CRC-Assisted Error Correction in a Convolutionally Coded System

Renqiu Wang, *Member, IEEE*, Wanlun Zhao, *Member, IEEE*, and Georgios B. Giannakis, *Fellow, IEEE*

***Abstract*—In communication systems employing a serially concatenated cyclic redundancy check (CRC) code along with a convolutional code (CC), erroneous packets after CC decoding are usually discarded. The list Viterbi algorithm (LVA) and the iterative Viterbi algorithm (IVA) are two existing approaches capable of recovering erroneously decoded packets. We here employ a soft decoding algorithm for CC decoding, and introduce several schemes to identify error patterns using the posterior information from the CC soft decoding module. The resultant iterative decoding-detecting (IDD) algorithm improves error performance by iteratively updating the *extrinsic* information based on the CRC parity check matrix. Assuming errors only happen in unreliable bits characterized by small absolute values of the log-likelihood ratio (LLR), we also develop a partial IDD (P-IDD) alternative which exhibits comparable performance to IDD by updating only a subset of unreliable bits. We further derive a soft-decision syndrome decoding (SDSD) algorithm, which identifies error patterns from a set of binary linear equations derived from CRC syndrome equations. Being non-iterative, SDSD is able to estimate error patterns directly from the decoder output. The packet error rate (PER) performance of SDSD is analyzed following the union bound approach on pairwise errors. Simulations indicate that both IDD and IVA are better tailored for single parity check (PC) codes than for CRC codes. SDSD outperforms both IDD and LVA with weak CC and strong CRC. Applicable to AWGN and flat fading channels, our algorithms can also be extended to turbo coded systems.**

***Index Terms*—Convolutional codes, CRC, error correction, list Viterbi decoding, iterative Viterbi decoding.**

I. Introduction

TRADITIONALLY, error detection and correction are performed independently. As the most popular error detection code (EDC), a cyclic redundancy check (CRC) code is usually appended to a data packet, which is subsequently encoded with an error correction code (ECC); e.g., a convolutional code (CC). Correspondingly, ECC decoding is performed first at the receiver, and CRC is applied afterwards. If CRC detects errors, then the entire packet is discarded. When the signal to noise ratio (SNR) is relatively high, only a small number of errors are typically present in an erroneously decoded packet. Instead of discarding the entire packet, the theme is this paper is to possibly recover it by utilizing jointly ECC and CRC.

We will study a system with serially concatenated CRC and CC (CRC-CC). Being a special case of conventional serially concatenated codes (CSCC) [1], CRC-CC has been widely applied in wireless communications, for example, in an IS-95 system. Various approaches are available to recover erroneous packets following the Viterbi decoding stage. One of them is based on the list Viterbi algorithm (LVA), which produces a list of L most promising sequence estimates after a trellis search [2]. CRC is used to check if the correct codeword is in the list, that is, if any of L sequences passes CRC. LVA has been extended to list-SOVA using the LLR of a soft output VA (SOVA) [3]. Both LVA and list-SOVA improve error performance by increasing the list size L, but they only rely on the trellis structure to recover the codeword, since they employ CRC only for error detection.

Associated with each CRC generator polynomial, is an equivalent parity check matrix $\mathbf{H}$. The overall parity check matrix $\mathbf{H}_a$ for CC codewords can be found in [4] and [5]. If CRC detects errors after Viterbi decoding, the iterative VA (IVA) updates the weights of each convolutionally coded bit with a randomly selected parity check node associated to this bit defined by $\mathbf{H}_a$. The updated weights lead to a new iteration of Viterbi decoding. The iterative process is terminated right after finding a packet that passes CRC. The idea behind the iterative decoding-detecting (IDD) algorithm we put forth in this paper, is to employ a soft decoding algorithm for decoding the CC, and directly use $\mathbf{H}$ to iteratively update soft information of the CRC packet. Assuming errors only happen to unreliable bits characterized by small absolute values of the corresponding log-likelihood ratio (LLR), we also derive a partial IDD (P-IDD) algorithm by only updating soft information for a subset of unreliable bits.

IVA, IDD, and P-IDD algorithms improve performance of the CRC-CC system by exchanging soft information between CRC and CC modules. IVA/IDD based algorithms prefer simple parity check (PC) codes, but are not as effective for standard CRC. For this reason, we also derive an algorithm where CRC is invoked to aid error correction without iteratively decoding the CC. Supposing again that errors happen only to unreliable bits, we transform the set of highly underdetermined CRC syndrome equations to a smaller set of binary linear equations after cancelling reliable bits. Based on this smaller set of equations, we estimate the error patterns.

Paper approved by A. Banihashemi, the Editor for Coding and Communication Theory of the IEEE Communications Society. Manuscript received April 14, 2005; revised September 22, 2006. This work was supported through collaborative participation in the Communications and Networks Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon. This paper was presented in part at IEEE International Conference on Communications, Istanbul, Turkey, June 11-15, 2006.

R. Wang and W. Zhao are with Qualcomm Inc., San Diego, CA 92121, USA (e-mail: {rwang, wzhao}@qualcomm.com).

G. B. Giannakis is with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455, USA (e-mail: georgios@umn.edu).

Digital Object Identifier 10.1109/TCOMM.2008.060543

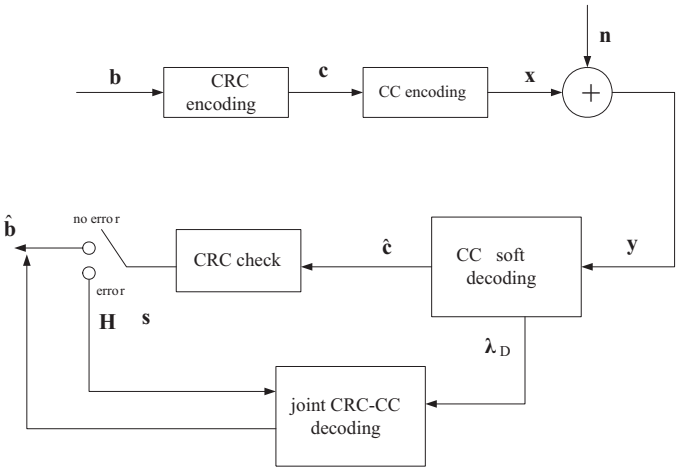


Fig. 1. System model.

Simulations indicate that in both AWGN and flat fading channels, this soft decision syndrome decoding (SDSD) algorithm outperforms LVA and IDD when weak CC and strong CRC are used. But IDD works better than SDSD for strong CC and simple parity check (PC) codes.

Our proposed algorithms are distinct from LVA when it comes to handling error events. LVA focuses on the trellis structure, whereas our approach capitalizes on the CRC structure and employs CRC for error correction. An attractive feature of our algorithms is that they require only soft information from the ECC decoding module, and can be easily extended to other ECC systems. An example using turbo codes [6], [7] will be provided in the simulations Section V. The rest of the paper is structured as follows: In Section II we describe the CRC-CC system model. In Sections III and IV we develop our IDD/P-IDD and the SDSD algorithms, respectively. Simulation results are presented for different CRC and CC choices in Section V. Finally, we conclude this paper in Section VI.

II. System Model

Consider the concatenated system model depicted in Figure 1. We use bold lower case fonts and the argument (x) to represent a column vector and its corresponding polynomial form. For notation simplicity, indices of vector entries will start from 0. If $\mathbf{b}$ denotes a $k \times 1$ information vector with entries $b_i \in \{0, 1\}$, then $b(x)$ represents the binary information polynomial. The degree m CRC generator polynomial is given as $g(x) = x^m + \sum_{i=1}^{m-1} g_i x^i + 1$. Let $\mathrm{rem}(a, g)$ denote the remainder polynomial of $a(x)$ divided by $g(x)$, and let $r(x) = \mathrm{rem}(c, g)$. The CRC codeword polynomial is given as

$$c(x) = x^m b(x) + r(x). \tag{1}$$

With $n = k + m$, the $n \times 1$ vector $\mathbf{c} = [\mathbf{r}^T \ \mathbf{b}^T]^T$ is the CRC packet, where ^{T} stands for the transposition operation, and $\mathbf{r} := [r_0 \ r_1 \ \ldots \ r_{m-1}]^T$ is the parity check vector. The $n \times 1$ vector $\mathbf{c}$ is encoded using a CC with memory v. For simplicity, we use a terminated CC, i.e., v zero bits are padded after $\mathbf{c}$ to set the state back to 0 at the end of the encoding process. For a terminated CC with rate $1/2$, we have $R = n/(n + v)/2$. The n/R bits $\mathbf{x}$ are then converted to binary channel symbols $(+1$ or $-1)$, which are transmitted over a single-input single-output channel. The received vector $\mathbf{y} = \mathbf{x} + \mathbf{n}$, where $\mathbf{n}$ is the white Gaussian noise.

We use a soft decoder to decode the CC at the receiver, which computes the LLR of *a posteriori* probabilities per bit c_j; namely

$$\lambda_{D,c_j|\mathbf{y}} := \ln \frac{P(x_j = +1|\mathbf{y})}{P(x_j = -1|\mathbf{y})},$$

which constitutes the posterior information about c_j.

The hard decision of c_j is

$$\hat{c}_j = \begin{cases} 1 & \text{if } \lambda_{D,c_j|\mathbf{y}} > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

If $\hat{\mathbf{c}} := [\hat{c}_1 \ \ldots \ \hat{c}_n]^T$ denotes the hard decision vector of $\mathbf{c}$, and $\mathbf{e}$ the error pattern vector, then $\hat{\mathbf{c}} = \mathbf{c} + \mathbf{e}$. Clearly when an entry of $\mathbf{e}$ is 1, the corresponding bit in $\mathbf{c}$ is wrong.

Define $h_i(x) = \mathrm{rem}(x^i, g), \quad i = 0, \ldots, n-1$. Let $\mathbf{h}_i$ be the column vector associated with $h_i(x)$, and $\mathbf{H} := [\mathbf{h}_0 \ \ldots \ \mathbf{h}_{n-1}]$ the $m \times n$ equivalent CRC parity matrix. To test whether $\hat{\mathbf{c}}$ is valid, is equivalent to testing the syndrome $\mathbf{s} = \mathbf{H}\hat{\mathbf{c}}$, which is also expressed as

$$\mathbf{s} = \mathbf{H}\mathbf{e}. \tag{3}$$

Let $\gamma_j = |\lambda_{D,c_j|\mathbf{y}}|$ denote the absolute value of $\lambda_{D,c_j|\mathbf{y}}$, and $\boldsymbol{\gamma} = [\gamma_1, \gamma_2, \ldots, \gamma_n]^T$. The probability that $\hat{c}_j$ is correct is therefore

$$P(\hat{c}_j = c_j) = P(e_j = 0) = \frac{e^{\gamma_j}}{1 + e^{\gamma_j}} = \frac{1}{2}[1 + \tanh(\boldsymbol{\gamma}_j)], \tag{4}$$

which is a monotonically increasing function of γ_j and always exceeds 0.5. Thus, γ_j indicates the reliability of the hard decision $\hat{c}_j$. If CRC detects errors, a joint CRC-CC decoding algorithm is used to recover the message based on soft information γ_j as we will see in the ensuing sections.

Remark: There exist several optimal or near-optimal soft decoding algorithms for CC. The maximum *a posteriori* probability (MAP) decoding minimizes the bit error rate (BER). The BCJR algorithm implements MAP decoding of a CC in the probability domain [6], [8], and amounts to the log-MAP algorithm when working in the log-domain [9], [10]. Based on suitable approximations, max-log-MAP and SOVA offer two near-optimal soft decoding alternatives trading off error performance for reduced complexity [9], [11], [12]. Interestingly, it has been found recently that improved SOVA outperforms MAP in an iterative setting [12], [13]. The joint CRC-CC algorithms presented in this paper exploit the CRC structure to recover packets and are not confined to a specific soft CC decoder.

III. Iterative Decoding-Detecting Algorithms

If $\hat{\mathbf{c}} \neq \mathbf{c}$, errors are likely to occur with higher probability in unreliable bits with a small γ_j than to bits with larger γ_j. A naive approach is thus to flip the unreliable bits starting from the most unreliable one to the most reliable one, until a codeword passing the CRC is found. Such a bit-flipping idea has been used in various algorithms; e.g., in the Chase algorithm for decoding binary block codes [14].

The problem with bit-flipping is that decoding complexity grows exponentially. LVA based algorithms generate a list of candidates according to the probability of corresponding error events [2]. With increased complexity, LVA based algorithms improve the error performance of VA by increasing the list size. A post-processor for providing reliability information about every decoded data symbol at the output of an arbitrary decoder is proposed in [15]. These algorithms focus only on the structure of error correction codes and do not account for the structure of the CRC. Although primarily used for error detection, CRC can also provide additional information for finding error patterns,

The Tanner graph is a useful tool to graphically represent parity check codes [16]. If the parity check matrix has low density, it has been shown that the sum-product and min-sum algorithms are near-optimal [17]–[19]. However, error floors occur if there are many small cycles in the Tanner graph, as for CRC. This explains why a straightforward approach of employing sum-product or min-sum algorithms to decode CRC-CC will not lead to near optimal performance.

IVA provides a mechanism for dealing with small loops in the Tanner graph of a CRC code. A related idea is used in developing our iterative decoding-detecting (IDD) algorithm. For each bit c_i, we randomly choose a connected parity check code j defined by the CRC parity check matrix $\mathbf{H}$, and also a variable node c_l ($l \neq i$) connected to check node j. Let $\hat{cs} = \mathbf{h}_j^T \hat{\mathbf{c}} + \hat{c}_i + \hat{c}_l$ be the checksum of all variable nodes connected to parity check nodes j excluding c_i and c_j, where, $\mathbf{h}_j^T$ denoting the $j$the row of $\mathbf{H}$, the extrinsic information about c_i is found as

$$\lambda_{E,c_i}^{CRC} = \alpha(-1)^{\hat{cs}} \lambda_{D,c_l|\mathbf{y}}, \tag{5}$$

where $\alpha \in (0,1)$ is used to control error propagation and is usually set to 0.25. We can now use the extrinsic information about $\mathbf{c}$ as *a priori* information for the next iteration of the soft decoder. The iterative process continues until a valid CRC packet is found, or, until I_t, the number of iterations, reaches a threshold.

The major computational burden of the IDD process comes from the soft CC decoder, which is in general of order $O(2^v n)$. The additional computation of $\hat{cs}$ and $\lambda_{E,c_i}{}^{CRC}$ is $O(n)$, which is negligible compared to the CC decoder with long memory. Thus, an approximate IDD decoding complexity is $O(2^v n I_t)$ for a rate-1/2 convolutional code. Compared to IVA with decoding complexity $O(2^v n I_t)$ [5] and parallel LVA with decoding complexity $O(2^v n L)$ [2], IDD exhibits comparable complexity order but might require more computation due to soft decoding. The circuit of the soft decoder, if SOVA decoding is used, can be made less than twice that of conventional VA [20].

Depending on $\boldsymbol{\gamma}$, some bits are more reliable than others. Let us divide $\{c_i\}_{i=0}^{n-1}$ into two sets: $\mathcal{C}_u$ the set of unreliable bits containing the bits with l lowest γ values, and $\mathcal{C}_r$ the set of reliable bits containing the rest. In every iteration after the CC soft decoding, P-IDD calculates the extrinsic information about the unreliable bits in $\mathcal{C}_u$ according to (5). Finding the l unreliable bits out of n bits incurs worst case complexity of $O(nl + n + l - l^2)$ with the Z algorithm [21], [22]. In the next iteration, the corresponding soft information of unreliable bits and their neighbors within the constraint length are updated by the soft CC decoder, which incurs worst case complexity $2^v lv$. The worst case complexity of P-IDD is, therefore, $O(2^v n + nl + n + l - l^2 + 2^v lv(I_t - 1))$. At high SNR when $lv \ll n$ and $l \ll 2^v$, the computation is dominated by the first iteration of conventional soft CC decoder and P-IDD requires less computations than IDD. P-IDD trades off performance for complexity as will be verified in Section V.

It has been shown that IVA improves performance more when $g(x)$ contains a smaller number of 1's [4]. That is, IVA prefers simple parity check (PC) codes with $g(x) = x^m + 1$ rather than strong CRC codes. In Section V, our simulations will indicate that IDD and P-IDD exhibit similar behavior. In the ensuing section, we will propose another algorithm that works better for standard CRC codes.

IV. SDSD Algorithm

A. Estimating error patterns from CRC syndrome equations

Different from the iterative decoding pursued by IVA/IDD based algorithms, we aim here to identify the error pattern $\mathbf{e}$ directly after CC decoding. Let us turn our attention to (3). For practical CRC codes, we have $k \gg m$, which means that $\mathbf{H} = [\mathbf{I}_m \ \mathbf{P}]$ is a fat matrix, and (3) is highly underdetermined. There will be multiple solutions to (3). Let us expand (3) to

$$\begin{bmatrix} \mathbf{I}_m & \mathbf{P} \end{bmatrix} \begin{bmatrix} \mathbf{e}_I \\ \mathbf{e}_P \end{bmatrix} = \mathbf{s}, \tag{6}$$

where the $m \times 1$ binary vector $\mathbf{e}_I$ and the $k \times 1$ binary vector $\mathbf{e}_P$ are the two parts of the error pattern corresponding to the parity check and information bits, respectively. Given a nonzero $\mathbf{s}$ with a certain $\mathbf{e}_P$, there will be a unique $\mathbf{e}_I$ to satisfy (6): $\mathbf{e}_I = \mathbf{s} + \mathbf{P}\mathbf{e}_P$. Since there are totally 2^k possible $\mathbf{e}_P$ vectors, there are 2^k solutions of the form: $\mathbf{e} = [\mathbf{e}_I^T \ \mathbf{e}_P^T]^T$. If $P(\mathbf{e})$ denotes the probability of a certain error event $\mathbf{e}$, the ML estimate of $\mathbf{e}$ is

$$\hat{\mathbf{e}}_{ml} = \arg\max_{\mathbf{He}=\mathbf{s}} \ P(\mathbf{e}). \tag{7}$$

Finding $P(\mathbf{e})$ requires the joint probability density function (PDF) of $\{e_i\}_{i=0}^{n-1}$, which is difficult to quantify due to the correlation introduced by CC. For this reason, we resort to a sub-optimal approach whereby errors in $\mathbf{c}$ are assumed independent. This assumption becomes increasingly reasonable as the SNR increases because errors tend to be singular as SNR increases. Under this assumption, the probability for a certain error event $\mathbf{e}$ to happen is

$$P(\mathbf{e}) = \prod_{i=0}^{n-1} P(e_i) = \prod_{i \in I_0} P(e_i = 0) \prod_{i \in I_1} P(e_i = 1), \tag{8}$$

where I_0 and I_1 are the set of indices i for which $e_i = 0$ and $e_i = 1$, respectively. According to (4), $P(e_i = 0) = e^{\gamma_i}/(1 + e^{\gamma_i})$, and $P(e_i = 1) = 1/(1 + e^{\gamma_i})$, for which (7) becomes

$$P(\mathbf{e}) = \frac{\prod_{i \in I_0} e^{\gamma_i}}{\prod_{i=0}^{n-1} 1 + e^{\gamma_i}} \ ;$$

or equivalently in the log-domain,

$$\ln[P(\mathbf{e})] = \sum_{i \in I_0} \gamma_i - \sum_{i=0}^{n-1} \ln(1 + e^{\gamma_i}). \tag{9}$$

After eliminating the common term $\sum_{i=0}^{n-1}\ln(1+e^{\gamma_i})$ from all error patterns, and subtracting $(1/2)\sum_{i=0}^{n-1}\gamma_i$ from both sides, (7) yields

$$\begin{aligned}\hat{\mathbf{e}} &= \arg\max_{\mathbf{He}=\mathbf{s}} \ (1/2)\sum_{i\in I_0}\gamma_i - (1/2)\sum_{i\in I_1}\gamma_i \\ &= \arg\max_{\mathbf{He}=\mathbf{s}} \ (\mathbf{1}-2\mathbf{e})^T\boldsymbol{\gamma}, \end{aligned} \tag{10}$$

which provides a sub-optimal estimate for $\mathbf{e}$ in our CRC-CC system.

Because there are a total of 2^k solutions to $\mathbf{He}=\mathbf{s}$, computing the log-probability and finding the exactly maximum solution according to (10) will incur prohibitively high complexity for large k. Without additional information, it is impossible to estimate the error pattern.

B. SDSD algorithm

Define $\mathbf{Q}$ as the $n\times n$ permutation matrix used to arrange the entries of $\tilde{\boldsymbol{\gamma}}=\mathbf{Q}\boldsymbol{\gamma}$ in ascending order. Accordingly, define $\tilde{\mathbf{H}} := \mathbf{H}\mathbf{Q}^T$, $\tilde{\mathbf{c}} := \mathbf{Q}\mathbf{c}$, and $\tilde{\mathbf{e}} := \mathbf{Q}\mathbf{e}$. Since $\mathbf{Q}^T\mathbf{Q}=\mathbf{I}$, we have $\tilde{\mathbf{H}}\tilde{\mathbf{e}}=\mathbf{s}$. Let i_j stand for the index of the jth bit ($k=1,\ldots,n$) in $\tilde{\boldsymbol{\gamma}}$. For example, $i_8=30$ means γ_8 is the 30th lowest in $\boldsymbol{\gamma}$. The higher i_j is, the greater γ_j is, and c_j is correct with higher probability.

Let us further assume that the bits corresponding to the last $n-l$ entries of $\tilde{\boldsymbol{\gamma}}$ are reliable. That is, the last $n-l$ entries of $\tilde{\mathbf{e}}$ are all 0. The only unknowns in $\tilde{\mathbf{e}}$ are the first l entries. Let $\tilde{\mathbf{e}}_{[l]}$ stand for the vector comprising the first l entries of $\tilde{\mathbf{e}}$, and $\tilde{\mathbf{H}}_{[l]}$ the matrix formed by the first l columns of $\tilde{\mathbf{H}}$. Clearly, we have

$$\mathbf{s}=\tilde{\mathbf{H}}_{[l]}\tilde{\mathbf{e}}_{[l]}. \tag{11}$$

A reduced dimensionality problem results after cancelling reliable entries of $\mathbf{e}$. Notice that the $l\times 1$ error vector $\tilde{\mathbf{e}}_{[l]}$ in (11) contains less unknowns than the original $\mathbf{e}$. If we can solve (11) with affordable complexity, we can quantify the log-likelihood of $\tilde{\mathbf{e}}_{[l]}$ as

$$\ln[P(\tilde{\mathbf{e}}_{[l]})] \propto (1-2\tilde{\mathbf{e}}_{[l]})^T\tilde{\boldsymbol{\gamma}}_i. \tag{12}$$

Vector $\tilde{\mathbf{e}}_{[l]}$ can then be estimated as

$$\widehat{\tilde{\mathbf{e}}_{[l]}} = \arg\max_{\tilde{\mathbf{H}}_{[l]}\tilde{\mathbf{e}}_{[l]}=\mathbf{s}} \ (1-2\tilde{\mathbf{e}}_{[l]})^T\tilde{\boldsymbol{\gamma}}_i, \tag{13}$$

and $\tilde{\mathbf{e}}$ as

$$\widehat{\tilde{\mathbf{e}}} = \begin{bmatrix}\widehat{\tilde{\mathbf{e}}_{[l]}} \\ \mathbf{0}_{(n-l)\times 1}\end{bmatrix}, \tag{14}$$

where $\mathbf{0}_{(n-l)\times 1}$ is the $(n-l)\times 1$ all-zero vector. Upon permuting $\widehat{\tilde{\mathbf{e}}}$, we find

$$\hat{\mathbf{e}}=\mathbf{Q}^T\widehat{\tilde{\mathbf{e}}}. \tag{15}$$

The refined estimate for $\mathbf{c}$ is then obtained by

$$\check{\mathbf{c}}=\hat{\mathbf{c}}+\hat{\mathbf{e}}=\mathbf{c}+\mathbf{e}+\hat{\mathbf{e}}. \tag{16}$$

If $\hat{\mathbf{e}}$ is indeed the error pattern $\mathbf{e}$, we can successfully recover the transmitted codeword $\mathbf{c}$ and equivalently the information vector $\mathbf{b}$.

C. Implementation and complexity

1) Due to errors introduced when cancelling reliable bits, (11) is not always true. If l is too small, some erroneous bits will be regarded as reliable rendering it impossible to find the real error pattern from (11). Increasing l will increase the probability for (11) to hold true. However, if l is too large, solving (11) will be too complex. In the extreme case when $l=n$, (11) becomes a permuted version of (3). The ideal l is the index of the last erroneous bit in $\tilde{\boldsymbol{\gamma}}$; that is, l is just big enough to include all erroneous bits. Since it is generally impossible to know the error positions beforehand, we choose l in an incremental manner. For CC with memory v, a burst error length scales with v. We first set $l=v$. If a solution to (11) exists, we will assume that l is sufficiently large and then stop to estimate error patterns as in (13). If no solution exists, l will be increased to $2v$, and so forth until l reaches a preset threshold, e.g., $m+6$.

2) There is no guarantee that $\tilde{\mathbf{H}}_{[l]}$ has full column rank. Eq. (11) can be fully-determined, under-determined or over-determined. For notational brevity, let us consider the following general binary linear system:

$$\mathbf{A}\mathbf{v}=\mathbf{t}, \tag{17}$$

where the $m\times n$ binary matrix $\mathbf{A}$ and the $m\times 1$ binary vector $\mathbf{t}$ are known, and the $n\times 1$ binary vector $\mathbf{v}$ is unknown. Eq. (11) can be solved by setting $\mathbf{A}=\tilde{\mathbf{H}}_{[l]}$, $\mathbf{v}=\tilde{\mathbf{e}}_{[l]}$, and $\mathbf{t}=\mathbf{s}$.

2.a) First, we reduce the dimension of (17) by eliminating null equations: If $A_{i,j}=0$ for $j=0,\ldots,n-1$, and $t_i=0$, the ith scalar equation in (17) will be eliminated. On the other hand, if $A_{i,j}=0$ for $j=1,\ldots,n$, and $t_i=1$, then no solution exists and decoding is terminated.

We then iteratively eliminate equations with a single variable as described in [23]. Specifically, if there is any row of $\mathbf{A}$ having weight (the number of 1's) 1, the corresponding equation contains only a single unknown, which can be easily determined. Eliminating this variable from the set of equations yields a reduced set of equations. For example, if $A_{3,2}=1$ and $A_{3,j}=0$ for $j\neq 2$, we will have $v_2=t_2$. Define $\mathbf{t}^{(2)}=\mathbf{t}+v_2\mathbf{a}_2$, $\mathbf{A}^{(2)}=\mathbf{A}_{[2]}$, and $\mathbf{v}^{(2)}=\mathbf{v}_{[2]}$, where $\mathbf{A}_{[2]}$ is the sub-matrix of $\mathbf{A}$ obtained after omitting the 2nd column $\mathbf{a}_2$, and $\mathbf{v}_{[2]}$ is the sub-vector of $\mathbf{v}$ by omitting the 2nd entry v_2. The reduced system is

$$\mathbf{A}^{(2)}\mathbf{v}^{(2)}=\mathbf{t}^{(2)}, \tag{18}$$

and has only $n-1$ unknowns. We next eliminate iteratively null equations and/or equations with a single unknown, until all variables are solved or all rows of the new system have weight more than 1.

2.b) Suppose that after the iterative elimination process, the resulting system contains n_o unknowns and m_o equations: $\mathbf{A}_o\mathbf{v}_o=\mathbf{t}_o$. Following Gaussian elimination, we find the row and column permutation matrices $\mathbf{Q}_r$ and $\mathbf{Q}_c$ such that $\tilde{\mathbf{A}}_o=\mathbf{Q}_r\mathbf{A}_o\mathbf{Q}_c$ have as close to an upper triangular structure as possible. Upon defining $\tilde{\mathbf{v}}_o=\mathbf{Q}_c^T\mathbf{v}_o$ and $\tilde{\mathbf{t}}_o=\mathbf{Q}_r\mathbf{t}_o$, we obtain

$$\tilde{\mathbf{A}}_o\tilde{\mathbf{v}}_o=\tilde{\mathbf{t}}_o. \tag{19}$$

If $\tilde{\mathbf{A}}_o$ is an upper triangular matrix $\mathbf{U}_{n_o\times n_o}$, we have $n_o=m_o$, and the unique solution to (19) can be determined

by nulling-cancelling. First, we find $\tilde{v}_{o,n_o-1}$ through the last equation $\tilde{v}_{o,n_o-1} = \tilde{t}_{o,n_o-1}$. Upon eliminating $\tilde{v}_{o,n_o-1}$, we obtain the next single variable equation: $\tilde{v}_{o,n_o-2} = \tilde{t}_{o,n_o-2} + \tilde{v}_{o,n_o-1}u_{n_o-2,n_o-1}$. In this fashion, we can find all the unknowns in (19).

If $\tilde{\mathbf{A}}_o$ is in a form $[\mathbf{U}_{m_o\times m_o} \ \ \mathbf{B}_{m_o\times(n_o-m_o)}]$, where $\mathbf{B}_{m_o\times(n_o-m_o)}$ is an $m_o \times (n_o - m_o)$ arbitrary binary matrix, we rewrite (19) as $\mathbf{U}\tilde{\mathbf{v}}_{o1} = \tilde{\mathbf{t}}_o + \mathbf{B}\tilde{\mathbf{v}}_{o2}$. By enumerating all possible $\tilde{\mathbf{v}}_{o2}$'s, we can obtain the corresponding $\tilde{\mathbf{v}}_{o1}$'s. The solutions to (19) are therefore in the form $\tilde{\mathbf{v}}_o = [\tilde{\mathbf{v}}_{o1}^T \ \tilde{\mathbf{v}}_{o2}^T]^T$.

Suppose now that $\tilde{\mathbf{A}}_o$ has the following form:

$$\begin{bmatrix} \mathbf{U}_{m_1\times n_1} & \mathbf{B}_{m_1\times(n_o-n_1)} \\ \mathbf{0}_{(m_o-m_1)\times n_1} & \mathbf{0}_{(m_o-m_1)\times n_o-n_1} \end{bmatrix},$$

where $\mathbf{U}_{m_1\times n_1}$ is an upper triangular matrix. If the last $m_o - m_1$ entries of $\tilde{\mathbf{t}}$ are all zeros, we delete the last $m_o - n_o$ equations from (19) and solve the equation with $[\mathbf{U}\ \mathbf{B}]\tilde{\mathbf{v}}_o = \tilde{\mathbf{t}}_{o,[m_1]}$, where $\tilde{\mathbf{t}}_{o,[m_1]}$ is the first m_1 entries of $\tilde{\mathbf{t}}_o$. If any of the last $m_o - m_1$ entries of $\tilde{\mathbf{t}}$ is nonzero, no solution exists, and the process is terminated.

After we find a solution $\tilde{\mathbf{v}}_o$, we calculate $\mathbf{v}_o = \mathbf{Q}_c\tilde{\mathbf{v}}_o$. Combining the solutions from 2.a and 2.b, we find the complete solutions to (17).

3) The process of recovering erroneous packets with the SDSD algorithm proceeds in three steps. First, we need to find the l unreliable positions out of n bits with l smallest γ values. The Z algorithm with worst case complexity $O(nl+n+l-l^2)$ is capable of finding these l positions efficiently [21], [22]. Second, we need to obtain the set of solutions to equations (11) and (10). In most cases when there is a small set of solutions to eq. (11), the complexity is dominated by Gaussian elimination of complexity $O(l^3)$. In the worst case where exhaustive search is required, the complexity is $O(2^l)$. Third, we need to estimate the error pattern (15) and recover the codeword (16). The complexity is $O(n)$, negligible compared to the other two steps. The overall complexity of SDSD is then $O(l^3 + 2^v n + nl + n + l - l^2)$ in most cases, and $O(2^l + 2^v n + nl + n + l - l^2)$ in the worst case. At high SNR, $l \ll n$ and $l \ll 2^v$, SDSD has comparable complexity to soft CC decoding. In such a case, SDSD is favorable in terms of complexity compared to IDD algorithms because it does not require an iterative process.

D. Performance analysis for SDSD

Binary linear block and convolutional codes possess the uniform-error property. That is, for binary channel modulation, their error performance can be evaluated under the assumption that the all-zero codeword has been transmitted [24]. Consider an $(n,n/R)$ convolutional code with rate R. Let $\mathbf{c}_0$ and $\mathbf{x}_0$ represent the all-zero input message and output codeword, respectively. Further, let $\mathbf{c}$ and $\mathbf{x}$ denote a different input message and associated output codeword with Hamming weights w and d, respectively. In an AWGN channel, the pairwise error probability between $\mathbf{c}_0$ and $\mathbf{c}$ with ML decoding is [24]

$$P_2(x_0, x) = P_d = Q\left(\sqrt{\frac{2dRE_b}{N_0}}\right), \tag{20}$$

where $Q(\cdot)$ is the Gaussian Q function, E_b is the bit energy, and $N_0/2$ is the two-sided noise power spectral density. Using the Chernoff bound $Q(\sqrt{x+y}) \le Q(\sqrt{x})e^{-y/2}$, the PER is upper bounded by the union of all pairwise error probabilities:

$$\begin{aligned} P_e &\le \sum_{d=d_{min}}^{n/R} A_d P_d \\ &= \sum_{d=d_{min}}^{n/R} A_d Q\left(\sqrt{\frac{2dRE_b}{N_0}}\right) \\ &\le Q\left(\sqrt{\frac{2d_{min}RE_b}{N_0}}\right) e^{\frac{d_{min}RE_b}{N_0}} \sum_{d=d_{min}}^{n/R} A_d e^{\frac{-dRE_b}{N_0}}, \end{aligned} \tag{21}$$

where A_d is the number of codewords achieving Hamming distance d. The input-output weight enumerating function (IOWEF) is defined as [24]

$$A(W, D) = \sum_{w=0}^{n} \sum_{d=0}^{n/R} A_{w,d} W^w D^d \,, \tag{22}$$

where $A_{w,d}$ is the number of codewords with input weight w and output weight d. Thus,

$$A_d = \sum_{w=0}^{n} A_{w,d} \,. \tag{23}$$

SDSD improves error performance by eliminating some of the error patterns with low input weight. At high SNR, SDSD is capable of correcting some residual errors in $\mathbf{c}$ with weight w less than m. Let p_w represent the probability that an error pattern $\mathbf{e}$ with weight w can not be correctly identified with SDSD. Probability p_w can be approximately determined by Monto-Calo simulations. For a system with CRC-12 and CC-2, the probabilities are [0, 0, 0.006, 0.01, 0.06, 0.3, 0.5, 0.6, 0.7, 0.9, 0.9, 1] for $w = 1, \ldots, 12$, respectively. If $\mathbf{e}$ is identified correctly, the erroneous codewords $\mathbf{c}$ and $\mathbf{x}$ will be corrected, and thus they will not be counted in A_d. The equivalent A_d of the SDSD system becomes

$$\tilde{A}_d = \sum_{w=0}^{n} A_{w,d} p_w \,, \tag{24}$$

and (25) reduces to

$$\tilde{P}_e \le Q\left(\sqrt{\frac{2d_{min}RE_b}{N_0}}\right) e^{\frac{d_{min}RE_b}{N_0}} \sum_{d=d_{min}} \tilde{A}_d e^{\frac{-dRE_b}{N_0}} \,. \tag{25}$$

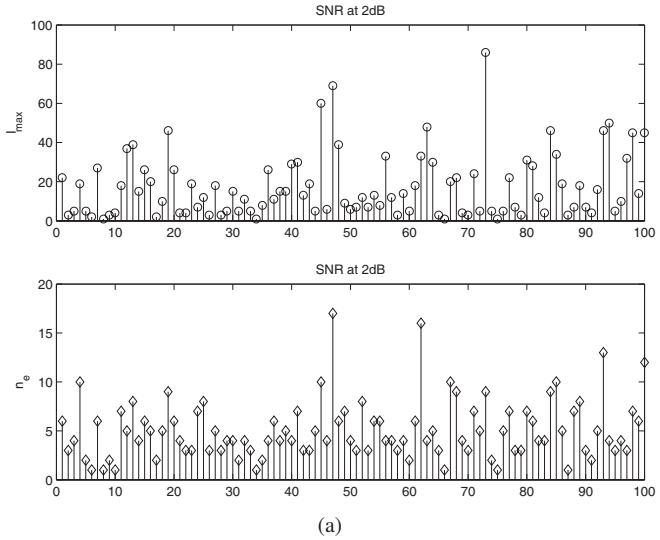
Since p_w is found using a numerical approach, (25) is only an approximation to the PER of SDSD. Nevertheless, the ensuing simulations will corroborate that (25) is indeed accurate at high SNR.

V. Simulations

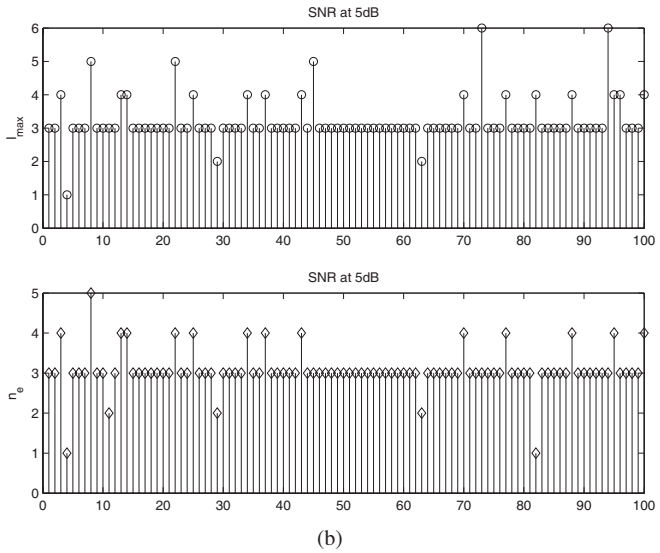
We now present simulation results of the proposed concatenated CRC-CC schemes for both AWGN and Rayleigh flat fading channels. Throughout the simulations, we will use packet error rate (PER) as our error performance measure.

We will test two CRC generator polynomials. CRC-12 with $g_{12}(x) = x^{12} + x^{11} + x^3 + x^2 + x + 1$; and CRC-24 with $g_{24}(x) = x^{24} + x^{23} + x^{14} + x^{12} + x^8 + 1$. We will also examine single PC codes with $g(x) = x^m + 1$.

Different CCs will be tested. CC-2 with memory 2 and generator $G_2(D) = [1+D+D^2;\ 1+D^2]$; CC-3 with memory



(a)



(b)

Fig. 2. For CC with memory 3 , $I_{\max}$ and n_e vary with SNR (a) at 2dB; (d) at 5dB.

3 and generator $G_3(D) = [1 + D^2 + D^3;\ 1 + D + D^2 + D^3]$; CC-4 with memory 4 and generator $G_4(D) = [1 + D^3 + D^4;\ 1 + D + D^2 + D^4]$; CC-5 with memory 5 and generator $G_5(D) = [1 + D^2 + D^3 + D^4 + D^5;\ 1 + D + D^3 + D^5]$; and finally CC-8 with memory 8 and generator $G_8(D) = [1+D+D^2+D^3+D^5+D^7+D^8;\ 1+D^2+D^3+D^4+D^5+D^8]$. We use the log-MAP algorithm for soft CC decoding.

Except for Simulation 2, the solid bold curves in PER plots will denote the performance of a conventional system with CC decoding followed by CRC detection only. In IDD, P-IDD, and IVA algorithms, the maximum number of iterations is 5. In SDSD, the maximum l we take is $m + 6$.

Simulation 1 (reliability indicator): For a certain error event with n_e bit errors, let I_j denote the index of the jth erroneous bit ($j = 1, \ldots, n_e$) in $\tilde{\boldsymbol{\gamma}}$. For example, $I_2 = 5$ means that the second erroneous bit is the fifth least reliable bit. Define $I_{\max} = \max_{j=1,\ldots,n_e} I_j$. To find the correct codeword, we need to enumerate $I_{\max}$ bit positions with the bit-flipping algorithm. The total number of codewords to test is $2^{I_{\max}}$.

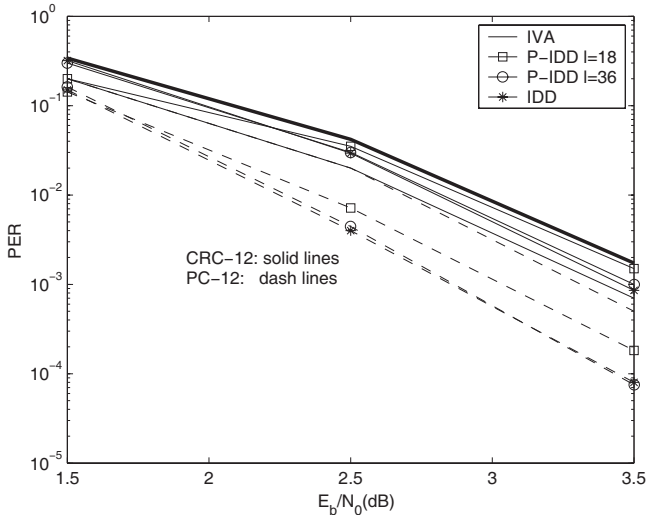


Fig. 3. IDD, P-IDD and IVA.

Figure 2 depicts how $I_{\max}$ varies with SNR when using a CC with generator $G(D) = [1+D^2+D^3\ ;\ 1+D+D^2+D^3]$. In each sub-figure, the y-axis with circled marks denotes $I_{\max}$ and the y-axis with diamond marks indices n_e. The x-axis includes 100 erroneous packets. Notice that at low SNR, n_e is large, and both the mean and variance of $I_{\max}$ are high. In this case, $I_{\max}$ is often a few times larger than n_e. The complexity of bit-flipping will be unaffordable in cases where $I_{\max}$ is large. As SNR increases, the number of errors decreases and so do the mean and variance of $I_{\max}$. At 5dB, $I_{\max}$ comes close to n_e, which suggests that with higher probability, errors happen at bit positions with smaller value of γ. That is, γ the absolute value of LLR becomes an increasingly accurate reliability indicator as SNR increases.

Simulation 2 (IVA, IDD, and P-IDD): We compare the proposed IDD and P-IDD algorithms with the IVA algorithm. We first test the IS-95 system with CC-8 and CRC-12 in AWGN. Each information block in both systems contains 172 bits. The length of each CRC packet is 184. We also test a system with the same CC, but with a single parity check code PC-12. The rate loss due to CRC parity bits and tail bits is deduced from the SNR. Figure 3 compares IDD, P-IDD and IVA. The performance improvement using both IVA and IDD is relatively small for CRC-12. Notice that IVA performs slightly better than IDD with CRC-12. Changing CRC-12 to PC-12 brings additional improvement. Further, IDD outperforms IVA with PC-12. We also observe that P-IDD enables trading off error performance for decoding complexity. Notice that P-IDD with $l = 36$, updates only about one fifth of $\mathbf{c}$, but achieves performance similar to IDD. For l values near 18, P-IDD incurs some degradation relative to IDD.

Simulation 3 (Performance of SDSD): Figure 4 depicts the performance with CC-2 and CRC-12 in AWGN. The number of information bits is 60. The rate loss due to CRC parity bits and tail bits is deduced from the SNR. We observe that SDSD brings considerable improvement to the CRC-CC system. At SNR=5dB, PER is reduced from 10^{-2} to below 10^{-4}. Both the union bound in (21) and the approximate expression in

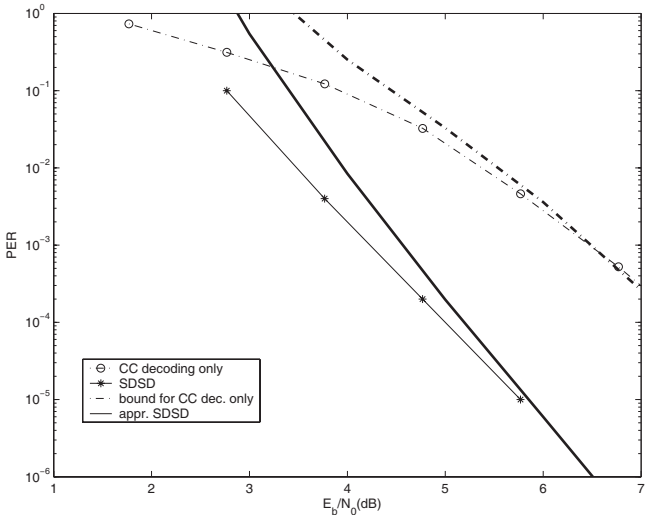


Fig. 4. Performance of SDSD for CC-2 and CRC-12.

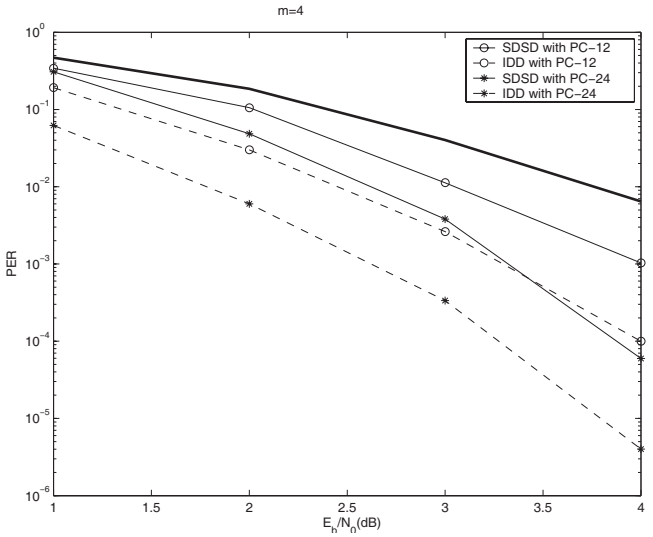


Fig. 6. SDSD compared with IDD for CC-4 and PC.

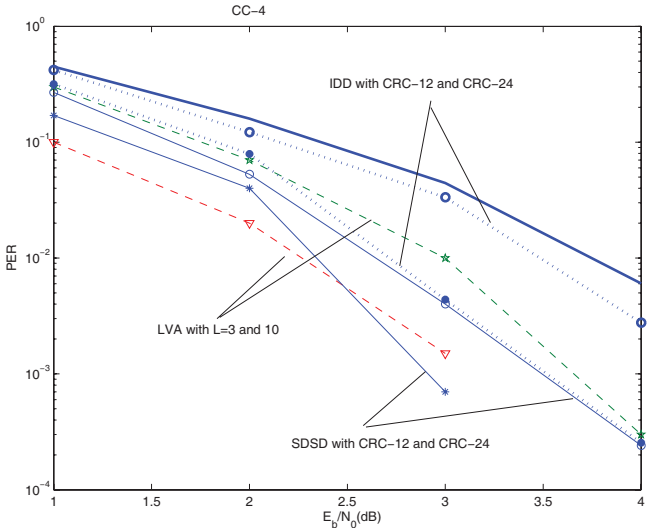


Fig. 5. SDSD compared with LVA and IDD for CC-4.

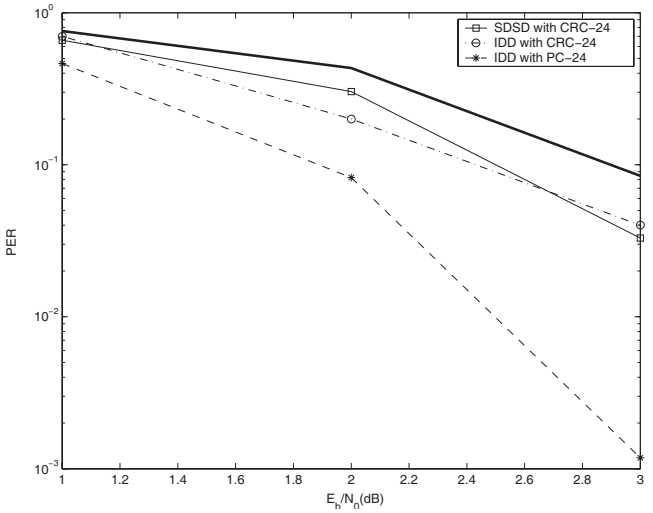


Fig. 7. SDSD compared with IDD for CC-8.

(25) approach the simulated performance at high SNR.

Simulation 4 (SDSD vs. LVA and IDD): Figure 5 depicts the performance of our algorithm compared with LVA in AWGN channels for different CRCs when CC-4 is used. The number of information bits is fixed to 60. Perfect EDC is assumed for LVA, and the rate loss introduced by CRC is not deducted from the SNR. It can be observed that with all algorithms, the PER performance is considerably improved.

SDSD shows no evident improvement relative to the conventional scheme at low SNR, e.g., below 2 dB. At high SNR, however, the performance advantage of SDSD increases as we employ more powerful CRCs; e.g., when going from CRC-12 to CRC-24. This is because as the number of equations in (11) increases, the ability of finding error patterns with SDSD increases correspondingly. Since LVA improves the performance by searching through the trellis, the improvement of LVA does not vary with the underlying CRC used. Compared with LVA, SDSD with CRC-12 outperforms LVA with list size $L = 3$. SDSD with CRC-24 outperforms LVA with $L > 10$.

IDD with CRC-12 shows no evident improvement over the conventional system. IDD with CRC-24 achieves comparable performance to LVA with list size $L = 3$, which has the same performance as SDSD with CRC-12. In this setup, SDSD outperforms IDD.

Simulations 5 (SDSD vs. IDD for more coding choices): We further compare SDSD with IDD. Figure 6 depicts the performance of the system with CC-4 and a single PC in AWGN. Again, the number of information bits is 60. Similar to Simulation 3, only the rate loss due to tail bits is deduced from the SNR. It can be observed that for both algorithms the improvement increases as the number of parity check bits increases. However, in this setup, IDD outperforms SDSD when the same PC is used. IDD with PC-12 achieves comparable performance as SDSD with PC-24. Figures 5 and 6 illustrate that for a weak CC, SDSD works better than IDD for CRC codes. But IDD works better for a single PC.

We now test the CRC-CC system with a stronger CC. The performance with CC-8 and CRC-24 or PC-24 is depicted

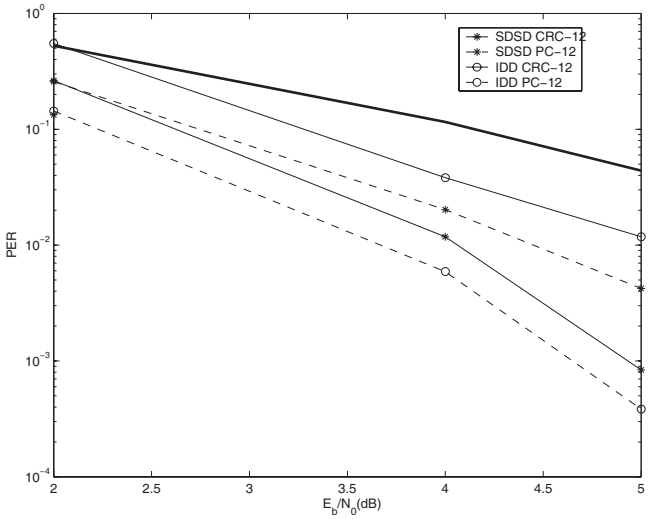


Fig. 8. SDSD compared with IDD for CC-3 in fading.

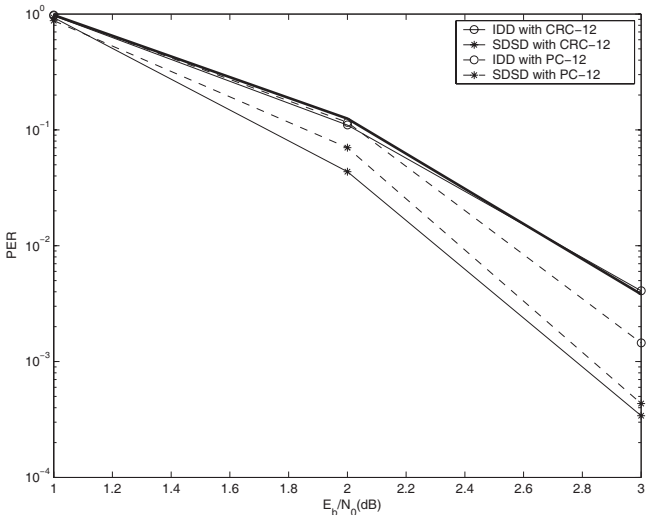


Fig. 9. SDSD compared with IDD for turbo-2 in AWGN.

in Figure 7. The number of information bits is 168. Now, the rate loss due to both CRC parity check bits and tail bits are deduced from the SNR. We observe that as CC becomes stronger, the improvement becomes less apparent. Neither IDD nor SDSD offers evident performance improvement with CRC-24. But IDD with PC-24 exhibits noticeable performance improvement. With a stronger CC, SDSD becomes less effective. The reason is that the burst length of CC scales with its memory. Once there are errors, the number of errors will be around 8, 16, 24, etc., for CC-8. For this reason, it will be difficult for SDSD to determine such error positions.

Simulations 6 (CRC-CC in flat fading channels): Figure 8 depicts PER performance of the proposed CRC-CC system with CC-3, CRC-12 and PC-12 in Rayleigh flat fading channels. The rate loss due to both CRC parity check bits and tail bits are accounted when calculating the SNR. We notice that the proposed IDD and SDSD schemes also improve performance in flat fading channels. SDSD works better than IDD for CRC codes, but IDD works better for a single PC.

Simulations 7 (SDSD with turbo codes): In this simulation, we test a CRC-turbo system in AWGN. We use a parallel concatenated convolutional code (PCCC) with a constitute CC-2 code. The number of information bits is 1024. We test the system with CRC-12 and PC-12. Three inner iterations of turbo decoding are performed before CRC checking. For IDD, one iteration of turbo decoding is performed after each IDD update. The rate loss due to both CRC parity check bits and tail bits are taken into account in the SNR calculation. PER is depicted in Figure 9, from which we infer that our SDSD algorithm improves performance. Compared with the CRC-CC system in Figure 4, the improvement is smaller. For example, SDSD in the CRC-CC system reduces PER from 10^{-2} to below 10^{-4} at 5dB. But SDSD in the CRC-PCCC system reduces PER from 10^{-2} only to 10^{-3} at 3dB. IDD in the CRC-turbo system with either PC or CRC, on the other hand, exhibits negligible improvement. The reason is that the turbo code itself is already approaching capacity. Therefore, the additional information from CRC provides no evident improvement.

VI. Conclusions

We investigated joint decoding schemes for serially concatenated cyclic redundancy check (CRC) and convolutional code (CC). We employed a soft decoding algorithm for CC decoding, and used CRC to check the hard decision output from the soft CC decoding module for errors. Conventionally, if there is any error, the erroneous packet is discarded. At high SNR, however, the number of errors is usually small, which makes it possible to identify the error positions and thus recover the message. The iterative Viterbi algorithm (IVA) and the list Viterbi algorithm (LVA) are two existing approaches for recovering the message after detecting errors. We developed several schemes to further exploit CRC as an aid to error correction.

Related to IVA, the iterative decoding-detecting (IDD) algorithm improves error performance by iteratively updating the *extrinsic* information according to the parity check matrix. Assuming errors only happen to unreliable bits indicated by small absolute values of the log-likelihood ratio (LLR), partial IDD (P-IDD) updates only a subset of unreliable bits, which enables trading off performance for complexity.

We also transformed a set of binary linear equations from highly under-determined CRC syndrome equations after cancelling reliable bits. Estimating error patterns becomes possible from this set of equations, and constitutes our soft decision syndrome decoding (SDSD) algorithm, which is capable of estimating error patterns from the soft output of the CC module without iterative decoding.

Simulations confirmed that both IDD and SDSD can considerably improve performance. But they have different advantages. IDD works better with single parity check (PC) codes than with CRC codes. SDSD outperforms both IDD and LVA for weak CC and strong CRC. Our algorithms work in both AWGN and flat fading channels. They can also be extended to concatenated turbo and CRC systems.

The algorithms in this paper recover erroneous packets and thereby boost performance at the physical layer. However,

similar to existing approaches our proposed algorithms allow erroneous packets to be present at the MAC and upper layers with higher probability than the conventional approach of discarding packets after detecting errors. This could cause serious problems in the case of data transferring when errors at the application layer are not tolerable. When SNR is sufficiently high, the number of errors in a erroneous packet is small. Consequently, the probability of falsely producing a recovered packet is low. Therefore, when applying our algorithms in a real system, the receiver can choose to perform joint CRC-CC decoding at sufficiently high SNR only. The study of MAC layer performance especially after combined with ARQ protocols constitutes an interesting direction for future research. [1]

REFERENCES

[1] G. D. F. Jr., *Concatenated Codes*. Cambridge, MA: MIT press, USA, 1966.

[2] N. Seshadri and C. E. W. Sundberg, "List Viterbi decoding algorithm with applications," *IEEE Trans. Commun.*, vol. 42, pp. 313–323, Feb. 1994.

[3] C. Nill and C. E. W. Sundberg, "List and soft symbol output Viterbi algorithms: Extensions and comparisions," *IEEE Trans. Commun.*, vol. 43, pp. 277–287, Feb. 1995.

[4] Q. Wang and L. Wei, "Graph-based iterative decoding algorithms for parity-concatenated trellis codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 3, pp. 1062–1074, Mar. 2001.

[5] L. Wei, "High-performance iterative Viterbi algorithm for conventional serial concatenated codes," *IEEE Trans. Inform. Theory*, vol. 48, no. 7, pp. 1759–1771, July 2002.

[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, G. Montorsi, D. Divsalar, and F. Pollara, "A soft-input soft-output maximum a posterior (MAP) module to decode parallel and serial concatenated codes," NASA JPL TDA Progress Report, vol. 42-127, Nov. 1996.

[8] F. J. L. R. Bahl, J. Cocke, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, pp. 284–287, 1974.

[9] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and suboptimal MAP decoding algorithms operating in the log domain," in *Proc. Intl. Conf. on Communications*, Seattle, Washington, June 1995, pp. 1009–1013.

[10] A. J. Viterbi, "An intuitive justification and a simplified implementation of the MAP decoder for conventional codes," *IEEE J. Select. Areas Commun.*, vol. 16, no. 2, pp. 260–264, Feb. 1998.

[11] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," in *Proc. GLOBECOM*, Dallas, TX, 1989, pp. 47.1.1–47.1.7.

[12] Z. W. K. Parhi, "High performance, high throughput turbo/SOVA decoder design," *IEEE Trans. Commun.*, vol. 51, no. 4, pp. 570–579, Apr. 2003.

[13] C. Huang and A. Ghrayeb, "A simple remedy for the exaggerated extrinsic information produced by the SOVA algorithm," *IEEE Trans. Wireless Commun.*, vol. 5, no. 5, pp. 996–1002, May 2006.

[14] D. Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. Inform. Theory*, vol. 18 , no. 1, pp. 170–182, Jan. 1972.

[15] N. Seshadri and P. Hoeher, "On post-decision symbol-reliability generation," in *Proc. IEEE International conference on Communications*, vol. 2, Geneva, May 23-26 1993, pp. 741–745.

[16] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. IT-27, no. 5, pp. 533–547, Sept. 1981.

[17] R. G. Gallager, "Low density parity check codes," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 21–28, Jan. 1962.

[18] G. D. F. Jr., "On iterative decoding and the two-way algorithm," in *Proc. Int. Symp. Turbo Codes and Related Topics*, Brest, France, Sept. 1997.

[19] N. Wiberg, "Codes and decoding on general graphs," Ph.D. dissertation," Linköping Univ., Linköping, Sweden, 1996.

[20] C. Berrou, P. Adde, E. Angui, and S. Faudeil, "A low complexity soft-output viterbi decoder architecture," in *Proc. IEEE International Conference on Communications*, vol. 2, Geneva, May 23-26 1993, pp. 737–740.

[21] V. Zabrodsky, "Variace na klasicke tema," *Elektronika(Czech magazine)*, vol. 6, pp. 33–34, 1992.

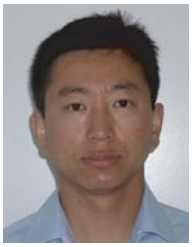
[22] ——, "A comparative study of three selection algorithms," http://www.geocities.com/SiliconValley/Garage/3323/3alg.html, 2002.

[23] T. Richardson and R. Urbanke, "Modern coding theory," downloadable from http://lthcwww.epfl.ch/papers/ics.ps.

[24] 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, no. 3, pp. 909–926, May 1998.

[1]The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government.

Renqiu Wang (Member'06) received her B.S. degree in Electrical Engineering and Information Science from the University of Science and Technology of China (USTC), 1998, the M.Sc. degree in Electrical and Computer Engineering from the University of Minnesota, 2003, and Ph.D. in Electrical and Computer Engineering from the University of Minnesota, 2006. She is now with the Qualcomm Inc. Her interests are in the general areas of signal processing, communications, and information theory and more focus on MIMO communications, coding, and cooperative communications.



Wanlun Zhao (Member'05) received his B.S. degree in Electrical Engineering and Information Science from the Huazhong University of Science and Technology, Wuhan, China, in 1996, and the M.Sc. degree in Applied and Computational Mathematics from the University of Minnesota, in 2000. He received his Ph.D. in Electrical Engineering at the University of Minnesota in 2005 and joint Qualcomm Inc. after graduation. His general research interests lie in areas of wireless communications, coding techniques, and cross-layer optimizations for wireless networks.

Georgios B. Giannakis (Fellow'97) received his Diploma in Electrical Engr. from the Ntl. Tech. Univ. of Athens, Greece, 1981. From 1982 to 1986 he was with the Univ. of Southern California (USC), where he received his MSc. in Electrical Engineering, 1983, MSc. in Mathematics, 1986, and Ph.D. in Electrical Engr., 1986. Since 1999 he has been a professor with the ECE Department at the Univ. of Minnesota, where he now holds an ADC Chair in Wireless Telecommunications in the ECE Department and serves as director of the Digital Technology Center.

His general interests span the areas of communications, networking and statistical signal processing, subjects on which he has published more than 275 journal papers, 450 conference papers, two edited books and two research monographs. Current research focuses on complex-field and network coding, cooperative wireless communications, cognitive radios, cross-layer designs, mobile ad hoc networks and wireless sensor networks.

G. B. Giannakis is the (co-)recipient of six paper awards from the IEEE Signal Processing (SP) and Communications Societies including the G. Marconi Prize Paper Award in Wireless Communications. He also received Technical Achievement Awards from the SP Society (2000), from EURASIP (2005), a Young Faculty Teaching Award and the G. W. Taylor Award for Distinguished Research from the University of Minnesota. He has served the IEEE in a number of posts, and is currently a Distinguished Lecturer for the IEEE-SP Society.