Abstract—List decoding of turbo codes is analyzed under the assumption of a maximum-likelihood (ML) list decoder. It is shown that large asymptotic gains can be achieved on both the additive white Gaussian noise (AWGN) and fully interleaved flat Rayleigh-fading channels. It is also shown that the relative asymptotic gains for turbo codes are larger than those for convolutional codes. Finally, a practical list decoding algorithm based on the list output Viterbi algorithm (LOVA) is proposed as an approximation to the ML list decoder. Simulation results show that the proposed algorithm provides significant gains corroborating the analytical results. The asymptotic gain manifests itself as a reduction in the bit-error rate (BER) and frame-error rate (FER) floor of turbo codes.

Index Terms—List decoding, turbo codes.

I. INTRODUCTION

Turbo coding refers to the parallel concatenation of recursive systematic convolutional (RSC) codes in which the data sequence is first encoded by an RSC encoder (RSC1) and then permuted using a pseudorandom interleaver and encoded with an identical RSC encoder (RSC2) [1]. The systematic output, along with the parity outputs of RSC1 and RSC2, are then transmitted over the channel. The parity bits can be punctured to produce higher rate turbo codes.

The main ingredients of turbo coding are the recursive encoder and the pseudorandom interleaver [2]. The remarkable performance of turbo codes is due to spectral "thinning," although turbo codes exhibit a bit-error rate (BER) floor due to their relatively small free distance [3]. Concatenating turbo codes with a Reed–Solomon or Bose–Chaudhuri–Hocquenghem (BCH) outer code can reduce the BER floor [4], [5].

This paper proposes list decoding as another technique for reducing the BER floor of turbo codes. List decoding is typically used with concatenated codes. The inner decoder uses a Viterbi algorithm or some other hard-output decoding algorithm and outputs a list of $L$ probable paths. The outer decoder then selects one of these paths. For packetized transmission, the cyclic redundancy check (CRC) code forms a convenient outer code. Various methods can be used to generate the list. When a soft-output inner decoder is used, the list can be generated by the $2^k$ method [6], a method that we have applied to turbo codes [7]. The $2^k$ method has the disadvantage of ignoring the correlation in the erroneous bit positions due to an error event. The more complex list output Viterbi algorithm (LOVA) [6], [8] is a better method for generating the list of probable paths. A LOVA produces a list of $L$ best paths in the maximum-likelihood (ML) sense. If $L = 1$, the LOVA is merely the Viterbi algorithm. Two variants of the LOVA, namely, the parallel (PLVA) and serial (SLVA), were introduced in [8]. The LOVA was shown to provide significant BER improvement over ML decoding [6], [8].

The remainder of the paper is organized as follows. Section II describes the system model. Sections III and IV extend the analysis in [8] to turbo codes, assuming an ML list decoder. The analysis shows some important differences between the performance of list decoding of convolutional codes and turbo codes. A practical SLVA list decoding algorithm for turbo codes is introduced in Section V. Section VI presents performance results for the SLVA list decoder and, finally, Section VII concludes the paper.

II. SYSTEM MODEL AND PRELIMINARIES

Fig. 1 shows the turbo encoder structure. The data sequence $a$ consists of the data bits, a $K$-bit CRC sequence, and a $v$-bit tail sequence that terminates the trellis of RSC1 in the all-zeros state. The trellis of RSC2 is left "open." The data sequence $a$ is interleaved and encoded. A codeword $x = (x_s, x_{1p}, x_{2p})$ is formed by concatenating the systematic component $x_s$ and the two parity components $x_{1p}$ and $x_{2p}$ of the encoder output. Binary phase-shift keying (BPSK) modulation is assumed.

Two types of channels are considered, the additive white Gaussian noise (AWGN) channel and the fully interleaved flat Rayleigh-fading channel. Assuming perfect symbol synchronization and carrier recovery, the samples at the receiver filter output are

$y_{k1} = \alpha_1 k_1 (2x_{k1} - 1) + \nu_{k1}$,

$y_{k2} = \alpha_1 k_2 (2x_{k2} - 1) + \nu_{k2}$,

and

$y_{2k} = \alpha_2 k_2 (2x_{2k} - 1) + \nu_{2k}$,

where the $\alpha_1 k_1$, $\alpha_1 k_2$, and $\alpha_2 k_2$ are the channel gains and the $\nu_{k1}$, $\nu_{k2}$, and $\nu_{2k}$ are independent zero-mean Gaussian random variables with variance $\sigma^2 = N_0/2$, where $N_0/2$ is the two-
sided noise power spectral density. For the AWGN channel, 
\( \alpha_{y_k} = \alpha_{y_k} = \alpha_{2y_k} = 1 \), while for the fully interleaved Rayleigh-fading channel, \( \alpha_{y_k}, \alpha_{1y_k}, \alpha_{2y_k} \) are independent Rayleigh-distributed random variables.

The samples \( y_k, y_{k+1}, y_{k+2} \) are processed with a turbo decoder that uses the modified maximum a posteriori (MAP) algorithm in [9]. The turbo decoder is an iterative structure consisting of many identical stages, each consisting of two MAP decoding units. Each MAP decoding unit generates the log-likelihood ratio (LLR) for each decoded bit based on the received signal and the extrinsic information generated by the other decoder. The iterative decoding operation can be explained as follows. At the \( m \)th iteration \( m \geq 1 \), the LLRs generated by the MAP decoders for data bit \( a_k \) are

\[
L_1^{(m)}(a_k) = L_{\text{sys}}(a_k) + L_{\text{ext}}^{(m-1)}(a_k) + L_{\text{ext1}}^{(m)}(a_k) \\
L_2^{(m)}(a_k) = L_{\text{sys}}(a_k) + L_{\text{ext}}^{(m-1)}(a_k) + L_{\text{ext2}}^{(m)}(a_k)
\]

(1)

where \( L_{\text{sys}}(a_k) = (2)^{1/2} \alpha_{y_k} y_k \) is the LLR due to the systematic component, and \( L_{\text{ext1}}^{(m)}(a_k) \) and \( L_{\text{ext2}}^{(m)}(a_k) \) are the extrinsic information for each bit generated at the \( m \)th decoding stage by MAP1 and MAP2, respectively. The iterative procedure is started with initial condition \( L_{\text{ext1}}^{(0)}(a_k) = 0 \). The bit decisions at stage \( m \) are made based on \( L_2^{(m)}(a_k) \).

In this paper a CRC is used as a criterion to stop the iterations in the turbo decoder. A serial mode version of the turbo decoder is implemented, such that after each iteration (consisting of a decoding process by MAP1 and a decoding process by MAP2) the decoded data at the output of MAP2 is checked for errors using a CRC. If the CRC is satisfied, then no more decoding stages are required; otherwise, iterative decoding continues for a maximum of \( M \) iterations.

Sometimes it is useful to view convolutional codes and turbo codes as equivalent block codes with length-\( N \) input sequences. For example, the turbo code in Fig. 1 can be viewed as a block code with \( 2^{N-\kappa} \) codewords of length \( n = 3N \), where \( N \) is the interleaver size. The distance spectrum of a block code is specified by the set of ordered pairs \( \{d_i, N_i\} \), where \( N_i \) is the number of codewords with Hamming weight \( d_i \). If the spectrum is arranged in ascending order of \( d_i \), i.e., \( d_1 < d_2 < \cdots < d_N \), then the Hamming weight \( d_1 \) is the free distance of the code and is denoted by \( d_{\text{free}} \). In the sequel, convolutional codes and turbo codes are discussed in parallel and, to avoid confusion, quantities associated with them are distinguished by the superscripts \( c \) and \( T \), respectively.

The interleaving operation in the turbo interleaver is denoted by the mapping \( I : i \rightarrow j \) or, equivalently, \( I(i) = j \). The choice of the turbo code interleaver is very important. For reasonably large interleaver sizes \( N \), random interleavers perform very well [10] and, moreover, they are easy to analyze. Therefore, random interleavers are used in our analysis and simulations.

For random interleavers, weight-two input sequences to the turbo encoder correspond to error events with minimum distance and minimum error event length [2], [10] with high probability. If the smallest Hamming weight of the parity sequences that correspond to all weight-two input sequences is denoted by \( d_{\text{eff}} \), then the free distance of the turbo code is \( d_{\text{free}} = 2 + 2d_{\text{eff}} \). Higher weight input sequences can be easily permuted by the interleaver and do not affect the turbo code performance significantly at high \( E_b/N_o \) [10].

Convolutional codes are time invariant [3]. Consider an error event \( e \), which is a path in the code trellis that diverges from the all-zeros path at epoch \( k \) and remerges with the all-zeros path at a later stage. This error event corresponds to an error in the \( k \)th input bit position. For convolutional codes, there exists an error event \( D_k^c \) (a shift of \( c \) by \( i \) epochs) corresponding to a bit error in the \( k+i \)th bit position for all \( i \). Consequently, \( N_{k}^{c, \text{free}} \approx N \). In contrast, turbo codes are not time invariant. If an error event \( e \) exists that corresponds to a bit error in the \( k \)th bit position, then with high probability there does not exist an error event of the form \( D_k^c \) that corresponds to a bit error in the \( (k+i) \)th bit position [3]. This phenomenon is a direct consequence of the pseudorandom interleaver and, as a result, the number of error events at small Hamming distances is very small, i.e., \( N_{k}^{c, \text{free}}, N_{k-1}^{c, \text{free}}, \cdots, N_{k}^{c, \text{free}}, N_{k}^{c, \text{free}}, \cdots \) from any given codeword is very small. This property of turbo codes makes the performance of list decoders (and other decoders) for turbo codes different from that for convolutional codes.

### III. Performance Analysis for AWGN Channel

Suppose that an ML list decoder is available that outputs a list of \( L \) probable paths in the ML sense. For \( L = 1 \) (ML decoding), the union bound on the probability of codeword (or frame) error is

\[
P_e \leq \frac{N/R}{d_{\text{free}}} Q\left(\sqrt{\frac{2R|E_b d|}{N_o}}\right)
\]

(2)

where \( R \) is the code rate. At high \( E_b/N_o \), the term corresponding to \( d_{\text{free}} \) dominates and therefore

\[
P_e \approx N_{\text{free}} Q\left(\sqrt{\frac{2R|E_b d|}{N_o}}\right).
\]

(3)

As explained previously, \( N_{k}^{c, \text{free}} \approx N \) for convolutional codes, while \( N_{k}^{e, \text{free}} \), \( N_{k-1}^{c, \text{free}} \), \( \cdots, N_{k}^{c, \text{free}} \), \( \cdots \approx N \) for turbo codes.

From (2), the frame-error rate (FER) of turbo codes is very low, even for low-to-moderate \( E_b/N_o \). However, since \( d_{\text{free}}^{c, \text{free}} \) is small, there appears to be an FER floor, the slope being determined by \( d_{\text{free}}^{c, \text{free}} \) in (3). We now proceed to analyze the performance of the list decoder for \( L = 2, 3, \cdots \).

#### A. Worst-Case Asymptotic Gain for \( L = 2 \)

When \( L = 2 \), \( P_e \) is the probability that the correct codeword is not among the two codewords hypothesized by the ML list decoder. In other words, \( P_e \) is the probability that the received signal vector is closer in Euclidean distance to at least two codewords other than the transmitted codeword. The situation is shown in Fig. 2(a) for convolutional codes, where \( A \) represents the transmitted all-zeros codeword, and \( B \) and

\[1\text{In the following analysis for convolutional and turbo codes the edge effects due to the } \nu \text{-bit tail sequence are ignored.}

\[2\text{Here we assumed that there is only one possible error event with Hamming distance } d_{\text{free}}^{c, \text{free}} \text{ that causes an error in the } k \text{th bit position.} \]
Fig. 2. Geometrical interpretation of neighboring codewords for (a) convolutional codes and (b) turbo codes.

are two codewords closest to in Euclidean distance. The three signal points , , and form a triangle. Let , , and be the Euclidean distance between the signal points , , and , respectively. For BPSK, the Euclidean distance is related to the Hamming distance by . The region in which the received signal point should lie for a decoding error to occur is defined by the shaded area in Fig. 2, which is all of the points that satisfy . The point closest to that can cause a decoding error is , which is the intersection of the perpendicular bisectors of the sides and . The distance between and is [8]

For convolutional codes, there always exist two codewords and such that . In fact, there exist many such and . The worst-case or smallest occurs when , in which case the asymptotic gain is the same as for convolutional codes as given by (7).

Case 2—: In this case there is only one codeword , and the geometric interpretation is a 3D tetrahedron with four codewords , , , and , as shown in Fig. 3. For convolutional codes, the worst case occurs when and , while . The tetrahedron is irregular with . Therefore, the worst-case asymptotic gain for turbo codes is

Hence, the worst-case asymptotic gain for turbo codes is

A comparison of (7) and (9) shows the difference between the asymptotic performance of list decoding for convolutional and turbo codes. The gain is independent of distance spectrum, while depends on the distance spectrum. In the worst case it depends on the ratio . Since , it follows that . Therefore, list decoding provides higher relative gains when used with turbo codes than with convolutional codes. Note that if the interleaver permutations are randomly chosen, the probability that is approximately [11].

Example 1: Consider the rate-1/3 four-state turbo code with generator polynomials . This code has and . Therefore, dB while dB.

B. Worst-Case Asymptotic Gain for

For , is the probability that the received signal vector is closer in Euclidean distance to at least three codewords other than the transmitted codeword. The geometric interpretation is a three-dimensional tetrahedron with four codewords , , and , in which case is equilateral and simplifies to [8]

From (5), list decoding is seen to have the effect of increasing and, hence, improve the FER performance. A quantitative measure of the performance improvement is the worst-case asymptotic gain for a list of size , defined as [8]

For turbo codes two situations are considered.

Case 1—: In this case there are at least three codewords at Hamming distance from . In the worst case the tetrahedron is regular (all of the sides have same length ). Therefore, the worst case occurs when and , while . The tetrahedron is irregular with .
The worst-case gain for turbo codes is (see the appendix)

\[ G_T^f = 10 \log \left( \frac{4}{3} + \frac{4 \left( \frac{d_{\text{free}}^f}{d_{\text{free}}} - 2 \right)^2}{36 \left( \frac{d_{\text{free}}^f}{d_{\text{free}}} - 12 \right)} \right) \]  

(11)

For the turbo code considered in Example 1, \( d_{\text{free}}^f = 10 \) and \( d_{\text{free}}^f = 12 \), and, therefore, \( G_T^f = 2.2 \) dB while \( G_3^f = 1.76 \) dB.

C. Worst-Case Asymptotic Gain for \( L > 3 \)

For convolutional codes, the worst case occurs when the geometric interpretation is an \( L-D \) simplex structure. For turbo codes, the geometric interpretation is a nonsimplex structure if \( N_{\text{free}}^T < L \). In general, the relative gains for convolutional codes \([8]\) and turbo codes are given by

\[ G_L^T = 10 \log \left( \frac{2L}{L+1} \right) \]

\[ G_T^T = G_L^T, \quad \text{for } N_{\text{free}}^T \geq L \]

\[ G_T^T > G_L^T, \quad \text{for } N_{\text{free}}^T < L. \]  

(12)

If the interleaver permutations are selected at random, \( N_{\text{free}}^T \) will be equal to two on average \([11]\). Therefore, on average, \( L > 2 \) should provide large asymptotic gains.

IV. PERFORMANCE ANALYSIS FOR RAYLEIGH-FADING CHANNEL

This section derives the worst-case gain for a fully interleaved flat Rayleigh-fading channel with perfect channel state information. In this case the FER is determined primarily by the time diversity of the code and is given by \([12]\)

\[ P_e \leq \frac{N_d}{2} \left[ 1 + \left( \frac{RE}{N_o} \right)^d \right]^d \]  

(13)

where \( d \) is the time diversity of the code. For \( L = 1, d = d_{\text{free}} \) and, hence, the free distance of the code determines the FER performance. Our analysis for \( L = 2, 3, \ldots \) again follows \([8]\).

We treat the fully interleaved flat Rayleigh-fading channel as an erasure channel when the channel is in a fade. Then \( d = 1 \) is the maximum number of erasures that can be allowed such that the codewords are still distinguishable.

A. Worst-Case Gain for \( L = 2 \)

\( P_e \) is the probability that the correct codeword is not among the two codewords hypothesized by the list decoder. This can be evaluated by determining the minimum number of erasures such that the correct codeword \( A \) and two nearest neighbors \( B \) and \( C \) will be indistinguishable. Instead of using geometry, we consider the code trellis. Such a diagram is shown in Fig. 4 for three codewords \( A, B, \) and \( C \). At epoch \( i \), codeword \( B \) diverges from the reference codeword \( A \) and it remerges with \( A \) at epoch \( i + L_e \), where \( L_e \) is the length of the error event. Likewise, at epoch \( j \) codeword \( C \) diverges from \( A \). The worst-case scenario occurs for some \( j = j' \), such that \( i < j' < i + L_e \) and \( i + L_e < j' + L_e < i + 2L_e \). For convolutional codes, such codewords \( B \) and \( C \) always exist (namely, \( C = D^{j-i}B \) and the worst case occurs when \( d_{AB} = d_{AC} = d_{BC} = d_{\text{free}} \) and \( B \) and \( C \) differ from \( A \) in \( \lfloor (3/2)d_{\text{free}} \rfloor \) bit positions \([8]\), leading to an effective time diversity of \( d_{\text{eff}} = \lfloor (3/2)d_{\text{free}} \rfloor \).

We will discuss the performance of turbo codes with \( L = 2 \) through an example. Consider again the turbo code in Example 1. Let \( A, B, \) and \( C \) be three codewords and \( A_i, B_i, \) and \( C_i \) be the corresponding input sequences. Let \( B_{1p}, B_{2p}, \) and \( B_{2p} \) denote the systematic component and the two parity components of codeword \( B \). Finally, let \( B_i \) denote the interleaved version of \( B \). Let \( A_i \) be the all-zeros input sequence and, consequently, let \( A \) be the all-zeros codeword. Let \( B \) correspond to a minimum weight codeword. Hence, \( B_i \) is of the form \( D^i(1 + D^3) \) such that \( B_i = D^{i}(1 + D^3) \) and, therefore, \( B_{1p} = D^i(1 + D^3) \), \( B_{2p} = D^i(1 + D + D^2 + D^3) \), and \( B_{2p} = D^i(1 + D + D^2 + D^3) \). Let \( C_i \) be an input sequence of the form \( D^{m}(1 + D^3) \) such that \( C_i = D^{m}(1 + D^3) \). Then \( C_{1p} = D^{m}(1 + D^3) \), \( C_{1p} = D^{m}(1 + D + D^2 + D^3) \), and \( C_{2p} = D^{m}(1 + D + D^2 + D^3) \). The worst-case time diversity occurs when \( m = i + 1 \) and \( n = j + 1 \). That is, \( B \) and \( C \) should correspond to error events of the form in Fig. 4 in both component encoders. When such an interleaver mapping occurs, \( B \) and \( C \) together differ from \( A \) in \( 4 + 5 + 5 = 14 \) bit positions and, consequently, the worst-case time diversity for \( L = 2 \) is 14. However, for a random interleaver, the probability that such an interleaver mapping occurs, that is, \( B_i = D^i(1 + D^3) \), \( B_i = D^i(1 + D^3) \), \( C_i = D^{m}(1 + D^3) \), \( C_i = D^{m}(1 + D^3) \), \( C_{2p} = D^{m}(1 + D + D^2 + D^3) \), \( C_{2p} = D^{m}(1 + D + D^2 + D^3) \), and \( C_{2p} = D^{m}(1 + D + D^2 + D^3) \), where \( m = i + 1 \) and \( n = j + 1 \), is extremely small. We now show this by finding the fraction of all possible interleavers in which we can find such mappings.

Again, the edge effects due to the finite length \( N \) are ignored. The total number of possible interleavers of length \( N \) is \( N! \). If an interleaver is chosen randomly, the \( i \)th position can be mapped onto any one of the \( N \) positions and, therefore, there are \( N \) choices for \( I(i) \). Since the sequence \( D^i(1 + D^3) \) must be mapped to \( D^{i}(1 + D^3) \), position \( i + 3 \) should be mapped to position \( j \) and, therefore, there are two choices.
for $I(i + 3)$. Furthermore, for the worst-case time diversity, position $i + 1$ should be mapped to one of the positions $j - 1, j + 1, j + 2, j + 4$ and, therefore, there are four possible choices for $I(i + 1)$. Once $I(i)$, $I(i + 3)$, and $I(i + 1)$ are fixed, there is only one possible position for the $i + 4$th position such that $B$ and $C$ differ in the minimum possible number of bit positions from $A$. Once the mapping for these bit positions is fixed, there are $N - 4$ other bit positions which can be permuted in $(N - 4)!$ ways. Therefore, the fraction of all possible interleavers in which, for a given $i$, the two input sequences $B_i = D^i(1 + D^3)$ and $C_i = D^{i+1}(1 + D^3)$ both get mapped onto sequences of the same form is

$$P = \frac{N \times 2 \times 4 \times 1 \times (N - 4)!}{N!} \approx \frac{8}{N^3} \tag{14}$$

which is extremely small. The probability of finding such a mapping for at least one $i$ is approximately $1 - \left(1 - \frac{8}{N^2}\right)^N \approx 8/N^2 \tag{10}$. Therefore, for large $N$, the probability that a randomly chosen interleaver results in a minimum time diversity of 14 is extremely small. Likewise, the probability that other input sequences produce overlapping error events in both of the component encoders is also extremely small.

For large $N$, a more probable situation for small time diversity is the event when the minimum weight codewords $B$ and $C$ do not overlap in time in the trellis of either of the component codes, that is, $|i - m| > 3$ and $|j - n| > 3$. In such a case the time diversity is

$$d_{\text{eff}}^2 = 4 + 2d_{\text{eff}} + 2d_{\text{eff}} = 2d_{\text{tree}} \tag{15}$$

where $d_{\text{eff}}$ is defined in Section II. It is obvious that the increase in time diversity for turbo codes is larger than that for convolutional codes and, hence, a significant improvement in performance should result. This claim will be substantiated by the results in Section VI.

V. LIST DECODER FOR TURBO CODES

The analysis in the previous sections assumed an ML list decoder. However, ML decoding algorithms for turbo codes are not available and, hence, ML list decoders for turbo codes are not available either. An approximation to the ML list decoder is now introduced that can be implemented in the form of an SLVA [6].

Our proposed overall receiver structure is formed by combining a list decoder with a conventional turbo decoding algorithm, which is implemented in serial mode. Upon receipt of a packet, the basic receiver operations are as follows. After each iteration in the turbo decoder (one decoding process by MAP1 and one decoding process by MAP2), the decoder output of MAP2 is checked for errors using the CRC. If the CRC is satisfied in $M$ or fewer iterations, then the packet is accepted. If a packet cannot be successfully decoded in $M$ iterations, then the list decoder is invoked at the end of the $M$th iteration.

A list decoder is proposed in which the extrinsic information produced by MAP2 at stage $M$ $L_{\text{ext},2}^{(M)}(a_k)$ is used as a priori information in an SLVA that generates a list of candidate paths for RSC1 (including the systematic and the parity components). The proposed list decoder then has a structure that is identical to the SLVA in [6], such that the branch metric for the transition from state $S'$ at epoch $k - 1$ to $S$ at epoch $k$ is

$$\gamma(S', S, a_k) = \log \left\{ \frac{\exp[x_{sk}L_{\text{ext},2}^{(M)}(a_k)]}{1 + \exp[L_{\text{ext},2}^{(M)}(a_k)]} \right\}$$

$$- \frac{1}{N_o} \left[ y_{sk} - \alpha_{sk}(2x_{sk} - 1)^2 \right]$$

$$+ \left[ y_{1pk} - \alpha_{1pk}(2x_{1pk} - 1)^2 \right] \tag{16}$$

where $a_k$ is the information bit associated with the transition from state $S'$ to $S$. By deleting all terms independent of $x_{sk}$ and $x_{1pk}$, the above metric simplifies to

$$\gamma(S', S, a_k) = x_{sk}L_{\text{ext},2}^{(M)}(a_k) + \frac{4}{N_o} \left\{ \alpha_{sk}y_{sk}x_{sk} + \alpha_{1pk}y_{1pk}x_{1pk} \right\} \tag{17}$$

Without the first term in (17), the algorithm is identical to the SLVA for RSC1 alone with the systematic and parity components. The first term represents the contribution of the parity symbols of RSC2 or extrinsic information. Note that the branch metric in (17) is equivalent to the branch metric used in the turbo decoder based on the soft-output Viterbi algorithm (SOVA) in [13].

The SLVA can also be used with RSC2 by using $L_{\text{ext},1}^{(M)}(a_k)$ as a priori information in the branch metric. It is also possible to generate lists of probable paths by using an SLVA with RSC1 and RSC2 independently, and then selecting $L$ globally best paths. Our simulations have used the extrinsic information from MAP2 as a priori information in an SLVA for RSC1. After generating each of the paths in the list, the path is checked for errors using the CRC. If the CRC is satisfied, then the packet is accepted; otherwise, the next element in the list is generated for up to a maximum of $L$ paths.

VI. SIMULATION RESULTS

The receiver in Section V was tested using a rate-1/3 four-state turbo code with generator polynomials $[1, (1 + D^2)/(1 + D + D^2)]$. Block lengths of 512 and 1696 bits were considered. A random interleaver (constructed by generating random numbers between 0 and $N - 1$) was used in all of the simulations. A 16-bit and a 24-bit CRC code was used as the outer code for block lengths of 512 and 1696, respectively. Approximately $2.5 \times 10^6$ frames were simulated at low FER’s.

Figs. 5 and 6 compare the FER performance of the proposed receiver with $L = 3$ to a conventional turbo decoder for block lengths of 512 and 1696, respectively. The performance of the $2^k$ list decoder [6], [7] is also shown for comparison. Note that significant gains can be achieved by list decoding even for a small list size of $L = 3$. For a block length of 1696 and an $E_b/N_0$ of 3.0 dB for the conventional turbo decoder, the gain achieved by the list decoder is about 1.6 dB while the theoretical asymptotic gain is 2.2 dB.

Figs. 7 and 8 show the FER comparisons for a fully interleaved flat Rayleigh-fading channel for block lengths of 512 and 1696, respectively. Large asymptotic gains are

3 The length of 1696 bits corresponds to four ATM cells.
achieved even with a small list size and the gains are larger for the Rayleigh-fading channel than the AWGN channel.

Fig. 9 compares the performance of the list decoder with that of a conventional receiver on an AWGN channel for $M = 3$ and $M = 7$, and a block length of 512. Note that the relative performance gains are higher for $M = 3$. More interesting is the fact that for moderate-to-high $E_b/N_0$, better performance can be achieved by using a conventional turbo decoder with three iterations followed by a list decoder than by using seven iterations of the conventional turbo decoder alone. That is, better performance can be achieved compared to the conventional turbo decoder with a reduced complexity.

Fig. 10 shows the performance for a rate-1/3 16-state turbo code with generator polynomials $[1, (1+D^2)/(1+D+D^2+D^3+D^4)]$ and a block length of 512. The performance of the rate-1/3 four-state code with generator polynomials $[1, (1+D^2)/(1+D+D^2)]$ is also shown for comparison. The interleavers were picked such that $d_{\text{free}}^{T} = 10$, $d_{\text{free}}^{F} = 12$.

$N_{\text{tree}}^{T} = 2$, and $N_{\text{tree}}^{F} = 2$ for both turbo codes. Although $d_{\text{free}}^{T}$, $d_{\text{free}}^{F}$, $N_{\text{tree}}^{T}$, $N_{\text{tree}}^{F}$ are the same for both turbo codes, it can be seen that the performance improvement for the 16-state code is higher than that for the four-state code. One possible reason is that higher weight codewords contribute more to the FER for the four-state code than for the 16-state code and, therefore, the improvement in FER performance for the 16-state code is greater. Another possible reason is that the performance of the iterative decoding algorithm is better for the 16-state code than for the four-state code. Since the iterative decoder affects the performance of the list decoder, the performance gains are higher for the 16-state code. However, it appears from Fig. 10 that the asymptotic (high $E_b/N_0$) gains of the list decoder for both codes will be comparable.

Fig. 11 shows the performance of the proposed list decoder for the four-state code and an AWGN channel for $L = 3$, 16,
Fig. 9. FER comparison with list decoding on an AWGN channel for $M = 3$ and $M = 7$.

Fig. 10. FER comparison with list decoding on an AWGN channel for four-state and 16-state codes.

Fig. 11. FER comparison with list decoding on an AWGN channel, $M = 7$, $L = 3, 16$, and 64.

Fig. 12. BER comparison with list decoding on an AWGN channel, $N = 1696$.

Note that the performance improvement for $N = 512$ is much less than that predicted from theory. One possible reason is that $N_{\text{free}}^T$, $N_{1}^T$, ..., decrease with increasing interleaver length [3], i.e., large interleavers are required to break up input sequences, thereby keeping $N_{\text{free}}^T$, $N_{1}^T$, ..., small. Another possible reason is that the performance of the turbo decoding algorithm is better for larger block lengths. At this point it should be noted that the gains computed from theory are asymptotic gains, while the simulation results do not show the asymptotic performance, since beyond a certain $E_b/N_0$, the FER is so small that computer simulations cannot be conducted. If properly extrapolated, the simulation results will be closer to the theoretically predicted gains. Nevertheless, list decoding yields significant performance gains over conventional turbo decoding.

It is important to note that although the asymptotic relative gain for turbo codes are higher than those for convolutional

and 64. Increasing the list size beyond $L = 3$ does not improve the performance as significantly as predicted by theory. The reason is that there are some occasional uncorrectable error patterns at the output of the turbo decoder due to the fact that the turbo decoder and the proposed list decoder are not ML decoders. The probability of occurrence of such error events upper bounds the performance improvement. To compute the probability of occurrence of such error events requires an analysis of the possible error patterns at the output of the turbo decoder, which is still an open problem.

Fig. 12 shows the BER performance of the proposed receiver. The improvement is significant at high $E_b/N_0$ but the FER is improved more than the BER. This is due to the fact that uncorrectable error patterns sometimes correspond to many bit errors within a packet. While such cases increase the BER, they result in only one frame error.
codes, this does not mean that for a given BER or FER the gains for turbo codes are larger than those for convolutional codes. This is because different codes will require a different $E_b/N_0$ to achieve the same FER. However, for a given $E_b/N_0$, the relative gains for turbo codes are larger than those for convolutional codes.

Finally, a note on the computational complexity. Most of the complexity of the SLVA is due to the Viterbi algorithm. If the SLVA is used instead of the last MAP decoding module, then overall complexity of the SLVA will be comparable to (slightly less than) the complexity of the conventional turbo decoder. Also, the SLVA is invoked only when the packet cannot be decoded in $M$ iterations of the turbo decoder, which occurs only infrequently. Another appealing aspect of the SLVA is that the basic structure of the algorithm is similar to the MAP, SOVA, etc., and, hence, it can be easily implemented as an add-on feature to the conventional decoding hardware.

VII. CONCLUSIONS

This paper has analyzed the performance of ML list decoding of turbo codes. Large asymptotic gains are possible for both the AWGN and the fully interleaved fast Rayleigh-fading channels, and the asymptotic performance gains achieved for convolutional codes are a lower bound on the asymptotic performance gains for turbo codes. The gains for turbo codes are typically higher than those for convolutional codes. We have proposed an approximation to the ML list decoder that can be implemented in the form of an SLVA. Finally, we have shown that the proposed list decoder provides a significant reduction in FER on both the AWGN and fully interleaved fast Rayleigh-fading channels, with almost the same computational complexity as the conventional turbo decoder.

APPENDIX

This appendix derives the worst-case gain $G_3^T$ for the AWGN channel when $N_{\text{free}} = 2$. Fig. 3 shows the geometrical interpretation. The worst case occurs when $A, B,$ and $C$ are at a distance of $D_1 = \sqrt{4E_d/d_{\text{free}}}$ from each other, and $E$ is at a distance of $D_2 = \sqrt{4E_d/d_{\text{free}}}$. Without loss of generality, assume that the coordinates for $A, B,$ and $C$ are given by $(0, 0, 0)$, $(D_1, 0, 0)$, and $(D_1/2, \sqrt{3}D_1/2, 0)$. The signal point closest to $A$ such that it is closer to $B, C,$ and $E$ than to $A$ is given by the intersection of the lines perpendicular to and passing through the midpoints of the lines $AB$, $AC$, and $AE$. It can be seen that this is equivalent to the intersection of the plane $P$ and the line through $F$ perpendicular to the $x-y$ plane. The coordinates of $F$, the centroid of the equilateral triangle $ABC$, are given by $(D_1/2, D_1/2\sqrt{3}, 0)$. Let the coordinates of $E$ be $(x_E, y_E, z_E)$, then $x_E = D_1/2$ and $y_E = D_1/2\sqrt{3}$. From $\triangle AFE$, $FE^2 = AE^2 - AF^2$ and, therefore, $z_E = \sqrt{D_1^2 - D_2^2}/\sqrt{3}$. The equation of the plane $P$ (which passes through the midpoint of $AE$ and is perpendicular to $AE$) is

$$x_E(x - x_E/2) + y_E(y - y_E/2) + z_E(z - z_E/2) = 0 \tag{18}$$

which can be simplified to

$$x_E x + y_E y + z_E z = D_2^2/2. \tag{19}$$

The intersection of $P$ with $FE$ is obtained by solving (19) for $z$ with $x = D_1/2$, $y = D_1/2\sqrt{3}$, yielding

$$z = \frac{1}{z_E} \left( \frac{D_2^2}{2} - \frac{D_1^2}{3} \right) = \frac{3D_2^2 - 2D_1^2}{\sqrt{3}D_2^2 - 12D_1^2}. \tag{20}$$

Therefore, the distance between $A$ and $D_{\text{eff}}$ is

$$D_{\text{eff}}^2 = \frac{D_1^2}{3} + \frac{3D_2^2 - 2D_1^2}{\sqrt{3}D_2^2 - 12D_1^2}. \tag{21}$$

The asymptotic gain defined in Section III is obtained by substituting $D_1^2 = 4E_d/d_{\text{free}}$ and $D_2^2 = 4E_d/d_{\text{free}}$, giving

$$G_3^T = 10 \log \frac{D_{\text{eff}}^2}{D_1^2/4} = 10 \log \left\{ \frac{4}{3} + \frac{\left( \frac{3D_2^2}{D_1^2} - 2 \right)^2}{\sqrt{3}D_2^2 - 12D_1^2} \right\}. \tag{22}$$

For convolutional codes, the asymptotic gain is obtained by substituting $D_2 = D_1$, giving

$$G_3^C = 10 \log \frac{3}{2}. \tag{23}$$

REFERENCES


Krishna R. Narayanan received the B.E. degree in electronics and communication engineering from Coimbatore Institute of Technology, India, in 1992, and the M.S.E.E. degree from Iowa State University, Ames, in 1994. He is currently working toward the Ph.D. degree at the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta.

His research interests include modulation and coding, equalization, and interference rejection.

Gordon L. Stüber (S’81–M’82–SM’96) received the B.A.Sc. and Ph.D. degrees in electrical engineering from the University of Waterloo, Waterloo, Ont., Canada, in 1982 and 1986, respectively.

In 1986 he joined the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, where he is currently a Professor. His research interests are in wireless communications and communication signal processing. He has authored the textbook Principles of Mobile Communication (Boston, MA: Kluwer, 1996).

Dr. Stüber has served as an Editor for Spread Spectrum of the IEEE TRANSACTIONS ON COMMUNICATIONS from January 1993 through December 1997. He served as the Technical Program Chair for the 1996 IEEE Vehicular Technology Conference. He is currently serving as the Technical Program Chair for the 1998 IEEE International Conference on Communications to be held in Atlanta, GA, June 7–11, 1998.