

# Performance Analysis and Design Criteria for Finite-Alphabet Source-Channel Codes

Ahmadreza Hedayat, *Student Member, IEEE*, and Aria Nosratinia, *Senior Member, IEEE*

**Abstract**—Efficient compression of finite-alphabet sources requires variable-length codes (VLCs). However, in the presence of noisy channels, error propagation in the decoding of VLCs severely degrades performance. To address this problem, redundant entropy codes and iterative source-channel decoding have been suggested, but to date, neither performance bounds nor design criteria for the composite system have been available. We calculate performance bounds for the source-channel system by generalizing techniques originally developed for serial concatenated convolutional codes. Using this analysis, we demonstrate the role of a recursive structure for the inner code and the distance properties of the outer code. We use density evolution to study the convergence of our decoders. Finally, we pose the question: Under a fixed rate and complexity constraint, when should we use source-channel decoding (as opposed to separable decoding)? We offer answers in several specific cases. For our analysis and design rules, we use union bounds that are technically valid only above the cutoff rate, but interestingly, the codes designed with union-bound criteria perform well even in low signal-to-noise ratio regions, as shown by our simulations as well as previous works on concatenated codes.

**Index Terms**—Concatenated coding, iterative decoding, joint source-channel coding, variable-length codes (VLCs).

## I. INTRODUCTION

**I**N THIS PAPER, we consider the problem of the transmission of discrete, finite-alphabet sources over a noisy channel. Since efficient entropy codes are often variable-length codes (VLCs), a conventional channel decoding followed by a typical symbol-by-symbol entropy decoding will result in error propagation, thus a single uncorrected channel error may result in a long sequence of data errors. This difficulty has led to a search for error-resilient entropy codes. A prominent example is the reversible (R)VLC [1], used in the video-coding standard H.263+ and its descendants. RVLC consists of a class of codes that have

Paper approved by F. Alajaji, the Editor for Source and Source-Channel Coding of the IEEE Communications Society. Manuscript received August 2, 2003; revised February 29, 2004, and June 4, 2004. This work was supported in part by the National Science Foundation under Grant CCR-9985171. The work of A. Hedayat was also supported in part by the Texas Telecommunications Engineering Consortium (TxTEC). This paper was presented in part at the Allerton Conference on Communications, Control, and Computing, Monticello, IL, 2002 and in part at the IEEE International Conference on Communications, Anchorage, AK, 2003.

The authors are with the Multimedia Communications Laboratory, University of Texas at Dallas, Richardson, TX 75083-0688 USA (e-mail: hedayat@utdallas.edu; aria@utdallas.edu).

Digital Object Identifier 10.1109/TCOMM.2004.836562

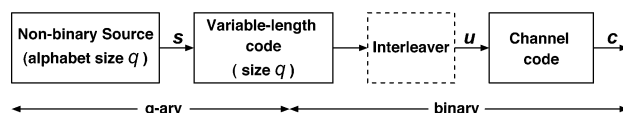


Fig. 1. System block diagram.

not only a prefix property, but also a suffix property, thus they can be decoded from both directions.

A more comprehensive attempt at introducing error resilience into variable-length entropy codes was made by Buttigieg [2], [3], who studied the general class of entropy codes with error-correction ability, and introduced various sequence-decoding algorithms. Subbalakshmi and Vaisey also provided a trellis for describing VLCs and introduced an optimal maximum *a posteriori* (MAP) probability decoder for variable-length encoded sources over a binary symmetric channel (BSC) [4], [5].

Error-resilient codes mentioned above are not strong enough to handle the error rates generated by most communication channels, thus a separate layer of channel coding is usually necessary (see Fig. 1). In such a concatenated system, iterative decoding methods, originally introduced for channel codes [6], [7], provide another opportunity for improved source-channel coding. To the best of our knowledge, the first attempt at iterative decoding of source and channel codes is due to Bauer and Hagenauer [8], [9], who proposed an iterative (turbo) decoding scheme between a channel code and the residual redundancy of an RVLC.<sup>1</sup> They reported a significant coding gain compared with a system with an equivalent transmission rate. Guyader *et al.* [11] proposed various algorithms in the framework of Bayesian networks for the iterative decoding of the chain in Fig. 1 with a Markov nonbinary source. Lakovic and Villasenor [12] studied the performance of VLCs followed by turbo codes, and suggested combining the trellises of the VLC and the upper convolutional code of the turbo code for more coding gain.

Despite many interesting and useful results, including those mentioned above, to date, neither a comprehensive analysis nor design criteria have been available for iteratively decoded source-channel coding systems. In this paper, we analyze this concatenated system, study design criteria for the constituent codes, and present comparisons of various tradeoffs in the design of such codes, supported by extensive simulations.

<sup>1</sup>For an example of iterative source-channel decoding of *fixed-length* codes, see [10].

To start, we generalize the source-channel structure by assigning the error-correcting entropy codes of Buttigieg as the outer code of the source-channel concatenated system. We study, via simulations, the performance of this generalized system.

The central contribution of this paper, however, is an analysis of the performance of the concatenated source-channel codes. We employ the techniques originally developed for serially concatenated convolutional codes (SCCCs) [7], with the critical difference that our outer codes (and hence, our overall codes) are nonlinear, thus the techniques of [7] need to be appropriately extended. Our analysis is general with respect to the choice of outer VLCs and inner channel codes. The outer code can be an RVLC similar to that in [8], [9], or [12], it can be a VLC with higher redundancy, similar to the codes introduced in [2], or a VLC with minimum redundancy, such as Huffman codes. The analysis clarifies the roles of the inner and outer codes in the overall performance, allowing us to make statements about the free distance of the outer code and the desirability of a recursive structure for the inner code. To the best of our knowledge, these or similar results have not been previously reported in the literature on source-channel coding.

Our analysis and design rules are based on union bounds, which are useful in the medium-to-high signal-to-noise ratios (SNRs), in particular above the cutoff rate. However, it has been observed and reported [7] that concatenated codes designed with union-bound criteria perform well even in low-SNR regions. We have observed the same phenomenon for our codes.

The analysis and simulations presented in this paper enable us to make several observations with practical implications. For example, the method of Bauer and Hagenauer [9] achieved significant gain compared with systems with a similar rate. We found, however, that it is possible to improve on the scheme of [9], while maintaining the same overall rate and complexity, by using separable source decoding and an iteratively decoded SCCC. Thus, in this case, investing computational resources into the channel decoder alone gives better returns in terms of system performance. This suggests that whenever the entropy code has small free distance (such as the RVLC used in [9]), one may be better off spending the computational budget mostly on the inner code and not on iterative decoding between source and channel codes. We also found that even with outer codes having larger free distance, iterative source-channel decoding may yield only a slight advantage compared with a separable baseline system of equivalent rate and complexity. These findings are expressed in more detail in the following.

## II. VLCs WITH ERROR-CORRECTING CAPABILITY

Buttigieg [2] introduced a class of entropy codes with error-correction ability under the name of variable-length error-correcting codes (VLECCs). These codes have entropy coding property, in the sense that low-probability symbols have longer codewords. On the other hand, these codes also have error-correction capability, arising from a careful assignment of codewords to symbols such that a minimum Hamming distance is maintained between all codeword pairs. Obviously, maintaining a minimum distance introduces redundancy into the code, such that its average length will be bounded away from the entropy of the source.

Consider a  $q$ -ary source with elements denoted by  $u$  and a VLC whose codewords are denoted by  $b(u)$ . The minimum and maximum length of  $b(u_i)$ 's are denoted by  $\ell_{\min}$  and  $\ell_{\max}$ , respectively, and the average length by  $\ell_{\text{ave}}$ . To perform maximum-likelihood decoding, we need to consider a sequence of  $K$  codewords. We now define such composite codewords. Assume the source sequence  $\mathbf{u} = (u_i : i = 1, \dots, K)$  is entropy-encoded to the bit sequence

$$\mathbf{c} = (b(u_1), b(u_2), \dots, b(u_K)) = (c_1, c_2, \dots, c_N).$$

Because the codewords  $b(u)$  are variable length, the length of the output sequence  $\mathbf{c}$ , denoted by  $N$ , is variable. This leads to difficulty in analysis, therefore, we partition the overall code  $\mathcal{C}$  into subcodes  $\mathcal{C}_i$  such that each partition consists only of codewords of length  $i$ . The *free distance* of  $\mathcal{C}$ , denoted  $d_f$ , is defined as the minimum value of the minimum Hamming distances of the individual binary codes  $\mathcal{C}_i$ . Note that  $\mathcal{C}_i$  are, in general, nonlinear codes.

Buttigieg [2] calculates the upper bound for the error-event probability of a VLC in the same manner as convolutional codes, by introducing the average number of converging paths on an appropriate trellis at a given Hamming distance. Unfortunately, this approach is not appropriate for our purposes since, unlike Buttigieg, we intend to use the VLCs in concatenation with another code. Instead, we use the codeword-enumeration technique [7]. Considering that  $N$ , the length of the bit-sequence  $\mathbf{c}$ , is a random variable that takes value in  $[N_{\min}, N_{\max}] = [\ell_{\min}K, \ell_{\max}K]$ , the upper bounds for the codeword-error probability  $P_E$  and symbol-error probability  $P_S$  are

$$\begin{aligned} P_E &\leq \sum_{N=N_{\min}}^{N_{\max}} \Pr(N) \sum_{h \geq d_f} A_h(N) P_h \\ &= \sum_{h \geq d_f} \left( \sum_{N=N_{\min}}^{N_{\max}} \Pr(N) A_h(N) \right) P_h \end{aligned} \quad (1)$$

$$\begin{aligned} P_S &\leq \frac{1}{K} \sum_{N=N_{\min}}^{N_{\max}} \Pr(N) \sum_{h \geq d_f} B_h(N) P_h \\ &= \frac{1}{K} \sum_{h \geq d_f} \left( \sum_{N=N_{\min}}^{N_{\max}} \Pr(N) B_h(N) \right) P_h \end{aligned} \quad (2)$$

where  $P_h$  is the pairwise error probability (PEP), which has value  $P_h = 0.5 \operatorname{erfc}(\sqrt{hE_s/N_0})$  in the additive white Gaussian noise (AWGN) channel, and  $A_h(N), B_h(N)$  are multiplicities. Specifically,  $A_h(N)$  is the number of codeword pairs in  $\mathcal{C}_N$  with Hamming distance  $h$ . Eventually, we are interested in the distance between symbol strings corresponding to codeword pairs with Hamming distance  $h$ . The average contribution of two codewords of Hamming distance  $h$  to the Levenshtein distance is denoted  $B_h(N)$ . Note that  $A_h(N)$  and  $B_h(N)$  are normalized by the size of the respective codebooks  $\mathcal{C}_N$ . The exchange of summations in (1) and (2) allow us to think of the terms inside parentheses as equivalent  $A_h$  and  $B_h$  for the entire code, without the need to consider individual code partitions separately. It is, in fact, more convenient to calculate  $A_h, B_h$  instead of  $A_h(N), B_h(N)$ ; see, for example, [2] and [8].

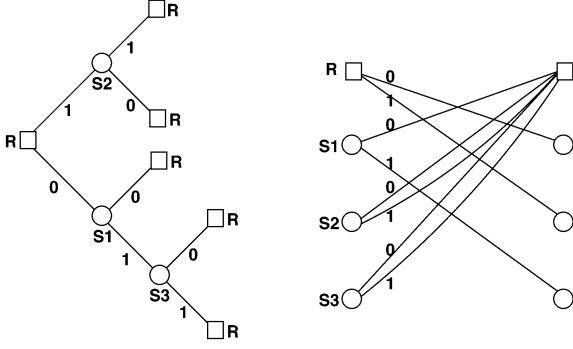


Fig. 2. Tree and bit-level trellis for  $C_1 = \{00, 11, 10, 010, 011\}$ .

The computation of  $B_h$  is based on the Levenshtein distance between the two symbol sequences,  $d_L(\mathbf{u}_i, \mathbf{u}_j)$ . The Levenshtein distance is defined as the minimum number of insertions, deletions, or substitutions to transform one symbol sequence into another [2], [13]. The Levenshtein distance is widely used as an error measure for VLCs, justified partly in light of the self-synchronization property of VLCs [2], and partly because of the lack of other more meaningful and useful distance measures for VLCs.

### III. TRELLIS REPRESENTATION OF VLCs

In this paper, we employ the bit-level trellis proposed by Balakirsky [14] and later used by Murad and Fuja [15], as well as Bauer and Hagenauer [8]. This trellis is obtained simply by assigning the states of the trellis to the nodes of the VLC tree. The root node and all terminal nodes are assumed to represent the same state, since they all show the start of a new sequence of bits. Other nodes, the so-called internal nodes, are assigned one by one to the other states of the trellis. The number of states of the trellis is equal to the number of internal nodes of the tree plus one. As an example, Fig. 2 shows the trellis corresponding to a Huffman code  $C_1 = \{00, 11, 10, 010, 011\}$ .

### IV. SERIAL CONCATENATION OF VLC AND CHANNEL CODES

In the following, functions and variables related to the inner code will be distinguished by the superscript  $i$ , and those related to the outer code with superscript  $o$ . Assume that the inner code  $C^i$  is a convolutional code with rate  $R^i = (k/n)$ . The input–output weight-enumerating function (IOWEF) of the equivalent block code of the inner convolutional code, with the input length  $N$ , is [7]

$$A^i(L, H) = \sum_{\ell=0}^N \sum_{h=d_f^i}^{N/R^i} A_{\ell,h}^i(N) L^\ell H^h \quad (3)$$

where  $A_{\ell,h}^i(N)$  represents the number of codewords with weight  $h$  generated by information words of weight  $\ell$ , and  $L$  and  $H$  are dummy variables.

We use a uniform interleaver, which maps a codeword of weight  $\ell$  into all distinct  $\binom{N}{\ell}$  permutations with equal probability.<sup>2</sup> The outer VLC has free distance  $d_f^o$ , therefore

$$\begin{aligned} A_h(N) &= \sum_{\ell=d_f^o}^N \frac{A_\ell^o(N) A_{\ell,h}^i(N)}{\binom{N}{\ell}} \\ B_h(N) &= \sum_{\ell=d_f^o}^N \frac{B_\ell^o(N) A_{\ell,h}^i(N)}{\binom{N}{\ell}} \end{aligned} \quad (4)$$

where  $A_\ell^o(N)$  and  $B_\ell^o(N)$  are the associated multiplicities for  $C_N$ , the length- $N$  subcode of the outer code, and  $A_{\ell,h}^i(N)$  are the multiplicities of the inner code. Summing the contributions over all of the possible  $\ell$ 's gives the associated coefficients  $A_h(N)$  and  $B_h(N)$ .

This derivation was facilitated by two facts. First, for two codewords  $\mathbf{a}$  and  $\mathbf{b}$ , we have  $d_H(\mathbf{a}, \mathbf{b}) = d_H(\pi(\mathbf{a}), \pi(\mathbf{b}))$ , where  $\pi$  is the interleaving function. Second, since the inner code is a linear code, we may speak equivalently of codeword weights or codeword pair distances. In particular, for the inner code, two sequences with distance  $\ell = d_H(\pi(\mathbf{a}), \pi(\mathbf{b}))$ , will result in two codewords with coded distance  $h$ . It is due to this “invariance” of the interleaver and the linearity of the inner code that our analysis remains tractable.

We can now calculate the FER  $P_E$  and symbol-error rate (SER)  $P_S$  of the concatenated scheme as follows:

$$\begin{aligned} P_E &\leq \sum_{N=N_{\min}}^{N_{\max}} \Pr(N) \sum_{h=d_f}^{N/R^i} A_h(N) P_h \\ &= \frac{1}{2} \sum_{N=N_{\min}}^{N_{\max}} \sum_{h=d_f}^{N/R^i} \sum_{\ell \geq d_f^o} \Pr(N) \frac{A_\ell^o(N) A_{\ell,h}^i(N)}{\binom{N}{\ell}} \\ &\quad \times \operatorname{erfc}(\sqrt{hE_s/N_0}) \quad (5) \\ P_S &\leq \frac{1}{K} \sum_{N=N_{\min}}^{N_{\max}} \Pr(N) \sum_{h=d_f}^{N/R^i} B_h(N) P_h \\ &= \frac{1}{2K} \sum_{N=N_{\min}}^{N_{\max}} \sum_{h=d_f}^{N/R^i} \sum_{\ell \geq d_f^o} \Pr(N) \frac{B_\ell^o(N) A_{\ell,h}^i(N)}{\binom{N}{\ell}} \\ &\quad \times \operatorname{erfc}(\sqrt{hE_s/N_0}) \quad (6) \end{aligned}$$

where  $d_f$  is the free distance of the concatenated code. Similar to (1) and (2), the above results may be presented in terms of equivalent  $A_\ell$  and  $B_\ell$ . This alternative form is omitted here for the sake of brevity. One may also obtain bounds similar to (5) and (6) for the average interleaver size  $N_{\text{ave}}$ . We note that the above union bounds can be used with different choices of inner code, for example, a convolutional code, as in [8], or a turbo code, as in [12].

The asymptotic performance of the bounds above can be studied by looking at the behavior of coefficients  $A_h$  and  $B_h$ . We mainly present the analysis for  $A_h$ ; similar developments are possible for  $B_h$ .

Following [16], the multiplicities  $A_h$  can be modeled as a polynomial function of interleaver size, i.e.,  $A_h \approx \beta_0 N^\alpha +$

<sup>2</sup>This was first proposed by Benedetto and Montorsi, and then used in [7] to analyze concatenated codes.

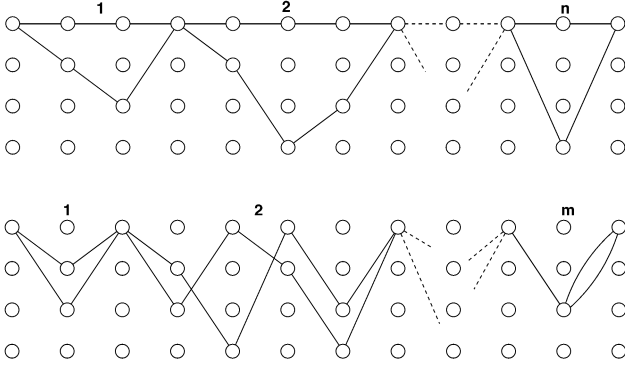


Fig. 3. Top: Concatenation of  $n$  error events with no gap in between, used in calculating the inner code multiplicity. Bottom: A pair of codewords showing concatenation of  $m$  error events with no trivial error event in between, used for calculating the VLC multiplicity.

$\beta_1 N^{(\alpha-1)} + \dots$ . We are particularly interested in the exponent  $\alpha$  of the highest order term in this polynomial, which is indicative of the asymptotic improvement of the multiplicity (and hence, code performance) with increasing interleaver length. To eliminate the dependency on  $h$ , define

$$\hat{\alpha} = \max_h \lim_{N \rightarrow \infty} \log_N A_h. \quad (7)$$

Thus,  $\hat{\alpha}$  is the dominant coefficient of  $\alpha(h)$ , where we here emphasize the dependence on  $h$ , the Hamming distance. The dominant multiplicity exponent  $\hat{\alpha}$  is referred to as *interleaver gain* in the literature [7]. The performance of the code, except in a very high SNR regime, is dependent on the multiplicities of the code, and therefore depends on  $\hat{\alpha}$ . Whenever  $\hat{\alpha} < 0$ , the dominant multiplicity gets smaller with increasing interleaver size, therefore, we will be motivated to design codes with  $\hat{\alpha} < 0$ .

We define  $\hat{h}$  as the Hamming distance of codeword pairs having the dominant multiplicity (the maximizer in the expression above). We now wish to calculate  $\hat{\alpha}$  and  $\hat{h}$ . The inner code multiplicity  $A_{\ell,h}^i$  can be expressed as

$$A_{\ell,h}^i \leq \sum_{n \geq 1} \binom{N/k}{n} A_{\ell,h,n}^i \quad (8)$$

where  $A_{\ell,h,n}^i$  are the multiplicities for codewords with input–output weights  $(\ell, h)$  having exactly  $n$  consecutive error events with no gap in between, as shown in Fig. 3 (top trellis). Equation (8) derives the overall multiplicities by inserting zero runs before some of the error events, such that the overall number of trellis sections is  $N/k$  [16].

For the outer VLC, a similar expression can be derived with certain modifications. Because of the nonlinearity of VLCs, weight enumeration has to be carried out through *all pairs of codewords*, as is shown in Fig. 3 (bottom trellis). All error events for the VLC must initiate and terminate in a “root state,” which is the state where the bit sequence for a source symbol begins or terminates (see Fig. 2). In Fig. 3, we show the root state as the top node. A pair of codewords of a VLC are illustrated in Fig. 3, using the trellis of Fig. 2. Following a similar argument as in [16], we obtain

$$A_{\ell}^o \leq \sum_{m \geq 1} \binom{N}{m} A_{\ell,m}^o \quad (9)$$

where  $A_{\ell,m}^o$  is the multiplicity of the pair of codewords at distance  $\ell$  consisting of a concatenation of exactly  $m$  simple error events, with no trivial error event in between. A trivial error event is a section of the bit trellis of the two codewords that are identical, and furthermore, it starts and ends in the root state. Equation (9) illustrates the expansion of simple codepaths with  $m$  error events, to compound codepaths that include trivial error events.

One may obtain the coefficients  $A_h$  of the concatenated code by substituting (8) and (9) into (4) to yield

$$\begin{aligned} A_h &\leq \sum_{\ell=d_f^o}^N \sum_{m \geq 1} \sum_{n \geq 1} \frac{\binom{N}{m} \binom{N/k}{n}}{\binom{N}{\ell}} A_{\ell,m}^o A_{\ell,h,n}^i \\ &\leq \sum_{\ell=d_f^o}^N \sum_{m \geq 1} \sum_{n \geq 1} \frac{\ell!}{m!n!k^n} N^{m+n-\ell} A_{\ell,m}^o A_{\ell,h,n}^i \end{aligned} \quad (10)$$

where the approximation of  $\binom{N}{x} \approx N^x/x!$  is used. Substituting (10) into (7), we obtain

$$\hat{\alpha} = \max_h (m+n) - \ell \leq \left\lfloor \frac{\ell}{d_f^o} \right\rfloor + \left\lfloor \frac{\ell}{w_{\min}} \right\rfloor - \ell \quad (11)$$

where the bound for the outer code,  $\lfloor \ell/d_f^o \rfloor$ , reflects the possibility of the concatenation of error events, all with minimum distance  $d_f^o$ , while the bound for the inner code,  $\lfloor \ell/w_{\min} \rfloor$ , shows the maximum number of concatenated error events with minimum uncoded weight  $w_{\min}$ . For block codes and nonrecursive convolutional codes,  $w_{\min} = 1$ , which results in a positive value for  $\hat{\alpha}$ , thus the concatenated code will not have any interleaving gain. For recursive convolutional codes  $w_{\min} = 2$ , since no finite error event with  $w = 1$  exists.<sup>3</sup> Evaluating the maximum of the right-hand side of (11) for the recursive convolutional code results in

$$\hat{\alpha} \leq - \left\lfloor \frac{d_f^o + 1}{2} \right\rfloor + 1 \quad (12)$$

offering interleaving gain for FER  $P_E$  whenever  $d_f^o$  is greater than two.<sup>4</sup>

To summarize, there are two important factors in the performance of source-channel concatenated codes: the free distance of the outer code and the recursive structure for the inner code. This was demonstrated by an extension of the techniques of [7] in order to accommodate the nonlinear codes of interest in source-channel coding. The design issues of the inner code are similar to the case of ordinary SCCCs, which have been well developed in the literature [7], [16].

Our analysis is based on the concept of union bound, which diverges at low  $E_b/N_0$  values (in particular, at SNR values below those corresponding to the channel cutoff rate [7]). There is no solid theoretical ground for using union-bound analysis below cutoff rate. However, as has been noted in the literature [7], design criteria based on these bounds perform surprisingly well, even at SNR values where the bounds do not converge. Our simulations also support that conclusion.

<sup>3</sup>In other words, in block codes and nonrecursive convolutional codes, a single “1” leads to a finite error event, while in recursive convolutional codes, at least two “1”s are needed in the data sequence for a finite error event.

<sup>4</sup>Similarly, calculations for  $B_h$  indicate that the interleaving gain for SER  $P_S$  is  $\hat{\alpha}_B = \hat{\alpha} - 1$ .

## V. ITERATIVE VLC AND CHANNEL DECODING

In iterative decoding, each decoder, in turn, processes the available information about the desired signal, typically log-likelihood ratios, thus modifying and hopefully improving in each iteration the pool of available information on the received signal. The additional information is called *extrinsic information* [6], [17]. Extrinsic information represents the new information obtained in each half-iteration by applying the constraint of a constituent decoder. An efficient way to calculate extrinsic information is via the soft-input soft-output (SISO) algorithm [18]. In the following, we discuss the structure of the SISO module for a channel code as well as a VLC.

### A. SISO Channel Decoder

A soft-output algorithm for channel decoding was introduced in [19]. A slightly different version of this algorithm, called the SISO module, was introduced in [18]. We give a system-level description of this block below.

The SISO module for the convolutional code, shown in Fig. 4, works on the channel code trellis. It accepts two probability streams  $\mathbf{P}(c; I)$  and  $\mathbf{P}(u; I)$  as inputs. The former is about the coded sequence  $\mathbf{c}$ , and the latter is about the information sequence  $\mathbf{u}$ . Applying the constraints provided by the channel code, additional information (extrinsic information) is obtained for both sequences,  $\mathbf{P}(c; O)$  and  $\mathbf{P}(u; O)$ , which, in turn, is passed to the other decoder. Each decoder repeats this process by using the extrinsic information that was fed back as its new input.

### B. Bit-Level SISO VLC Decoder

Many efficient channel-decoding algorithms are trellis-based. In particular, the Viterbi algorithm (VA) and SISO algorithms [18], [19] are all trellis-based. By building a trellis for a VLC, one may employ these algorithms in the decoding of VLCs.

The trellis-based algorithms for the VLC are simpler than those for the inner code for two reasons. First, for the VLC trellis, only one node (root node) has multiple incoming branches, thus the compare-select operation of the VA and selection of surviving path is done only for the root node. At other nodes, only the metric is calculated. Second, for VLC we do bit-level detection, and there is no reference to the input symbols except connections of the trellis, which simplifies the SISO module.

Based on the trellis representation of a VLC introduced in Section III, we derive a SISO algorithm for VLCs. Following the notation of [18], the extrinsic information is calculated as follows. At time  $k$ , the output probability distribution is evaluated as

$$\tilde{P}_k(u; O) = \tilde{h} \sum_{e:u(e)=u} A_{k-1}(s^S(e))B_k(s^E(e))P_k(u; I) \quad (13)$$

where  $e$  represents a branch of the trellis,  $u(e)$ ,  $s^S(e)$ , and  $s^E(e)$  are, respectively, the branch value, the starting state, and the ending state of the branch  $e$ , and the constant  $\tilde{h}$  is a normalizing factor to ensure  $\tilde{P}_k(0; O) + \tilde{P}_k(1; O) = 1$ . The quantities

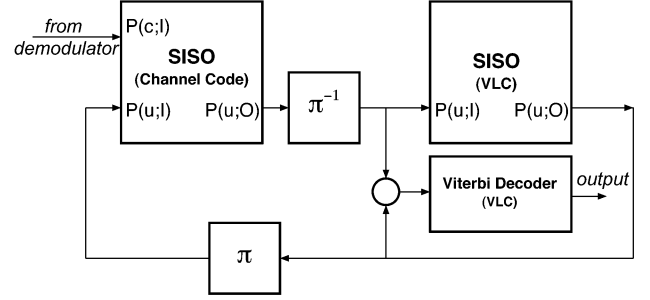


Fig. 4. Iterative VLC and convolutional decoding.

$A_k(\cdot)$  and  $B_k(\cdot)$  are calculated through forward and backward recursions, respectively, as follows:

$$A_k(s) = \sum_{e:s^E(e)=s} A_{k-1}(s^S(e))P_k(u; I)$$

$$B_k(s) = \sum_{e:s^S(e)=s} B_{k+1}(s^E(e))P_{k+1}(u; I)$$

with initial values  $A_0(s) = B_N(s) = 1$  for the root state (since the trellis always starts and ends at the root state), and  $A_0(s) = B_N(s) = 0$  for all other states. In order to exclude the input information  $P_k(u; I)$  from the output probability and obtain the so-called extrinsic information, both sides of (13) are divided by  $P_k(u; I)$ , i.e.,  $P_k(u; O) = \tilde{P}_k(u; O)/P_k(u; I)$ .

Therefore,  $P_k(u; I)$  (input probability) and  $P_k(u; O)$  (extrinsic information) together form the *a posteriori* probability (APP) of the input sequence. In practice, the additive (logarithmic) version of a SISO algorithm is employed to avoid multiplications and prevent numerical problems.

### C. Iterative Decoding and Density Evolution

An iterative decoder is shown in Fig. 4, using the SISO blocks already introduced. Blocks denoted  $\pi$  and  $\pi^{-1}$  are the interleaver and deinterleaver, respectively.

In each iteration, only the extrinsic information generated by each SISO block,  $P_{CC}(u; O)$  and  $P_{VLC}(u; O)$ , are exchanged between the soft-output decoders. After the final iteration, the log-scale soft-sequence,  $P_{VLC}(u; I) + P_{VLC}(u; O)$ , is decoded at symbol level by the Viterbi decoder over the same bit-level trellis.

The convergence of the iterative decoder can be illustrated by the density-evolution method [20], or alternatively, by extrinsic information transfer (EXIT) charts [21]. We use the density-evolution method for our analysis. We have verified the corresponding Gaussian approximation assumption in our setting. Density evolution treats the decoder as a nonlinear dynamical feedback system where the nonlinear input-output functions are calculated empirically. For further information on density evolution, we refer the reader to [20].

## VI. EXPERIMENTAL RESULTS

Table I shows the 5-ary source used in our experiments and various codes designed for this source.  $C_1$  is a Huffman code,  $C_2$  is an RVLC for this source reported in [8], and the codes  $C_3$ ,  $C_4$ , and  $C_5$  were designed by us with successively higher redundancies. The redundancy of the VLCs can be quantified by

TABLE I  
 VARIABLE-LENGTH CODES USED IN SECTION VI

$s$	$P_S(s)$	$C_1$	$C_2$ [8]	$C_3$	$C_4$	$C_5$
0	0.33	00	00	11	011	0100
1	0.30	11	11	001	1100	10011
2	0.18	10	010	0100	10111	101001
3	0.10	010	101	0101100	00000	011110
4	0.09	011	0110	0001010	000101	0000111
$E[L]$	H=2.14	2.19	2.46	3.61	4.13	5.13
$d_{free}$		1	2	3	4	5

comparing their average length with that of the Huffman code. The Levenshtein distance [13], [2] is used in reporting SERs.

It is noteworthy that despite the differences, the trellises of the different codes have roughly the same order of complexity. This arises due to the sparseness of the VLC trellises, which becomes more pronounced when the code has redundancy. For a more meaningful comparison of complexity, one may construct more compact trellises with minimal number of states. For example,  $C_3$  has a sparse trellis with 13 states and only 17 single-bit branches, which is equivalent to a four-state compact trellis with eight branches, the same as the equivalent trellis of code  $C_2$ . Thus, the complexities are comparable.

Our inner convolutional codes are four-state codes with rates  $1/2$  or  $2/3$  taken from [7]. The rate- $1/2$  codes are the recursive  $CC_1$  and nonrecursive  $CC_2$  with the following generator polynomials:  $G_{CC_1}(D) = (1, (1 + D^2)/(1 + D + D^2))$  and  $G_{CC_2}(D) = (1 + D^2, 1 + D + D^2)$ . The rate- $2/3$  codes,  $CC_3$  and  $CC_4$ , with the generator polynomials as shown below are considered

$$G_{CC_3}(D) = \begin{pmatrix} 1 & 0 & \frac{1 + D^2}{1 + D + D^2} \\ 0 & 1 & \frac{1 + D}{1 + D + D^2} \end{pmatrix}$$

$$G_{CC_4}(D) = \begin{pmatrix} 1 + D & D & 1 \\ 1 + D & 1 & 1 + D \end{pmatrix}.$$

In our experiments, a packet of  $K$  symbols is entropy encoded, interleaved, channel encoded, and transmitted using binary phase-shift keying (BPSK) modulation over an AWGN channel.

Our first experiment is designed to test the accuracy of our analysis. Fig. 5 shows union bounds and simulation results for the concatenated code  $C_2 + CC_1$ . There are several factors to consider when reading this plot. First, union bounds work in the high  $E_b/N_0$  regions, and the cutoff rate for this code is associated with  $E_b/N_0 = 2.45$  dB. Second, union bounds are calculated for the optimal (ML) decoder, while the simulations, by necessity, use iterative decoding. Finally, calculation of the multiplicities for a nonlinear, VLC is a lengthy and time-consuming process, thus we present “truncated bounds” calculated with the first ten terms of the multiplicities of the outer code that were available in [8]. The decoding experiment was performed with ten iterations, with packet lengths of 20 and 200. The bounds are in agreement with simulations.

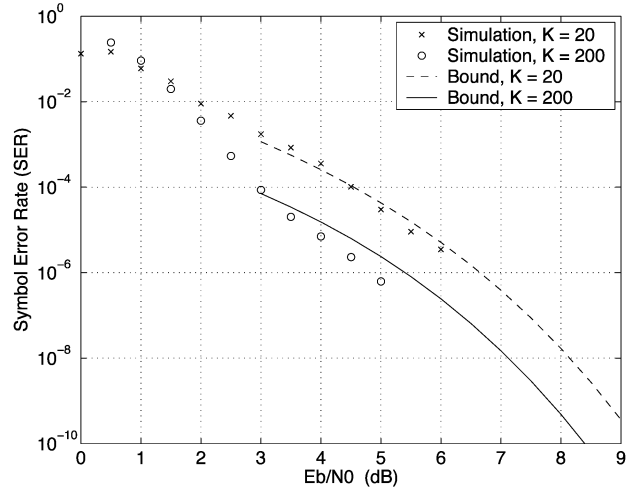


Fig. 5. SER of  $C_2 + CC_1$ ,  $K = 20$  and 200 symbols.

As mentioned in Section IV, higher redundancy in the outer VLC leads to higher interleaver gain. To demonstrate this effect, we simulated the VLCs  $C_4$  and  $C_2$ , each concatenated with the inner code  $CC_1$ . To maintain (approximately) the same overall rate, in the experiment with  $C_4$ , the inner code is punctured to rate- $8/9$ , hence the notation  $C_4 + CC_1(8/9)$ . The overall rates of the two experiments are 0.445 for  $C_2 + CC_1$ , and 0.471 for  $C_4 + CC_1(8/9)$ , where the equivalent “code rate” of a VLC is calculated by dividing the average length of the code by the average length of the Huffman code. The SERs and FERs are shown in Fig. 6 for  $K = 2000$  symbols. For small values of  $E_b/N_0$ , the code  $C_2 + CC_1$  outperforms  $C_4 + CC_1(8/9)$  due to the more powerful inner code. However, the latter starts outperforming the former for  $E_b/N_0 > 1.5$  dB. The sharper drop in error rate of the latter code is noteworthy.<sup>5</sup>

The corresponding density evolutions of the iterative decoders are shown in Fig. 7. We use the approximate Gaussian density evolution of [20]. We show density evolutions at  $E_b/N_0 = 1.5$  dB (the staircase plot). The corresponding iterative decoding thresholds are demonstrated in Fig. 7 as well.

For further comparisons, we used the code  $C_3$  with free distance  $d_f^o = 3$ , concatenated with the inner code  $CC_3$ , a rate- $2/3$  recursive convolutional code. The concatenated code has an overall rate of 0.404. The SER and FER of the this code are shown in Fig. 6. The code  $C_3 + CC_3$  outperforms  $C_2 + CC_1$  in the entire range of  $E_b/N_0$  after the second iteration of the decoder.

As mentioned previously, Bauer and Hagenauer [8] demonstrated coding gain via iterative source-channel decoding. However, in their case, the baseline system did not have the advantage of iterative decoding. We pose a slightly different question. Assuming we have a fixed computational and rate budget, we would like to compare the source-channel iterative decoder with a separable decoder whose channel decoder is iterative. For experimental verification of this and similar questions, we introduce three SCCC:  $SCCC_1 : CC_2 + CC_1(8/9)$  and  $SCCC_2 : CC_4 + CC_3$ , both with overall rate- $4/9$ , and  $SCCC_3 : CC_2 + CC_3$  with overall rate- $1/3$ .

<sup>5</sup>The BPSK-constrained Shannon limit for these codes is approximately 0 dB.

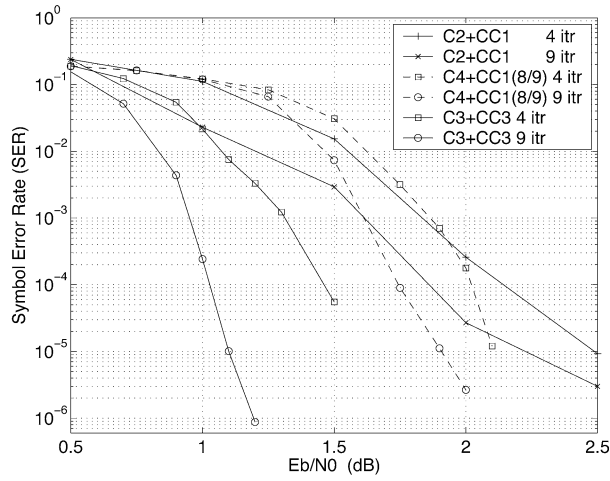


Fig. 6. Performance of  $C_2 + CC_1$ ,  $C_4 + CC_1$  (punctured to rate-8/9), and  $C_3 + CC_3$ .  $K = 2000$  symbols.

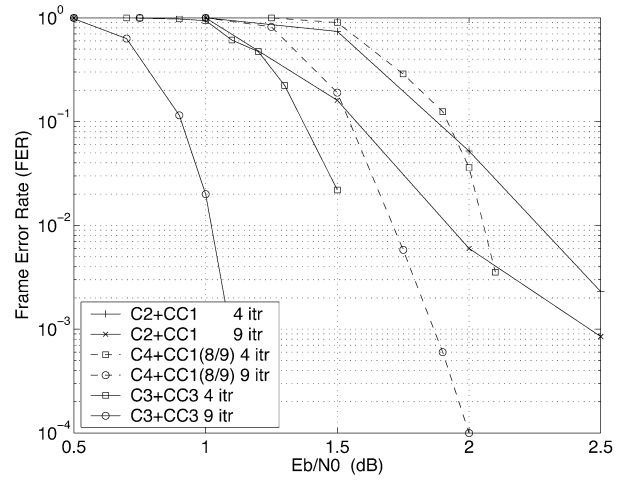


Fig. 7. Approximate Gaussian density evolution of  $C_2 + CC_1$  and  $C_4 + CC_1$  (punctured to rate-8/9),  $K = 2000$  symbols. An instance of the convergence of the decoders at  $E_b/N_0 = 1.5$  dB is shown.

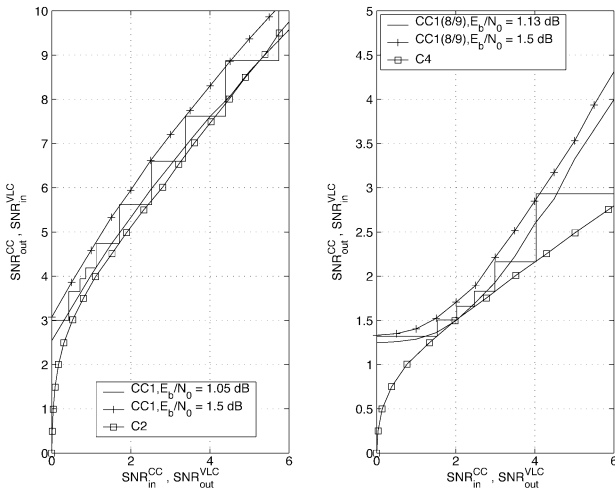


Fig. 8. Comparison of concatenated redundant VLC and convolutional codes versus concatenated Huffman code and SCCCs,  $K = 2000$  symbols.

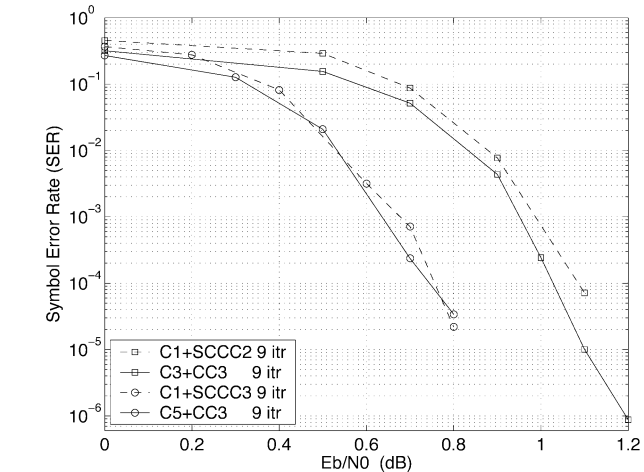


Fig. 9. Comparison of concatenated redundant VLC and convolutional codes versus concatenated Huffman code and SCCCs,  $K = 2000$  symbols.

We compare the iterative source-channel decoding of  $C_2 + CC_1$  with a system consisting of the Huffman code  $C_1$  concate-

nated with  $SCCC_2$ . The two systems have the same overall rate and decoder complexity. The simulation results are not shown in a separate figure, but one can compare the results of  $C_2 + CC_1$  in Fig. 6 with the results of  $C_1 + SCCC_2$  in Fig. 8 (both for nine iterations). The comparison indicates that the case of separable source-channel decoding is superior to joint source-channel decoding. We believe this is largely due to the small  $d_{free}$  of  $C_2$ , which is the RVLC used by Bauer and Hagenauer [8]. Therefore, it seems that separable decoding (with an iterative channel decoder) can be superior to iterative source-channel decoding when the outer code has small free distance.

Then one may ask, how does a joint source-channel decoder compare with a separable decoder if we increase the free distance of the outer source code? We designed several experiments to address this question. In a comparison of the separable code  $C_1 + SCCC_1$  with the joint source-channel decoding of  $C_4 + CC_1(8/9)$ , we found that especially at higher  $E_b/N_0$ , the joint source-channel decoding works much better, while at intermediate  $E_b/N_0$ , the two methods perform the same. We conducted two more experiments, whose results are shown in Fig. 8. The separable code  $C_1 + SCCC_2$ , is compared against  $C_3 + CC_3$  (both with rates  $\sim 4/9$ ), where  $C_3 + CC_3$  outperforms  $C_1 + SCCC_2$  slightly. In the same Fig. 8,  $C_1 + SCCC_3$  is compared against  $C_5 + CC_3$  (both with rate  $\sim 1/3$ ), and they perform roughly similarly.

Thus, the simulations did not point to a clear and universal advantage for either the joint or separable approach. In some cases, where the outer entropy code has low redundancy, the separable case is clearly better, while in other cases, either the joint or the separable solution might be superior. The design choices must be made on a case-by-case basis.

## VII. CONCLUSION

We obtain union bounds for the SERs and FERs of concatenated source-channel codes for finite-alphabet sources. We generalize the previous notions of outer entropy code by inserting an unrestricted redundant VLC; thus, our analysis is general with respect to the choice of the outer code, including nonredundant (Huffman) codes, RVLC codes, and the so-called VLECCs. We

use techniques originally developed for the SCCCs and adapt them so that they can be used with the nonlinear outer codes that are of interest in source-channel coding. By evaluating the union bounds of the concatenated scheme, we further studied the role of the constituent codes, and illustrated through simulations the relevance of the suggested design rules.

#### REFERENCES

- [1] Y. Takishima, M. Wada, and H. Murakami, "Reversible variable length codes," *IEEE Trans. Commun.*, vol. 43, pp. 158–162, Feb.-Apr. 1995.
- [2] V. Buttigieg, "Variable-length error-correcting codes," Ph.D. dissertation, Dept. Elec. Eng., Univ. Manchester, Manchester, U.K., 1995.
- [3] V. Buttigieg and P. G. Farrell, "Variable-length error-correcting codes," *Proc. IEE Commun.*, vol. 147, no. 4, pp. 211–215, Aug. 2000.
- [4] K. P. Subbalakshmi and J. Vaisey, "On the joint source-channel decoding of variable-length encoded sources: The BSC case," *IEEE Trans. Commun.*, vol. 49, pp. 2052–2055, Dec. 2001.
- [5] —, "Joint source-channel decoding of entropy coded Markov sources over binary symmetric channels," in *Proc. IEEE Int. Conf. Communications*, 1999, pp. 446–450.
- [6] C. Berrou and A. Glavieux, "Near-optimum error correcting coding and decoding: Turbo codes," *IEEE Trans. Commun.*, vol. 44, pp. 1261–1271, Oct. 1996.
- [7] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design, and iterative decoding," *IEEE Trans. Inform. Theory*, vol. 44, pp. 909–926, May 1998.
- [8] R. Bauer and J. Hagenauer, "On variable length codes for iterative source-channel decoding," in *Proc. Data Compression Conf.*, Apr. 2001, pp. 273–282.
- [9] —, "Iterative source-channel decoding using reversible variable-length codes," in *Proc. Data Compression Conf.*, Apr. 2000, pp. 93–102.
- [10] N. Görtz, "On the iterative approximation of optimal joint source-channel decoding," *J. Select. Areas Commun.*, vol. 19, pp. 1662–1670, Sept. 2001.
- [11] A. Guyader, E. Fabre, C. Guillemot, and M. Robert, "Joint source-channel turbo decoding of entropy-coded sources," *J. Select. Areas Commun.*, vol. 19, pp. 1680–1696, Sept. 2001.
- [12] K. Lakovic and J. Villasenor, "Combining variable-length codes and turbo codes," in *Proc. IEEE Vehicular Technology Conf.*, Spring 2002, pp. 1719–1723.
- [13] T. Okuda, E. Tanaka, and T. Kasai, "A method for correction of grabbed words based on the Levenshtein metric," *IEEE Trans. Comput.*, vol. C-25, pp. 172–176, Feb. 1976.
- [14] V. B. Balakirsky, "Joint source-channel coding with variable length codes," in *Proc. IEEE Int. Symp. Information Theory*, June 1997, p. 419.
- [15] A. Murad and T. Fuja, "Robust transmission of variable-length encoded sources," in *Proc. IEEE Wireless Communications, Networking Conf.*, vol. 2, Sept. 1999, pp. 968–972.
- [16] D. Divsalar and R. J. McEliece, "On the design of generalized coding systems with interleavers," Jet Propulsion Lab. Prog. Rep., Aug. 1998.
- [17] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 429–445, Mar. 1996.
- [18] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "A soft-input soft-output APP module for iterative decoding of concatenated codes," *IEEE Commun. Lett.*, vol. 1, pp. 22–24, Jan. 1997.
- [19] L. R. 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, Mar. 1974.
- [20] D. Divsalar, S. Dolinar, and F. Pollara, "Iterative turbo decoder analysis based on density evolution," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 891–907, May 2001.
- [21] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, pp. 1727–1737, Oct. 2001.



**Ahmadreza Hedayat** (S'00) received the B.S.E.E. and M.S.E.E. degrees from the University of Tehran, Tehran, Iran, in 1994 and 1997, respectively, and the Ph.D. degree in electrical engineering from the University of Texas at Dallas, Richardson, in 2004.

From 1995 to 1999, he was with Pars Telephone Kar and Informatic Services Corporation, Tehran, Iran. His current research interests include MIMO signaling and techniques, channel coding, and source-channel schemes.



**Aria Nosratinia** (M'97–SM'04) received the B.S. degree in electrical engineering from the University of Tehran, Tehran, Iran, in 1988, the M.S. degree in electrical engineering from the University of Windsor, Windsor, ON, Canada, in 1991, and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign in 1996.

From 1995 to 1996, he was with Princeton University, Princeton, NJ. From 1996 to 1999, he was a Visiting Professor and Faculty Fellow at Rice University, Houston, TX. Since 1999, he has been with the University of Texas at Dallas, Richardson, where he is currently an Associate Professor of Electrical Engineering. His research interests are in the broad area of communication and information theory, particularly coding and signal processing for the communication of multimedia signals.

Dr. Nosratinia is currently an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING. He was the recipient of the National Science Foundation Career award in 2000 and has twice received chapter awards for outstanding service to the IEEE Signal Processing Society.