Low-Complexity MAP Decoding for Turbo Codes

Duanyi Wang and Hisashi Kobayashi

Department of Electrical Engineering, Princeton University
Princeton, NJ 08544, U. S. A.
Tel: (609) 258-4643 Fax: (609) 258-2158
duanyi@ee.princeton.edu hisashi@ee.princeton.edu

Abstract - Two simplified MAP algorithms for iterative decoding of turbo codes are presented. By using a parameter "decoding depth", our algorithms do not require computation of the *a posteriori* probability of each decoded information bit throughout the entire trellis, thus reduce the computational complexity and decoding delay considerably. One of the algorithms can achieve the performance very close to the conventional MAP algorithm; the other performs better than SOVA, while retaining a comparable process cost. Their advantages over the MAP and SOVA algorithms are demonstrated in both theoretical analyses and computer simulation assuming a Rayleigh fading channel.

I. Introduction

Maximum *a posteriori* (MAP) decoding, based on the BCJR algorithm [1], has recently seen a resurgence of interest since its application to the powerful turbo codes [2]. The MAP algorithm can not only make optimum symbol-by-symbol decisions, but also provide soft reliability information that is necessary in iterative decoding, one of the key strategies of turbo codes. Turbo codes will be used in the 3rd generation wideband DS-CDMA systems called IMT-2000. There are increasing demands for the design of practical MAP decoders in order to use turbo codes in a wide range of applications, such as mobile communication systems and digital recording devices. However, the original MAP algorithm suffers from serious drawbacks in its implementation. Most notably, the algorithm requires that an entire sequence must be received before decoding can commence, using both forward and backward recursions. Thus it incurs not only a considerable decoding delay, but also a substantial amount of storage. Although some sub-optimal versions, such as SOVA, Max-Log-MAP and Log-MAP, have been proposed to reduce these costs [3,4], a further study on low-complexity MAP-based algorithm with minimal performance penalty is called for. In view of this, we have been investigating efficient methods to produce soft output information, based on prior results on the reliability measure introduced for general concatenated systems [5]. The present paper proposes two simplified algorithms, denoted S-MAP and Max-Log-S-MAP respectively.

The principles of our simplified algorithms are as follows: The principle of S-MAP is to apply a generalized soft output generation method to turbo decoding. By using a parameter "decoding depth", our algorithm does not require computation of the *a posteriori* probability (APP) of each information bit throughout the entire trellis, thus reduces the computational complexity and decoding delay considerably. The principle of Max-Log-S-MAP, which is a further simplified variation of S-MAP, is to perform the S-MAP computation in the logarithmically transformed domain, and avoid the operations that involve exponentiation and multiplication. We demonstrate the advantages of our simplified algorithms over the conventional MAP and SOVA by comparing their computational complexity, decoder storage and decoding delay, and by conducting computer simulations assuming a Rayleigh fading environment. The BER performances of these algorithms show that S-MAP can be very close to MAP and much better than SOVA; Max-Log-S-MAP is inferior to S-MAP but better than SOVA. Its performance approaches to that of MAP, as SNR of the receiver input increases.

II. System Model and Notation

Consider the transmission system model of Figure 1. The turbo encoder is constructed by parallel concatenation of two identical rate ½, constraint length K ($K = v+1$, v is the number of memory elements) recursive systematic convolutional constituent encoders, RSC1 and RSC2, with a turbo internal interleaver in-between, as shown in Figure 2. For each input information sequence block $\mathbf{d} = \{d_1, d_2, \cdots, d_N\}$ of length N, whose elements take on values of 0 or 1 equiprobably, RSC1 operates directly on it and produces the first parity sequence $\boldsymbol{Y}_1 = \{Y_{11}, Y_{12}, \ldots, Y_{1N}\}$; RSC2 operates on the interleaved version of $\mathbf{d}$ and produces the second parity sequence $\boldsymbol{Y}_2 = \{Y_{21}, Y_{22}, \ldots, Y_{2N}\}$. So the overall turbo-coded sequence $\mathbf{C} = \{C_1, C_2, \cdots, C_N\}$ is the sum of the three parts, i.e., $\mathbf{C} = (\boldsymbol{X}, \boldsymbol{Y}_1, \boldsymbol{Y}_2)$. The resultant coding rate of this turbo code is 1/3. Higher rates can be obtained by proper puncturing of $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$. We assume that the encoded sequence is transmitted over a fading channel, during which the channel interleaving and a QPSK modulation are employed. At the receiver side, the sequence applied to the turbo decoder is denoted $\mathbf{R} = \{R_1, R_2, \cdots, R_N\}$, where $\mathbf{R} = (\boldsymbol{x}, \boldsymbol{y}_1, \boldsymbol{y}_2)$ and $R_k = (x_k, y_{1k}, y_{2k})$ is the noise

0-7803-5718-3/00/$10.00 ©2000 IEEE.

corrupted version of C_k at time k (assume sufficient channel interleaving)

$$x_k = a_k(2X_k - 1) + p_k$$
$$y_{ik} = a_k(2Y_{ik} - 1) + q_k \qquad i = 1 \text{ or } 2 \quad (1)$$

where a_k is the instantaneous fading amplitude with a Rayleigh pdf, $p_A(a_k) = 2a_k e^{-a_k^2}$ for $a_k > 0.$, and p_k and q_k are two independent noise variables, both with the zero mean and variance σ^2.

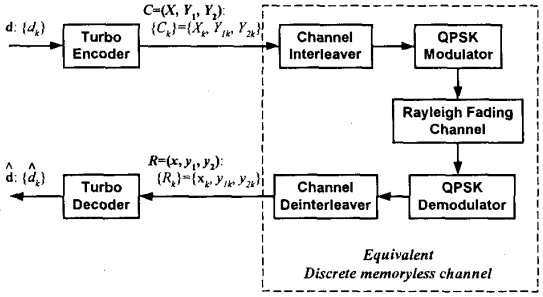

Figure 1 Transmission model

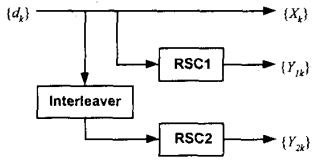

Figure 2 Turbo encoder

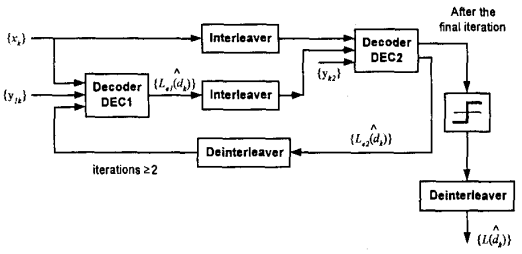

Figure 3 Turbo decoder

The global turbo decoder structure, shown in Figure 3, includes two constituent decoders, DEC1 and DEC2, implementing *a posteriori* probability, and interleavers/ deinterleavers. There are three types of soft inputs to each constituent decoder (DEC1 or DEC2): $\boldsymbol{x}$, $\boldsymbol{y}_i$ (i = 1 or 2), and the *a priori* information, which is the extrinsic information provided by the other constituent decoder from the previous step of decoding process. The soft output generated by each constituent decoder at time k also consists of three components: a weighted version of x_k, the *a priori* value (i.e., the previous extrinsic information) and a newly generated extrinsic information, which is then provided to the other constituent decoder as *a priori* information for the next step of decoding. Such iterative steps will continue with ever-updating extrinsic information to be exchanged between two decoders until a reliable hard decision can be made.

Let the state of the RSC encoder at time k be s_k, which takes on an integer value between 0 and M-1 ($M = 2^v$). The kth information bit d_k drives the encoder to change its state from $s_{k-1} = m'$ to $s_k = m$. When the MAP algorithm is adopted for every constituent decoder, the soft output for each decoded bit d_k is determined from the log-likelihood ratio (LLR) as follows:

$$\Lambda(d_k) = \ln \frac{P_r\{d_k = 1 | \mathbf{R}\}}{P_r\{d_k = 0 | \mathbf{R}\}} = \ln \frac{\sum_m \sum_{m'} \gamma_k^1(R_k, m', m)\alpha_{k-1}(m')\beta_k(m)}{\sum_m \sum_{m'} \gamma_k^0(R_k, m', m)\alpha_{k-1}(m')\beta_k(m)} \quad (2)$$

where $\alpha_k(m)$ and $\beta_k(m)$ are recursively obtainable from the "branch transition probability" γ_k^i as follows:

$$\alpha_k(m) = \frac{\sum_{m'} \sum_i \gamma_k^i(R_k, m', m)\alpha_{k-1}(m')}{\sum_m \sum_{m'} \sum_i \gamma_k^i(R_k, m', m)\alpha_{k-1}(m')} \quad (3)$$

$$\beta_k(m) = \frac{\sum_{m'} \sum_i \gamma_k^i(R_k, m, m')\beta_{k+1}(m')}{\sum_m \sum_{m'} \sum_i \gamma_k^i(R_k, m', m)\alpha_k(m')}, \quad (4)$$

and

$$\gamma_k^i(R_k, m', m) = \mathrm{P}_r\{d_k = i, s_k = m, R_k | s_{k-1} = m'\} \quad (5)$$

can be computed based on the transition probabilities of the channel and RSC encoders..

For the turbo decoder in Figure 3, $\Lambda(d_k)$ can be decomposed into the following three terms:

$$\Lambda(d_k) = L_c x_k + L_a(\hat{d}_k) + L_e(\hat{d}_k) = L_c x_k + L_{e,in}(\hat{d}_k) + L_{e,out}(\hat{d}_k) \quad (6)$$

where L_c is the channel reliability values. For a Rayleigh fading channel, $L_c = \frac{2a_k}{\sigma^2}$. The term $L_a(\hat{d}_k)$ is the *a priori* information generated by the previous constituent decoder. It is usually set 0 at the beginning of the iterative decoding process. And $L_e(\hat{d}_k)$ is the extrinsic information.

III. Principles of Simplified MAP Algorithm

(1) Simplified MAP Algorithm (S-MAP)

For the model of Figure 1, let $\mathbf{D}_N$ be the set of all possible information sequences of length N, and assume that all the information sequences are equally likely. Let $\mathbf{D}_{N,k}^i$ be the set of all information sequences of length N

with $d_k = i$ (i = 0,1). So for any given coordinate k, $\mathbf{D}_N$ can be partitioned into $\mathbf{D}^1_{N,k}$ and $\mathbf{D}^0_{N,k}$. The APP for each decoded bit $d_k = i$ in the conventional MAP algorithm can be represented as follows:

$$P_r(d_k = i \,|\, \mathbf{R}) = \frac{P_r\{d_k = i, \mathbf{R}\}}{P_r\{\mathbf{R}\}} = \frac{\sum_{\mathbf{d}\in\mathbf{D}'_{N,k}} P_r\{\mathbf{R},\mathbf{d}\}}{\sum_{\mathbf{d}\in\mathbf{D}_N} P_r\{\mathbf{R},\mathbf{d}\}} = \frac{\sum_{\mathbf{d}\in\mathbf{D}'_{N,k}} P_r\{\mathbf{R}|\mathbf{d}\}P_r\{\mathbf{d}\}}{\sum_{\mathbf{d}\in\mathbf{D}_N} P_r\{\mathbf{R}|\mathbf{d}\}P_r\{\mathbf{d}\}} = \frac{\sum_{\mathbf{d}\in\mathbf{D}'_{N,k}} P_r\{\mathbf{R}|\mathbf{d}\}}{\sum_{\mathbf{d}\in\mathbf{D}_N} P_r\{\mathbf{R}|\mathbf{d}\}} \tag{7}$$

For a memoryless channel,

$$P_r\{\mathbf{R}|\mathbf{d}\} = \prod_{k=1}^{N} p(R_k|d_k, s_{k-1}) = \prod_{k=1}^{N} p(R_k|s_k, s_{k-1}) \,. \tag{8}$$

So we have

$$P_r(d_k = i \,|\, \mathbf{R}) = \frac{\sum_{\mathbf{d}\in\mathbf{D}'_{N,k}} \prod_{j=1}^{N} p(R_j|s_j, s_{j-1})}{\sum_{\mathbf{d}\in\mathbf{D}_N} \prod_{j=1}^{N} p(R_j|s_j, s_{j-1})} \tag{9}$$

Let $\mathbf{S}_N$ be the set of all possible paths of length k. Suppose the encoder starts from a given initial state (usually all-zero state) in the trellis, and any information sequence $\mathbf{d} \in \mathbf{D}_N$ can always generate a unique corresponding state path $\mathbf{s} \in \mathbf{S}_N$. So the terms "sequence" and "path" here may be used interchangeably. We can rewrite

$$P_r(d_k = i \,|\, \mathbf{R}) = \frac{\sum_{\mathbf{s}\in\mathbf{S}'_{N,k}} \prod_{j=1}^{N} p(R_j|s_j, s_{j-1})}{\sum_{\mathbf{s}\in\mathbf{S}_N} \prod_{j=1}^{N} p(R_j|s_j, s_{j-1})} \tag{10}$$

where $\mathbf{S}^i_{N,k}$ is the set of all state paths of length N with $d_k = i$ (i = 0, 1). Let $\mathbf{S}^m_k$ be the set of all possible paths of length k that cause the encoder to terminate at state $s_k = m$. Let

$$\lambda^m_k = \sum_{\mathbf{s}\in\mathbf{S}^m_k} \prod_{j=1}^{k} p(R_j|s_{j-1}, s_j) = \sum_{\mathbf{s}\in\mathbf{S}^m_k} \prod_{j=1}^{k} p(R_j|s_j = m, s_{j-1} = m') \tag{11}$$

By the definition of encoder state, it can be shown that

$$\lambda^m_k = p(R_k|s_k = m, s_{k-1} = m'_1)\lambda^{m'_1}_{k-1} + p(R_k|s_k = m, s_{k-1} = m'_0)\lambda^{m'_0}_{k-1} \tag{12}$$

where m'_1 and m'_0 are such two states in the trellis diagram which make transitions to a common state m when the encoder input d_k is 1 and 0, respectively.

Define the row vector $\boldsymbol{\lambda}_k$ by

$$\boldsymbol{\lambda}_k = (\lambda^0_k, \lambda^1_k, \cdots, \lambda^{M-1}_k) \,, \tag{13}$$

and an $M \times M$ matrix $\mathbf{A}_k = [a_{m,m'}]$ ($0 \le m, m' \le M$-1) by

$$a_{m,m'} = p(R_k|s_k = m, s_{k-1} = m') = K_1 e^{\{\frac{a_k}{\sigma^2}x_k + \frac{1}{\sigma^2_L}L_{e\_in}(\hat{d}_k)\}(2i-1) + \frac{a_k}{\sigma^2}y_{jk}(2Y_{jk}-1)} \,, \tag{14}$$

which K_1 is a constant, σ^2_L is the variance of $L_{e\_in}(\hat{d}_k)$, j = 1 or 2.

Note that $\mathbf{A}_k$ has $2M$ nonzero entries only at positions (m, m'_i), $0 \le m, m'_i \le M$-1, i = 0, 1. Then it is not difficult to see that the following recursion holds:

$$\boldsymbol{\lambda}_{k+1}{}^T = \mathbf{A}_k \boldsymbol{\lambda}_k{}^T \,. \tag{15}$$

We further decompose the vector

$$\boldsymbol{\lambda}_k = \boldsymbol{\alpha}_k + \boldsymbol{\beta}_k \tag{16}$$

with

$$\boldsymbol{\alpha}_k = (\alpha^0_k, \alpha^1_k, \cdots, \alpha^{M-1}_k) \tag{17}$$

where

$$\alpha^m_k = p(R_k|s_k = m, s_{k-1} = m'_1)\lambda^{m'_1}_{k-1} \tag{18}$$

and

$$\boldsymbol{\beta}_k = (\beta^0_k, \beta^1_k, \cdots, \beta^{M-1}_k) \tag{19}$$

where

$$\beta^m_k = p(R_k|s_k = m, s_{k-1} = m'_0)\lambda^{m'_0}_{k-1} \tag{20}$$

Using above matrix notations (16), (17) and (19), and let $\mathbf{e} = (1, 1, \ldots, 1)$, we then obtain the following form from (10):

$$P_r\{d_k = 1|\mathbf{R}\} = \frac{(\boldsymbol{\alpha}_k \mathbf{A}_k{}^T \mathbf{A}_{k+1}{}^T \cdots \mathbf{A}_N{}^T)\mathbf{e}^T}{\boldsymbol{\lambda}_N \mathbf{e}^T} \tag{21}$$

$$P_r\{d_k = 0|\mathbf{R}\} = \frac{(\boldsymbol{\beta}_k \mathbf{A}_k{}^T \mathbf{A}_{k+1}{}^T \cdots \mathbf{A}_N{}^T)\mathbf{e}^T}{\boldsymbol{\lambda}_N \mathbf{e}^T} \,. \tag{22}$$

The above estimations of (21) and (22) require the reception of the complete sequence before any soft outputs are available. To reduce the memory space and decoding delay, we introduce a "decoding depth δ", like the situation in the Viterbi algorithm (VA) [5,6] , and assume that if the δ is large enough, one can reasonably expect that the information at time $k+\delta$ is enough to decide the APP for bit d_k, instead of computing $\boldsymbol{\lambda}_N$, i.e.,

$$P_r\{d_k = 1|\mathbf{R}\} \approx \frac{(\boldsymbol{\alpha}_k \mathbf{A}_k \mathbf{A}_{k+1} \cdots \mathbf{A}_{k+\delta-1})\mathbf{e}^T}{\boldsymbol{\lambda}_{k+\delta}\,\mathbf{e}^T} \tag{23}$$

$$P_r\{d_k = 0|\mathbf{R}\} \approx \frac{(\boldsymbol{\beta}_k \mathbf{A}_k \mathbf{A}_{k+1} \cdots \mathbf{A}_{k+\delta-1})\mathbf{e}^T}{\boldsymbol{\lambda}_{k+\delta}\,\mathbf{e}^T} \tag{24}$$

and the LLR for d_k is

$$\Lambda(d_k) = \ln \frac{P_r\{d_k = 1|\mathbf{R}\}}{P_r\{d_k = 0|\mathbf{R}\}} = \frac{P_r\{d_k = 1|\mathbf{R}\}}{1 - P_r\{d_k = 1|\mathbf{R}\}} \tag{25}$$

The simplified MAP algorithm can be outlined as follows:

S-MAP Algorithm

1) Initialize $\boldsymbol{\lambda}_0 = (1, 0, \cdots, 0)$;
2) For each observation R_k, compute $\mathbf{A}_k$ by using (14), $\boldsymbol{\lambda}_{k+1}$ by (15) and $\boldsymbol{\alpha}_{k+1}$ by $\boldsymbol{\alpha}_{k+1}{}^T = \mathbf{A}_k \boldsymbol{\alpha}_k{}^T$;
3) At time $k+\delta-1$, compute LLR by

$$P_r\{d_k = 1 | \mathbf{R}\} \approx \frac{(\boldsymbol{\alpha}_k \mathbf{A}_k{}^T \mathbf{A}_{k+1}{}^T \cdots \mathbf{A}_{k+\delta-1}{}^T)\mathbf{e}^T}{\boldsymbol{\lambda}_{k+\delta}\, \mathbf{e}^T}$$

$$\Lambda(d_k) = \ln \frac{\mathrm{P}_r\{d_k = 1 | \mathbf{R}\}}{1 - \mathrm{P}_r\{d_k = 1 | \mathbf{R}\}}$$

4) Compute the extrinsic information by

$$L_{e,out}(\hat{d}_k) = \Lambda(d_k) - \frac{2a_k}{\sigma^2} x_k - L_{e,in}(\hat{d}_k) \qquad 6)$$

(2) Logarithmic Simplified MAP Algorithm (Max-Log-S-MAP)

From the view point of implementation, it is highly desirable that the decoder should avoid complicated arithmetic operations, such as multiplication and exponentiation. Some approximations of the conventional MAP algorithm have been derived, which work exclusively in the logarithmic domain, and so values and operations (addition and max-function) are easier to handle. Following Robertson *et al* [4], we develop the logarithmic form of our simplified MAP algorithm described in Section 3(1).

Define the "logarithm"s of λ_k, α_k and β_k as follows

$$\bar{\boldsymbol{\lambda}}_k = (\bar{\lambda}_k^0, \bar{\lambda}_k^1, \cdots, \bar{\lambda}_k^{M-1}), \qquad (27)$$

$$\bar{\boldsymbol{\alpha}}_k = (\bar{\alpha}_k^0, \bar{\alpha}_k^1, \cdots, \bar{\alpha}_k^{M-1}), \quad \bar{\boldsymbol{\beta}}_k = (\bar{\beta}_k^0, \bar{\beta}_k^1, \cdots, \bar{\beta}_k^{M-1}), \qquad (28)$$

where

$$\bar{\lambda}_k^m = \ln \lambda_k^m \qquad (29)$$

$$\bar{\alpha}_k^m = \ln \alpha_k^m \quad \text{and} \quad \bar{\beta}_k^m = \ln \beta_k^m \ . \qquad (30)$$

By using the following approximation [4]:

$$\ln\left(e^{\theta_0} + e^{\theta_1} + \ldots + e^{\theta_{M-1}}\right) \approx \max_{i \in \{0,1,\mathrm{L},M-1\}} \theta_i \ , \qquad (31)$$

where $\max_{i \in \{0,\cdots,M-1\}} \theta_i$ can be calculated by successively using M-1 maximum functions over only two values, we have

$$\begin{aligned} \bar{\lambda}_k^m &= \ln\left[e^{\ln p(R_k | s_k = m, s_{k-1} = m_1^{'})\bar{\lambda}_{k-1}^{m_1^{'}}} + e^{\ln p(R_k | s_k = m, s_{k-1} = m_0^{'})\bar{\lambda}_{k-1}^{m_0^{'}}}\right] \\ &\approx \max\left[\ln p(R_k | s_k = m, s_{k-1} = m_1^{'}) + \bar{\lambda}_{k-1}^{m_1^{'}}, \right. \\ &\qquad \left. \ln p(R_k | s_k = m, s_{k-1} = m_0^{'}) + \bar{\lambda}_{k-1}^{m_0^{'}}\right] \\ &= \max\left[\bar{\alpha}_k^m, \bar{\beta}_k^m\right] \ . \end{aligned} \qquad (32)$$

Proceeding in a manner similar to (1), we define an $M \times M$ matrix $\mathbf{B}_k$ with $2M$ nonzero entries on positions $(m, m_i^{'})$, $0 \le m, m_i^{'} \le M$-1, i = 0, 1, and the $(m, m_i^{'})$ -th entry $b(m, m_i^{'})$ of $\mathbf{B}_k$ is

$$\begin{aligned} b_{m,m_i^{'}} &= \ln p(R_k | s_k = m, s_{k-1} = m_i^{'}) \\ &= K_2\left\{\left[\frac{a_k}{\sigma^2} x_k + \frac{1}{\sigma_L^2} L_{e\_in}(\hat{d}_k)\right](2i-1) + \frac{a_k}{\sigma^2} y_{jk}(2Y_{jk} - 1)\right\} \end{aligned} \ , \qquad (33)$$

where K_2 is a constant, j = 1 or 2. We also define

$$\bar{\boldsymbol{\lambda}}_{k+1}{}^T = \mathbf{B}_k * \bar{\boldsymbol{\lambda}}_k{}^T \ , \qquad (34)$$

where the operation $*$ means:

$$\lambda_{k+1}^m = \max_{j \in \{0,1,\ldots,M-1\}} \left[b_{m,j} \lambda_k^j\right] \ , \quad m = 0, 1, \ldots, M\text{-}1 \ . \qquad (35)$$

So $\mathbf{B}_k * \bar{\boldsymbol{\lambda}}_k$ here consists of $2M$ additions and M comparisons. Similarly, we have

$$\bar{\boldsymbol{\alpha}}_{k+1}{}^T = \mathbf{B}_k * \bar{\boldsymbol{\alpha}}_k{}^T \quad \text{and} \quad \bar{\boldsymbol{\beta}}_{k+1}{}^T = \mathbf{B}_k * \bar{\boldsymbol{\beta}}_k{}^T \ . \qquad (36)$$

Therefore, the LLR for d_k can be written as

$$\begin{aligned} \Lambda(d_k) &= \ln \frac{\mathrm{P}_r\{d_k = 1 | \mathbf{R}\}}{\mathrm{P}_r\{d_k = 0 | \mathbf{R}\}} = \ln \frac{\boldsymbol{\alpha}_{k+\delta} \mathbf{e}^T}{\boldsymbol{\beta}_{k+\delta} \mathbf{e}^T} \\ &= \ln \frac{\alpha_{k+\delta}^0 + \alpha_{k+\delta}^0 + \cdots + \alpha_{k+\delta}^{M-1}}{\beta_{k+\delta}^0 + \beta_{k+\delta}^0 + \cdots + \beta_{k+\delta}^{M-1}} \\ &= \ln\left(e^{\bar{\alpha}_{k+\delta}^0} + e^{+\bar{\alpha}_{k+\delta}^1} + \cdots + e^{\bar{\alpha}_{k+\delta}^{M-1}}\right) \\ &\quad - \ln\left(e^{\bar{\beta}_{k+\delta}^0} + e^{+\bar{\beta}_{k+\delta}^1} + \cdots + e^{\bar{\beta}_{k+\delta}^{M-1}}\right) \\ &\approx \max_{m \in \{0,1,\cdots,M-1\}} \left(\bar{\alpha}_{k+\delta}^m\right) - \max_{m \in \{0,1,\cdots,M-1\}} \left(\bar{\beta}_{k+\delta}^m\right) \end{aligned} \qquad (37)$$

where $\bar{\boldsymbol{\alpha}}_{k+\delta}$ and $\bar{\boldsymbol{\beta}}_{k+\delta}$ can be obtained from the following calculations

$$\bar{\boldsymbol{\alpha}}_{k+\delta} = \bar{\boldsymbol{\alpha}}_k * \mathbf{B}_k{}^T * \mathbf{B}_{k+1}{}^T * \cdots * \mathbf{B}_{k+\delta-1}{}^T \qquad (38)$$

$$\bar{\boldsymbol{\beta}}_{k+\delta} = \bar{\boldsymbol{\beta}}_k * \mathbf{B}_k{}^T * \mathbf{B}_{k+1}{}^T * \cdots * \mathbf{B}_{k+\delta-1} T \ . \qquad (39)$$

The logarithmic version of our simplified MAP algorithm can be outlined as follows:

Max-Log-S-MAP Algorithm

1) Initialize $\bar{\boldsymbol{\lambda}}_0 = (1, 0, \cdots, 0)$;
2) For each observation R_k, compute $\mathbf{B}_k$ by using (33), $\bar{\boldsymbol{\lambda}}_{k+1}$ by (34) and (35), $\bar{\boldsymbol{\alpha}}_{k+1}$ and $\bar{\boldsymbol{\beta}}_{k+1}$ by (36).
3) At time $k+\delta-1$, compute LLR $\Lambda(d_k)$ by (37).
4) Compute the extrinsic information by (26).

IV Algorithm Comparison and Performance Simulation

A comprehensive comparison between our simplified algorithms and the conventional MAP and SOVA is given in Table 1. In computational complexity, S-MAP is still much higher compared with Max-Log-S-MAP

and SOVA because of its operations of multiplication and exponentiation. Max-Log-S-MAP costs more than twice as much as SOVA in terms of addition; In terms of decoder storage S-MAP, Max-Log-S-MAP and SOVA require much less than MAP, and Max-Log-MAP and SOVA require twice as much as S-MAP; In decoding delay, S-MAP, Max-Log-S-MAP and SOVA all depend on the decoding depth δ, which is feasible for some implementation of turbo codes with short constraint length. While MAP is related to the length of the received sequence. This is obviously prohibitive for many practical applications.

The BER (bit error rate) performances for the four algorithms are estimated by computer simulation under a fully interleaved Rayleigh fading channel environment. An 8-state ($K = 4$) rate-1/3 turbo code $(1, 17/15, 17/15)_{oct}$ is adopted with a block interleaver of length 256 for eight iterations. With reference to the decoding depth used in the VA, where $\delta = (5\sim10)K$, we take $\delta = 24$ for the simplified MAP algorithms. The simulation result is shown in Figure 4. S-MAP can achieve the performance very close to MAP and much better than SOVA; Max-Log-S-MAP is inferior to S-MAP but better than SOVA, and it comes close to MAP as the value of SNR increases. When SNR = 5.0dB, S-MAP, Max-Log-S-MAP achieve almost similar performance to MAP.

V Conclusion and Discussions

In this paper, we have investigated two low-complexity MAP algorithms for turbo decoding. By proper use of "decoding depth", our algorithms can reduce the computational complexity and decoding delay considerably, compared with the conventional MAP and SOVA algorithms, respectively. The advantages of our simplified algorithms are confirmed by both theoretical analyses and computer simulations assuming a Rayleigh fading environment.

One of the practical concerns regarding the simplified algorithms is the effect of finite decoding depth. Obviously, choosing too large a value for the decoding depth may lead to a significant increase in both computational complexity and storage space; taking too small a value may increase an estimation error. We continue to investigate such a tradeoff between the implementation cost and performance for this class of MAP decoding algorithms.

VI Acknowledgement

This work has been supported, in part, by grants from the national Science Foundation, the NJ Center for Wireless Technology, Asahi Kasei Corporation, Mitsubishi Electric Information Technology Center America, Inc, and the Ogasawara Foundation for the Promotion of Science and Engineering.

Table 1 Comparison between MAP, S-MAP, Max-Log-S-MAP and SOVA

Algorithm		MAP	S-MAP	Max-Log-S-MAP	SOVA
Operation	Addition	-	$M(\delta+2)$	$4M\delta$	$2M$
	Comparison	-	0	$2M(\delta+2)$	$2M+(\delta+1)$
	Multiplication	-	$4M\delta$	0	0
	Exponentiation	Have	Have	None	None
Memory		$2MN$	$M(\delta+1)$	$2M\delta+M$	$2M\delta+M$
Delay		N	δ	δ	δ

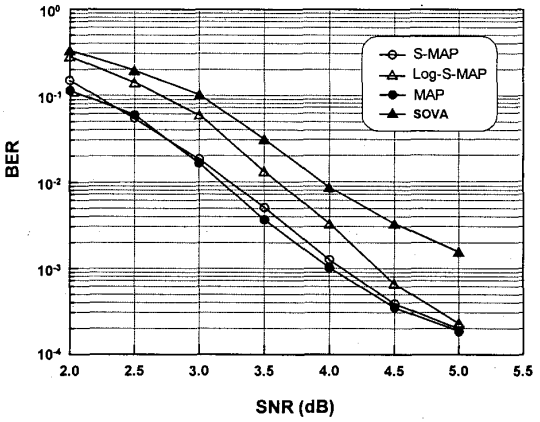

Figure 4 BER performance of 8-state 1/3 turbo code with four algorithms

References

[1] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," IEEE Trans. Inform. Theory, vol.IT-20, pp.284-287, March 1974

[2] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes (1)," Proc. IEEE Int. Conf. on Communications (Geneva, Switzerland, May 1993), pp.1064-1070

[3] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," Proc. GLOBECOM'89, pp.1680-1686, Nov. 1989

[4] P. Robertson *et al*, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," Proc. ICC'95, pp.1009-1013, 1995

[5] X. Wang and S.B. Wicker, "A soft-output decoding algorithm for concatenated systems," IEEE Trans. Inform. Theory, vol.42, no.2, pp.543-553, March 1996

[6] G.D. Forney, "The Viterbi algorithm," Proc. IEEE, vol.61, no.3, pp.268-278, March 1973