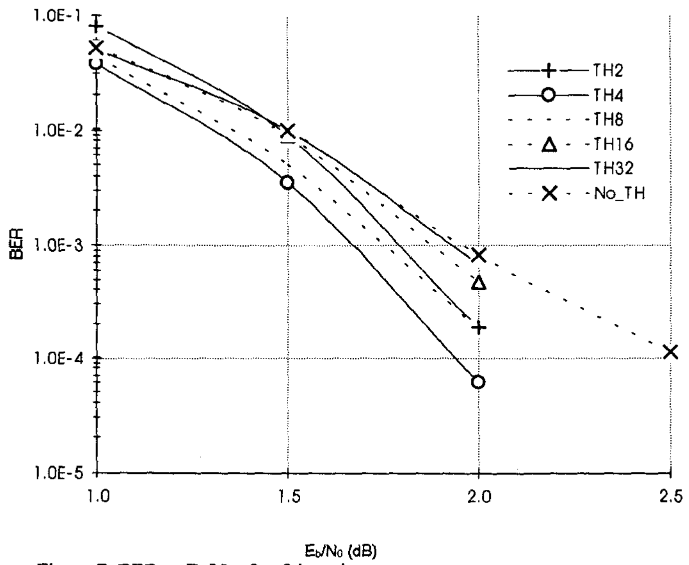

Figure 7: BER vs $E_b/N_0$ after 8 iterations
16-state code, 400 bits frame, traditional block interleaver
No_TH: HR reference
TH2, TH4, TH8, TH16 and TH32: $\Delta_{TH}$ = 2.0, 4.0, 8.0, 16.0 and 32.0

Figure 7 shows that after 8 iterations, the result with $\Delta_{TH}$ = 4.0 yields the best performance. The $\Delta_{TH}$=4.0 curve reaches the lowest BER, achieving a gain 0.5 dB against the one without threshold at BER of $10^{-4}$. We also observe that the $\Delta_{TH}$ = 4.0 is optimal for all $E_b/N_0$ between 1.0 dB and 2.0 dB, inclusively.

## 5 INTERLEAVER DESIGN FOR CANCELING TAIL EFFECT

When decoding only one frame of data at a time, the bits near the end of a frame will have less accurate estimates of their reliability values. We refer to this effect as the tail effect and it limits the overall performance achieved by Turbo Codes, especially when a traditional block interleaver (TBI) is used. For the sake of simplicity and symmetry, we only consider the square interleavers in this section.

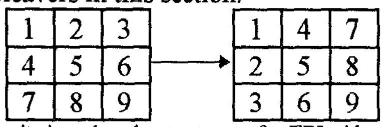

Figure 8: write-in and read-out patterns of a TBI with m=3

$$\begin{array}{|c|c|c|}\hline 1 & 2 & 3 \\ \hline 4 & 5 & 6 \\ \hline 7 & 8 & 9 \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|c|}\hline 7 & 4 & 1 \\ \hline 8 & 5 & 2 \\ \hline 9 & 6 & 3 \\ \hline \end{array}$$

Figure 9-a: write-in and read-out patterns of the new interleaver with m=3

$$\begin{array}{|c|c|c|}\hline 1 & 2 & 3 \\ \hline 4 & 5 & 6 \\ \hline 7 & 8 & 9 \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|c|}\hline 3 & 6 & 9 \\ \hline 2 & 5 & 8 \\ \hline 1 & 4 & 7 \\ \hline \end{array}$$

Figure 9-b: write-in and read-out patterns of the new deinterleaver with m=3

A traditional (square) block interleaver (TBI) formats the data in a square array of m rows by m columns. The data are written in row-wise and read out in column-wise. Figure 8 gives the write-in and read-out patterns of a traditional block interleaver for m=3. In this case, the deinterleaver is identical to the interleaver.

When using a TBI, the tail bits of the input frame remain around the end of the output frame after shuffling. It means that $L(u_k)$ associated with those bits will not be updated sufficiently according to any updating rule of SOVA. Therefore, those $L(u_k)$ are likely to be much larger than what they should be. Moreover, their inaccuracy is also very likely to affect the reliability estimates of their neighboring bits in successive decoding stage. Hence, the overall performance of SOVA decoding will be degraded. Generally, it is expected that the tail effect is more serious in a short frame than in a long frame.In order to make the $L(u_k)$ for bits at the end of a frame more reliable, we propose a new square block interleaver. Figure 9-a and 9-b give the write-in and the read-out patterns of our proposed interleaver and those of its deinterleaver.

From Figure 9, it is easy to see that the read-out pattern of the interleaver is the write-in pattern rotated clockwise by $90^{\circ}$. In contrast to this, the deinterleaver reverses the operation by rotating the data anti-clockwise by $90^{\circ}$. Due to this structure, we call the interleaver, rotated block interleaver (RBI). Comparing with the traditional block interleaver, there is no significant difference other than that the tail bits will not remain at the frame end after shuffling. An evidence of this is that both of them can not break the error pattern mentioned by Divsalar [10].

The simulation results in Figure 10 show that when HR-SOVA is performed, there is no significant difference between using TBI and RBI. When BR-SOVA is used, the difference between using TBI or RBI increases corresponding to the increase in $E_b/N_0$. When the threshold limit is included in HR-SOVA, we find that the RBI provides no significant advantage when $E_b/N_0$ is less than 2 dB. However, at 2.5 dB, using RBI reduces the BER by more than an order of magnitude.

The results can be explained in this way. At low $E_b/N_0$ where error events are evenly distributed in a data frame and the reliability values are not very accurate, the degradation caused by the tail effect is not very serious. However, at comparatively high $E_b/N_0$, with the fairly precise reliability estimates, the error events brought by the tail effect will dominate the overall performance of the coding scheme.

In [11], a rate 1/3 4-state Turbo Code with a 100-bit frame and generators (5, 7) was used to evaluate the author's MAP-based algorithm. Here, we also use the same code to measure the difference of the performance of their method and ours. In Figure 11, we provide the simulation results of using our modified SOVA with $\Delta_{TH} = 4.0$ and both TBI and RBI.

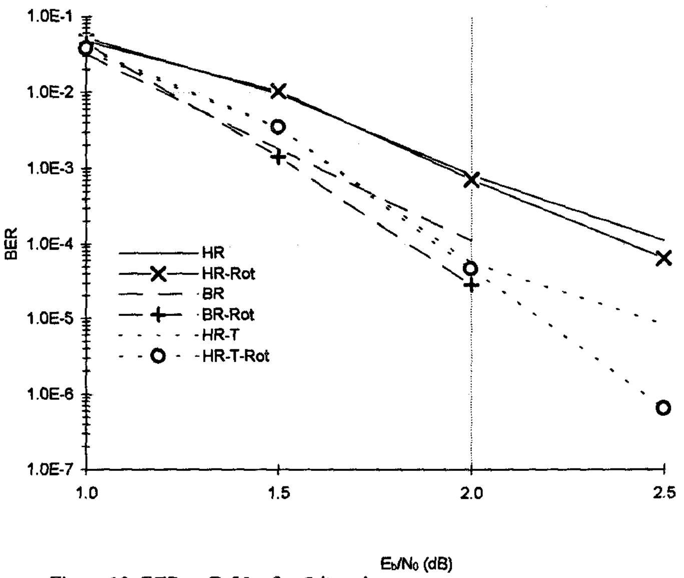

Figure 10: BER vs $E_b/N_0$ after 8 iterations
16-state code, m = 20
HR, HR-Rot: HR with block interleaver or rotated block interleaver
BR, BR-Rot: BR with block interleaver or rotated block interleaver
HR-T, HR-T-Rot: HR ($\Delta_{TH} = 4$) with and block interleaver or rotated block interleaver

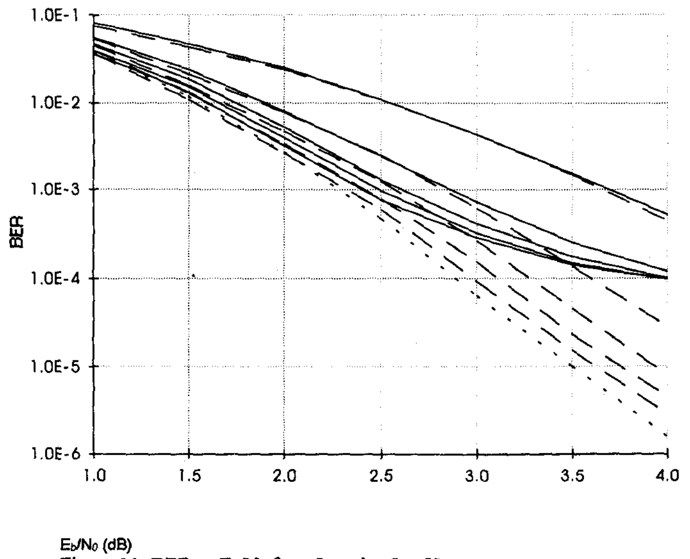

Figure 11: BER vs $E_b/N_0$ from Iteration I to V
4-state code, 100 bits frame (m = 10)
——— results with TBI
- - - - - results with RBI
········ result with s-random interleaver after 5 iterations.

When TBI is used and $E_b/N_0$ is less than 2dB, we find that the performance achieved by the modified SOVA is equivalent to that reported in [11]. Unfortunately, when $E_b/N_0$ becomes larger, the performance of our modified SOVA becomes flat. However, when RBI is used, the modified SOVA gives comparable results at all $E_b/N_0$ ranging from 1.0 dB to 4.0 dB.

In Figure 11, for reference, we also present a simulation result with a s-random interleaver invented by D. Divsalar [10]. At BER of $10^{-5}$, there is only a 0.1 dB gain over the result with RBI.

## 6 CONCLUSION

It is known that the original HR-SOVA in [1] for Turbo Code decoding gives too optimistic soft output values. It is also known that the updating rule proposed by Battail can give small soft output values as it keeps updating the reliability values associated with the concurrent path. We compare the performance of these two SOVA schemes for Turbo Code decoding and find that the performance of BR-SOVA is better than HR-SOVA. Also, we find that BR-SOVA gives smaller average value of the reliability information and the extrinsic information than HR-SOVA at the first several iterations where most of bit errors are corrected.

Motivated by this and also by ease of hardware implementation, we propose to limit the reliability value in a small range to compensate for the defect brought by overestimating the reliability. From simulation, this method can improve the performance of HR-SOVA by 0.5 dB at BER of $10^{-4}$, when a rate 1/3 16-state Turbo Code with a 400-bit frame and the traditional block interleaver is used.

In order to combat the tail effect in Turbo Codes, which is critical in the relatively high $E_b/N_0$ situation, we propose to use a rotated block interleaver. Our simulation shows that a recently reported result achieved by a MAP algorithm with a random interleaver can be attained by the SOVA-based decoding with both of our modifications.

## ACKNOWLEDGMENT

The authors would like to thank Dr. Justin C-I Chuang for his consistent support for our research work.

## REFERENCES

[1] J. Hagenauer and P. Robertson, "Iterative ("TURBO") Decoding of Systematic Convolutional Codes with the MAP and SOVA Algorithms," *ITG-Fachberichte*, v. 130, pp.21-29, 1995

[2] C. Berrou, P. Combelles, P. Pénard, B. Talibart, "An IC for Turbo-Codes Encoding and Decoding," *Proc. of ISSCC*, pp.90-1, Feb 1995

[3] L. Papke and P. Robertson, "Improved Decoding with the SOVA in a Parallel Concatenated (Turbo-code) Scheme," *Proc. of ICC*, July 1996

[4] G. Battail, "Pondération des symboles décodés par l'agorithem de Viterbi (in French)," *Ann. Télécommun.*, Fr., 42, N 1-2, pp. 31-38, Jan 1987

[5] S. Benedetto and G. Montorsi, "Design of parallel concatenated convolutional codes," *IEEE Trans. Commun.*, vol. 44, iss. 5, pp. 591-600, June 1996

[6] J. Hagenauer and P. Hoeher, "A Viterbi Algorithm with Soft-Decision Outputs and its Application," *Proc. of the IEEE GLOBECOM*, pp. 47.1.1-47.1.7, Nov 1989

[7] J. Hagenauer, "Source-Controlled Channel Decoding," *IEEE Trans. Commun.*, vol. 43, pp. 2449-2457, Sept 1995

[8] C. Berrou, P. Adde, E. Angui and S. Faudeil, "A Low Complexity Soft-Output Viterbi Decoder Architecture," *Proc. of ICC*, pp. 737-740, 1993

[9] O. Joeressen, M. Vaupel and H. Meyr, "High-Speed VLSI Architectures for Soft-Output Viterbi Decoding," *Proc. of ICASAP*, pp. 373-84, Aug 1992

[10] D. Divsalar, "Turbo Codes for PCS Applications," *Proc. of ICC*, pp. 54-9, 1995

[11] S. Benedetto and G. Montorsi, "Unveiling Turbo Codes : Some Results on Parallel Concatenated Coding Schemes," *IEEE Trans. Inform. Theory*, vol. 42, no. 2, Mar 1996