

Zigzag Codes and Concatenated Zigzag Codes

Li Ping, *Member, IEEE*, Xiaoling Huang, and Nam Phamdo, *Senior Member, IEEE*

Abstract—This paper introduces a family of error-correcting codes called *zigzag codes*. A zigzag code is described by a highly structured zigzag graph. Due to the structural properties of the graph, very low-complexity soft-in/soft-out decoding rules can be implemented. We present a decoding rule, based on the Max-Log-APP (MLA) formulation, which requires a total of only 20 addition-equivalent operations per information bit, per iteration. Simulation of a rate-1/2 concatenated zigzag code with four constituent encoders with interleaver length 65 536, yields a bit error rate (BER) of 10^{-5} at 0.9 dB and 1.4 dB away from the Shannon limit by optimal (APP) and low-cost suboptimal (MLA) decoders, respectively. A union bound analysis of the bit error probability of the zigzag code is presented. It is shown that the union bounds for these codes can be generated very efficiently. It is also illustrated that, for a fixed interleaver size, the concatenated code has increased code potential as the number of constituent encoders increases. Finally, the analysis shows that zigzag codes with four or more constituent encoders have lower error floors than comparable turbo codes with two constituent encoders.

Index Terms—Low-complexity decoding, parallel-concatenated codes, turbo codes, zigzag codes.

I. INTRODUCTION

TURBO codes, [1], [2], have attracted much interest due to their powerful performance. At a bit error rate (BER) of 10^{-5} , a 16-state rate-1/2 turbo code with interleaver length 65 536 is 0.5 dB away from the Shannon limit for a binary-input additive white Gaussian noise (AWGN) channel [1], [2]. However, the turbo decoder based on the *a posteriori* probability (APP) algorithm [3] is highly complex. To achieve the aforementioned performance, the required decoder complexity is about 192 floating-point operations per information bit, per iteration (FLOP/IB/Iter).¹

Alternative low-complexity concatenated codes based on single-parity-check (SPC) codes have been explored [4]. Near-

capacity performance has been reported for such codes at high rates [4]. Due to the simple structure of the SPC codes, the related decoding cost is very low. For example, the suboptimal Max-Log-APP (MLA) algorithm in [4] requires only 16 addition-equivalent operations (AEOs)² per information bit, per iteration (AEO/IB/Iter) for a code with four constituent encoders. Concatenated SPC codes, however, perform well only at high rates [4]. They also have relatively high error floors.

In this paper, we present a class of codes called *zigzag codes* and their concatenation schemes. The performance of the concatenated zigzag codes are close to the standard turbo codes (with only about 0.3-dB difference). However, the decoding complexity of the concatenated zigzag codes is considerably lower. For instance, the MLA algorithm costs only 20 AEO/IB/Iter for rate-1/2 concatenated zigzag codes with four constituent encoders. (As a comparison, according to [5], the MLA decoding of an N -state turbo code with two constituent encoders costs about $30N$ AEO/IB/Iter.) The proposed codes work well for medium to high rates (rates of 1/2 and higher). They also have very low error floors, which appear to be lower than turbo codes.

II. DESCRIPTION OF THE CODE

Adopting the framework of [6], a zigzag code can be described graphically as shown in Fig. 1. Here, white nodes represent information bits: $\{d(i, j)\}$, $i = 1, 2, \dots, I$, $j = 1, 2, \dots, J$. Black nodes represent parity bits: $\{p(i)\}$, $i = 1, 2, \dots, I$. We call

$$[p(i-1), d(i, 1), d(i, 2), \dots, d(i, J), p(i)]$$

a *segment*. The parity bits are chosen such that each segment on the graph contains an even number of ones. The code is systematic with coding rate $J/(J+1)$. The encoding process is straightforward. The parity bits are formed progressively as follows:

$$p(1) = \sum_{j=1}^J d(1, j) \bmod 2$$

$$p(i) = \sum_{j=1}^J d(i, j) + p(i-1) \bmod 2, \quad i = 2, 3, \dots, I. \quad (1)$$

Note that the zigzag code is completely parameterized by the pair (I, J) . The error-correcting capability of the zigzag code itself is weak since it has minimum distance $d_{\min} = 2$ for any

²An AEO (addition-equivalent operation) is either an addition, a subtraction, or a comparison.

Manuscript received December 15, 1999. The material in this paper was presented in part at the IEEE Information Theory and Networking Workshop, Metsovo, Greece, June/July 1999.

L. Ping is with the Department of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong (e-mail: eeliping@cityu.edu.hk).

X. Huang is with the Department of Electrical and Computer Engineering, State University of New York at Stony Brook, Stony Brook, NY 11794-2350 USA (e-mail: xhuang@ece.sunysb.edu).

N. Phamdo was with the Department of Electrical and Computer Engineering, State University of New York at Stony Brook, Stony Brook, NY 11794-2350. He is now with the Applied Physics Laboratory, The Johns Hopkins University, Laurel, MD 20723 USA (e-mail: nam.phamdo@jhuapl.edu).

Communicated by R. J. McEliece, Guest Editor.

Publisher Item Identifier S 0018-9448(01)00716-7.

¹A FLOP involves a multiplication and an addition. The complexity is estimated as follows: (16 states) * (two branches entering and leaving each state) * (once going forward + once going backward) * (two constituent encoders) * (1/2 FLOP for $\gamma + 1$ FLOP for α) see [1], [2]. Approximately 20 iterations are needed to achieve the best performance. This is true for all decoding rules discussed in this paper.

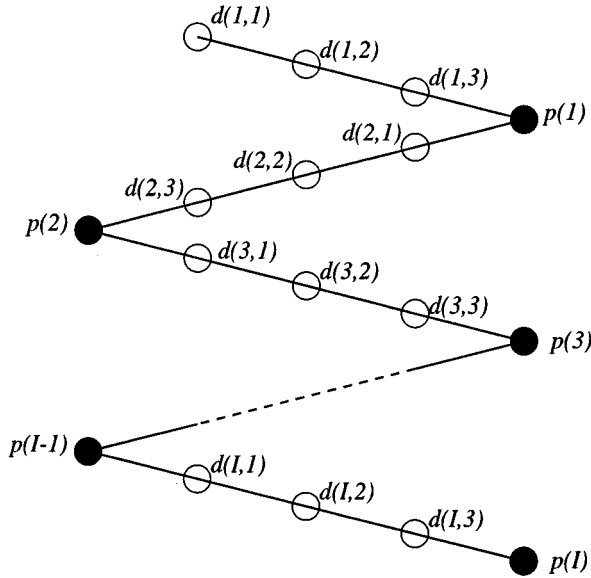


Fig. 1. Graph representation of zigzag code; white nodes represent information bits; black nodes represent parity bits; each segment has even parity; $J = 3$.

pair (I, J) . However, as we will later see, it is very useful in a concatenated construction.

III. RELATION WITH OTHER CODES

There are several other ways of looking at the zigzag code, as listed below.

A. Modified SPC Code

One can consider the zigzag code as a modified form of the parallel SPC code array used in [4]. In the parallel SPC code, one parity-check bit is added to every row of data

$$\hat{p}(i) = \sum_{j=1}^J d(i, j) \bmod 2, \quad i = 1, 2, \dots, I. \quad (2)$$

The zigzag check bits $p(i)$ could be obtained from the SPC check bits $\hat{p}(i)$ as follows:

$$p(i) = \sum_{i'=1}^i \hat{p}(i') \bmod 2. \quad (3)$$

B. Two-State Convolutional Code

One can also consider the zigzag code as a two-state rate-1/2 systematic convolutional code with generator polynomial matrix $G(X) = [1 \ 1/(1+X)]$. The parity sequence is punctured so that only one parity bit is transmitted for every J information bits—thus, reducing the rate to $J/(J+1)$. The trellis-based APP and MLA decoders of the convolutional code, however, is much less efficient than the alternative algorithms based on the zigzag representation.

C. Low-Density Parity-Check Code

It is obvious that for small values of J , the zigzag code has a very sparse parity-check matrix. Therefore, the zigzag code can be regarded as a special case of the low-density parity-check (LDPC) code considered in [7]. Actually, it is a graphical form

of semirandom LDPC codes [8]. Due to its regular structure, we can develop exact formulas for the APP and MLA decoders for the zigzag code. Furthermore, the encoding process and the analysis of zigzag code are much simpler than LDPC code.

IV. EFFICIENT SOFT-IN/SOFT-OUT DECODING OF ZIGZAG CODES

The zigzag code falls into the general code family with tree structures [6]. The two-way schedule described in [6] can be applied to develop its APP decoding algorithm. For the constituent SPC code, the local decoding can be accomplished based on the parity probability ratio (ppr) function studied in [9]. The resultant global APP decoder of the zigzag code requires about $(7 + 3/J)$ multiplications and three additions per information bit. We will derive a very efficient MLA decoding algorithm below. Its performance is slightly worse (about 0.5 dB) than the APP decoder, but its cost is only $(3 + 4/J)$ AEO/IB.

Let $D = \{d(i, j)\}$ be an $I \times J$ array of information bits and let $P = \{p(i)\}$ be the $I \times 1$ parity vector. Let $Z = (D, P)$ be the $\{+1, -1\}$ -modulated codeword (“zero” $\mapsto +1$, “one” $\mapsto -1$) and $\tilde{Z} = (\tilde{D}, \tilde{P})$ be the noisy received vector. Let

$$W(a_1, a_2, \dots, a_n) \triangleq \left[\prod_{j=1}^n \text{sign}(a_j) \right] \min_{1 \leq j \leq n} |a_j| \quad (4)$$

and assume that the implementation of the MLA decoding algorithm is as follows [4], [5], [10]–[13].

- 1) Calculate the *forward* MLA of the parity bits

$$\begin{aligned} F[p(0)] &= +\infty \\ F[p(i)] &= \tilde{p}(i) + W\left(F[p(i-1)], \tilde{d}(i, 1), \tilde{d}(i, 2), \dots, \tilde{d}(i, J)\right), \\ & \quad i = 1, 2, \dots, I. \end{aligned} \quad (5)$$

- 2) Calculate the *backward* MLA of the parity bits

$$\begin{aligned} B[p(I)] &= \tilde{p}(I), \\ B[p(i-1)] &= \tilde{p}(i-1) + W\left(\tilde{d}(i, 1), \tilde{d}(i, 2), \dots, \tilde{d}(i, J), B[p(i)]\right), \\ & \quad i = I, I-1, \dots, 2. \end{aligned} \quad (6)$$

- 3) Determine the MLA of the information bits as follows:

$$\begin{aligned} L[d(i, j)] &= \tilde{d}(i, j) + W\left(F[p(i-1)], \tilde{d}(i, 1), \tilde{d}(i, 2), \dots, \right. \\ & \quad \left. \tilde{d}(i, j-1), \tilde{d}(i, j+1), \dots, \tilde{d}(i, J), B[p(i)]\right). \end{aligned} \quad (7)$$

Consider one segment of the code and ignore the operations for taking the absolute value and negation. The evaluation of (5) requires J comparisons and one addition. The result of (5) can be used for evaluating (6) with one comparison and one addition. The evaluation of $W(\cdot)$ in (7) is performed once for the whole segment. We first find the elements among

$$[F[p(i-1)], \tilde{d}(i, 1), \tilde{d}(i, 2), \dots, \tilde{d}(i, J), B[p(i)]]$$

with the minimum and the second minimum amplitudes. This costs $1 + J$ comparisons [Notice that the minimum element can be found with only one extra comparison based on the result of (6) and the second minimum requires J comparisons]. The output of $W(\cdot)$ in (7) is either the minimum or the second min-

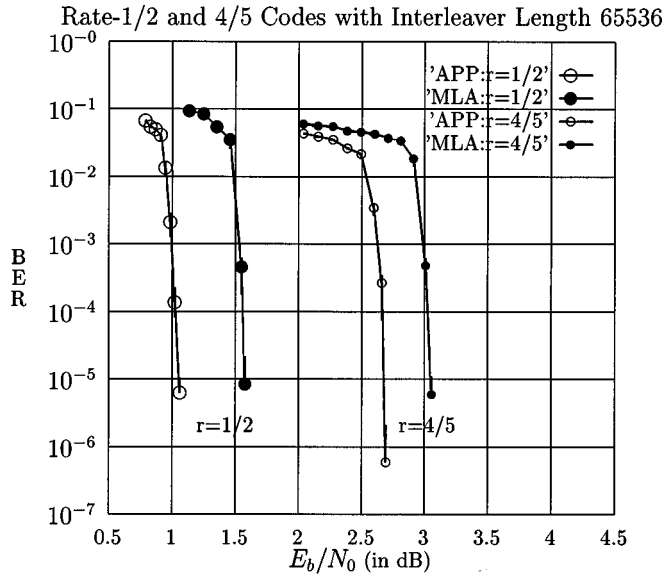


Fig. 2. Performance of concatenated zigzag codes with APP and MLA decoding; for rate-1/2, $(I, J, K) = (16384, 4, 4)$, and the Shannon limit is 0.2 dB; for rate-4/5, $(I, J, K) = (4096, 16, 4)$ and the Shannon limit is 2.1 dB.

imum values obtained above with the appropriate sign (see [4, eq. (6)]). Finally, (7) requires J additions per segment. Since there are J information bits per segment, the overall complexity of the MLA decoder is $(3 + 4/J)$ AEO/IB.

V. CONCATENATED ZIGZAG CODES

A concatenated zigzag code is described by a triplet (I, J, K) . Let $D_k = \pi_k(D)$ be an interleaved version of the data matrix D , $k = 1, 2, \dots, K$ (the number of constituent encoders). For each D_k , we form an $I \times 1$ parity column vector P_k according to (1). The transmitted codeword consists of $[D, P_1, P_2, \dots, P_K]$ and the overall code rate is $J/(J + K)$, i.e., $(n, k) = (IJ + IK, IJ)$. For each constituent code, we use the soft-in/soft-out MLA algorithm described in the previous section. The overall decoding structure of the concatenated zigzag code is exactly the same as in [4].

For concatenated codes, we need one additional AEO/IB/Iter to process the extrinsic information before the MLA decoding of each constituent encoder [4]. Therefore, the decoding cost for a concatenated zigzag code with K constituent encoders is $K(4 + 4/J)$ AEO/IB/Iter, e.g., with $K = J = 4$, the decoding cost is 20 AEO/IB/Iter.

VI. SIMULATION PERFORMANCE

In Fig. 2, we present simulation results of the rate-1/2 $((I, J, K) = (16384, 4, 4))$ and rate-4/5 $((I, J, K) = (4096, 16, 4))$ concatenated zigzag codes. The interleaver length (i.e., the number of information bits involved) is 65 536 in both cases. For each point on the performance curves, we simulate a minimum of 100 bit errors. The results presented are for 20 iterations. Note that MLA decoding is inferior to APP decoding by approximately 0.5 dB.

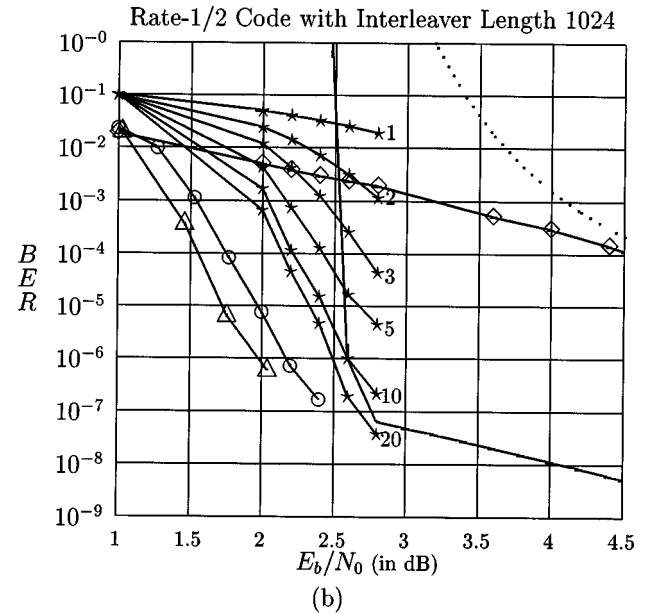
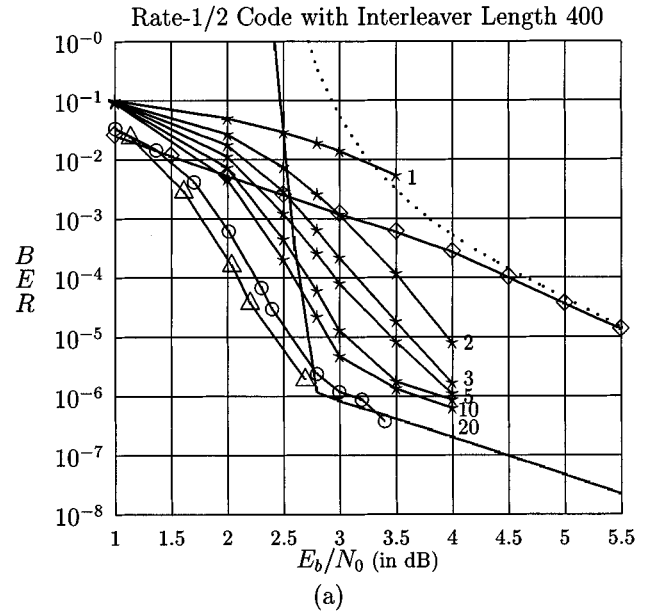


Fig. 3. (a) Performance of concatenated zigzag code with $(I, J, K) = (100, 4, 4)$. (b) $(I, J, K) = (256, 4, 4)$. MLA with 1, 2, 3, 5, 10, and 20 iterations (*); zigzag union bound (solid lines); APP with 20 iterations (\circ); turbo code (23,37) with APP and 18 iterations (Δ); SPC MLA 20 iterations (\diamond); SPC union bound (dotted line).

In Fig. 3(a) and (b), we present simulation results of the rate-1/2 codes with interleaver length of 400 $((I, J, K) = (100, 4, 4))$ and interleaver length of 1024 $((I, J, K) = (256, 4, 4))$, respectively. We also plot in these figures the performance with fewer (than 20) iterations and the union bound of the code assuming a random uniform interleaver and maximum-likelihood (ML) decoding. The derivation of the union bound is based on [14] and is described in the following section. Also presented in this figure is the performances of the 16-state turbo code with two constituent encoders in octal form (23,37) with 18 iterations of APP decoding [15]. No attempt has been made to optimize the interleavers. One can see clearly from Fig. 3(a) and (b) that for low E_b/N_0 , the performances

of the concatenated zigzag codes are slightly (~ 0.3 dB) worse than those of the turbo code.

Finally, comparing the performance of the concatenated zigzag code with the concatenated SPC code [with component codes defined in (2)] [4], we see that the zigzag code has a much better performance.

VII. BER BOUNDS

The simulation results presented in the previous section are valid for low signal-to-noise ratio (SNR) in the range of $E_b/N_0 = 4$ dB and below. For higher SNR, a union bound analysis is used to obtain performance results. In the following, we use C to represent the constituent code and C^K to represent the concatenated code with K constituent encoders.

A. General Parallel-Concatenated Codes

The following is an outline of the procedure developed in [14] for finding the BER union bound of a general parallel-concatenated code. Let

$$A^C(W, Z) \triangleq \sum_{w=0}^k \sum_{j=0}^{n-k} A_{w,j}^C W^w Z^j \quad (8)$$

be the *input-redundancy weight enumerating function* (IRWEF) for an (n, k) linear systematic block code C , where $A_{w,j}^C$ is the number of codewords with input weight w , parity weight j , and codeword weight $w + j$. Numerically, the coefficients of $A^C(W, Z)$ can be represented by a matrix of size $(k + 1) \times (n - k + 1)$, with row and column indexes corresponding to w and j , respectively.

The *conditional weight enumerating function* (CWEF) [14] for C is defined as

$$A_w^C(Z) \triangleq \sum_{j=0}^{n-k} A_{w,j}^C Z^j, \quad w = 0, 1, \dots, k. \quad (9)$$

Numerically, the coefficients of $A_w^C(Z)$ form the w th row of the matrix for $A^C(W, Z)$. Based on the uniform interleaver assumption of [14], the CWEF of C^K can be found from that of C as

$$A_w^{C^K}(Z) = \frac{[A_w^C(Z)]^K}{\binom{IJ}{w}^{K-1}}, \quad w = 0, 1, \dots, IJ \quad (10)$$

where IJ is the interleaver size. Using the coefficients of $A_w^{C^K}(Z)$ as the w th row, we can find $A^{C^K}(W, Z)$ for the concatenated code C^K . Finally, the union bound on BER for C^K is given by [14]

$$P_b \leq \frac{1}{2} \sum_{w=0}^k \sum_{j=0}^{n-k} \frac{w}{k} A_{w,j}^{C^K} \cdot \operatorname{erfc} \left(\sqrt{(w+j) \frac{R_c E_b}{N_0}} \right) \quad (11)$$

where $R_c = k/n$ is the code rate and E_b/N_0 is the bit SNR. (Note that w/k is the conditional BER given that a (w, j) -type decoding error has occurred.) In (11), we have assumed binary-phase shift keying (BPSK) modulation, an AWGN channel, and soft-decision ML decoding.

The key issue for the method outlined above is to derive the IRWEF for the constituent code C . This can be a complicated issue for a general code (e.g., a general convolutional code). However, we will show in the following that, due to the simple

code structures of the SPC arrays and zigzag codes, their IRWEF can be generated by some very simple formulas.

B. Parallel-Concatenated Zigzag Codes

In the following, we will develop a recursive method for generating the IRWEF of a zigzag code. Let $A^{C_i}(W, Z)$ be the IRWEF of a zigzag code with i rows and J columns. We can decompose $A^{C_i}(W, Z)$ into even and odd parts as

$$A^{C_i}(W, Z) = A_{\text{even}}^{C_i}(W, Z) + A_{\text{odd}}^{C_i}(W, Z) \quad (12)$$

where $A_{\text{even}}^{C_i}(W, Z)$ (resp. $A_{\text{odd}}^{C_i}(W, Z)$) includes all the terms with even (resp., odd) powers of W . Clearly, for $i = 1$

$$A_{\text{even}}^{C_1}(W, Z) = \sum_{j=0}^{\lfloor J/2 \rfloor} \binom{J}{2j} W^{2j} \triangleq B_{\text{even}}(W) \quad (13)$$

$$A_{\text{odd}}^{C_1}(W, Z) = \sum_{j=0}^{\lfloor (J-1)/2 \rfloor} \binom{J}{2j+1} W^{2j+1} Z \triangleq B_{\text{odd}}(W)Z. \quad (14)$$

Suppose that

$$A^{C_{i-1}}(W, Z) = A_{\text{even}}^{C_{i-1}}(W, Z) + A_{\text{odd}}^{C_{i-1}}(W, Z)$$

is known and consider an extra i th row. Notice that from (1)

$$p(i) = p(i-1) + \sum_{j=1}^J d(i, j) = \sum_{i'=1}^i \sum_{j=1}^J d(i', j). \quad (15)$$

Thus, $p(i)$ actually indicates the parity of the total input weight from row 1 to row i . Let n_i be the parity of $d(i, 1), \dots, d(i, J)$. $A_{\text{even}}^{C_i}(W, Z)$ corresponds to two situations: $p(i-1)$ and n_i are both even or they are both odd. In either case, $p(i)$ is zero. This leads to

$$A_{\text{even}}^{C_i}(W, Z) = A_{\text{even}}^{C_{i-1}}(W, Z) B_{\text{even}}(W) + A_{\text{odd}}^{C_{i-1}}(W, Z) B_{\text{odd}}(W). \quad (16)$$

Similarly, $A_{\text{odd}}^{C_i}(W, Z)$ corresponds to the situation where $p(i-1)$ is odd and n_i is even, or *vice versa*. In either case, $p(i)$ is one. This leads to

$$A_{\text{odd}}^{C_i}(W, Z) = \left[A_{\text{odd}}^{C_{i-1}}(W, Z) B_{\text{even}}(W) + A_{\text{even}}^{C_{i-1}}(W, Z) B_{\text{odd}}(W) \right] Z. \quad (17)$$

Finally, the IRWEF, $A^{C^K}(W, Z)$, of the parallel-concatenated zigzag code can be obtained by applying the one-dimensional convolution $K - 1$ times to each row of $A^{C_1}(W, Z)$ and then dividing by $\binom{IJ}{w}^{K-1}$, just as in(10).

C. Some Numerical Issues

In computing the union bound, we may encounter two numerical problems when the interleaver size IJ is large. The first is that the values of $A_{w,j}^C$ may be extremely large (well beyond the limit of IEEE floating-point numbers, which is 1.79×10^{308} [16]). The second problem is that the number of terms in the double sum in (11) is very large—resulting in a significant amount of computation. The first problem is solved by representing each value of $A_{w,j}^C$ by a floating-point number (mantissa) and a 16-bit integer (exponent). Special C subroutines are written for multiplication/division and addition/subtraction. The second problem is solved by approximating the double sum in (11) by a smaller sum, which includes only those values of

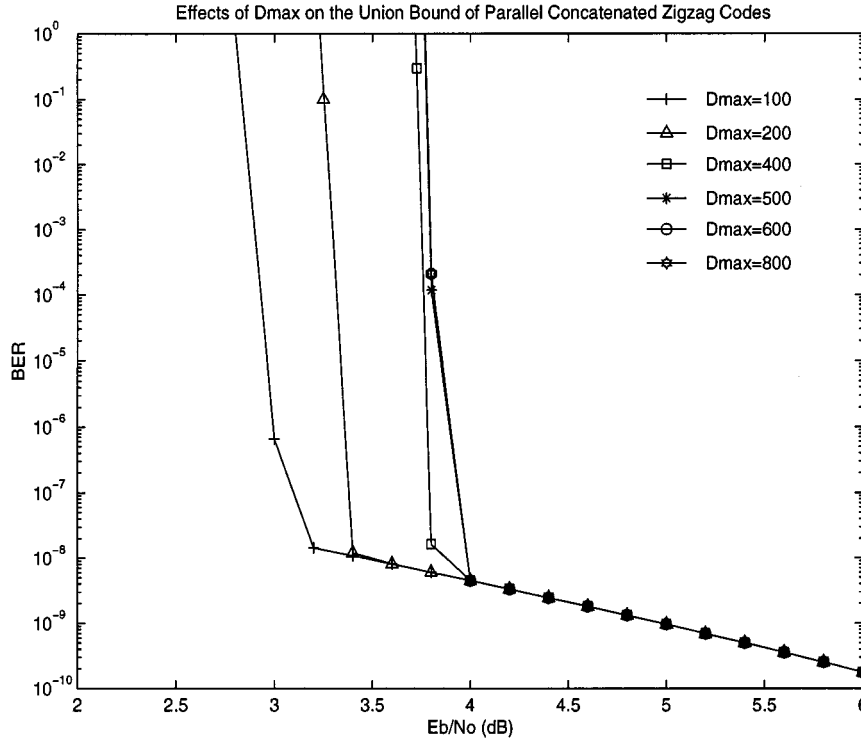


Fig. 4. Effects of D_{\max} on the union bound of parallel-concatenated zigzag codes. $(I, J, K) = (256, 16, 4)$, $(n, k) = (5120, 4096)$, and rate = $4/5$.

(w, j) for which $0 \leq w + j \leq D_{\max}$. Note that with this approach, the whole matrix $A^{C^K}(W, Z)$ need not be calculated. This approach is similar to that in [14], and is based on the observation that the upper bound is dominated by the codewords with low and moderate Hamming weights. To obtain an accurate analysis, the value of D_{\max} must be sufficiently large. Fig. 4 reports the effects of D_{\max} on the bound of parallel-concatenated zigzag code with $(I, J, K) = (256, 16, 4)$. The figure illustrates that setting $D_{\max} = 500$ will yield sufficiently accurate results. In the following, we report some analytical results on parallel-concatenated SPC arrays and zigzag codes according to this approximation. In all cases, we choose $D_{\max} = 500$.

VIII. ANALYTICAL RESULTS

Fig. 5 presents the analytical union bounds of parallel-concatenated zigzag codes (with four constituent encoders) with a fixed rate of $1/2$ and different interleaver sizes ($IJ = 16, 64$, and 256). For comparison purposes, we include the union bounds of comparable concatenated SPC codes [4]. The latter is similar to the concatenated zigzag codes except that the parity bits are generated based on (2) instead of (1). Note that for the concatenated SPC codes, increasing the interleaver size does not improve the performance much in the medium E_b/N_0 range. The reason for this is explained below [17].

There are two terms in (11) which dominate the performance of parallel-concatenated SPC array.

- The first term, $A_{1,K}^{C^K}$, corresponds to codewords with input weight one and parity weight K (P_1, P_2, \dots, P_K each has Hamming weight one). There are IJ such codewords, so $A_{1,K}^{C^K} = IJ$.

- The second term, $A_{2,0}^{C^K}$, corresponds to the situation where the input weight is two, and furthermore the only two 1's appear in the same segment in every constituent code (hence P_1, P_2, \dots, P_K are all zeros). It can be shown that

$$A_{2,0}^{C^K} = I \binom{J}{2} \left(\frac{J-1}{I \times J - 1} \right)^{K-1}.$$

We observed that these two terms dominate the performance of a concatenated SPC array and their relative effects are different in different E_b/N_0 ranges. For medium E_b/N_0 , the first term dominates since typically (for sufficiently large IJ) $A_{1,K}^{C^K}$ is much larger than $A_{2,0}^{C^K}$. In this case, the BER performance can be approximated by

$$P_b \approx \frac{1}{2} \cdot \frac{A_{1,K}^{C^K}}{IJ} \cdot \operatorname{erfc} \left(\sqrt{(K+1) \frac{R_c E_b}{N_0}} \right). \quad (18)$$

This approximation, shown by the dashed line in Fig. 6 for $(I, J, K) = (256, 4, 4)$, is quite accurate for E_b/N_0 from 4 to 10 dB. Notice that (18) is independent of the interleaver size IJ . Hence increasing the interleaver size beyond 1024 will not improve the performance of a concatenated SPC array in this range.

At very high E_b/N_0 , the effect of the second term takes over since it has lower weight. In this case, the BER performance can be approximated by

$$P_b \approx \frac{1}{2} \cdot \frac{2}{IJ} \cdot A_{2,0}^{C^K} \cdot \operatorname{erfc} \left(\sqrt{\frac{2R_c E_b}{N_0}} \right). \quad (19)$$

The above approximation is shown by the dotted line in Fig. 6 for $(I, J, K) = (256, 4, 4)$. Increasing interleaver length can

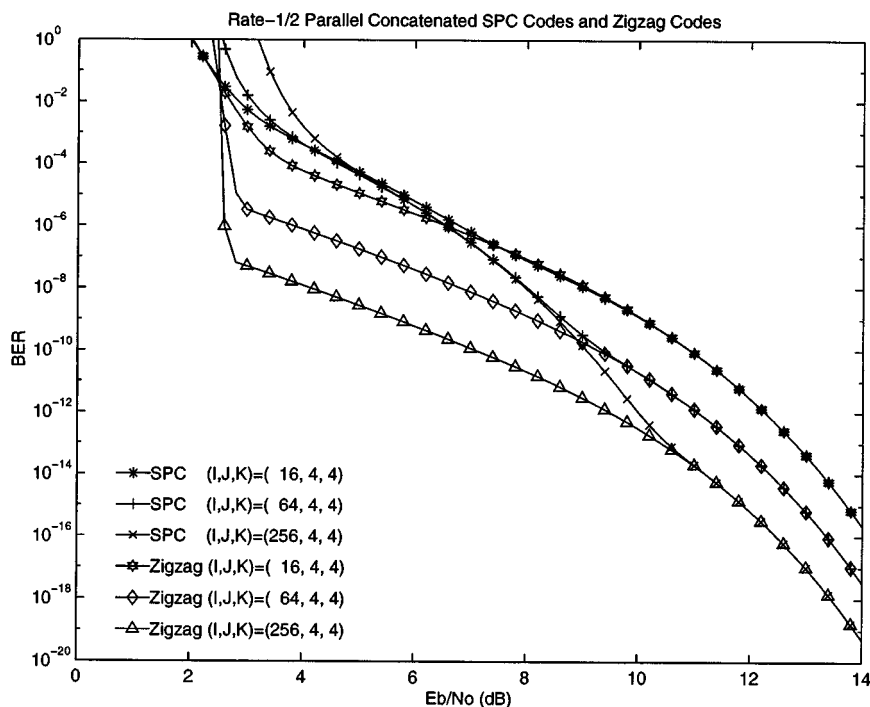


Fig. 5. Upper bounds to the BER of parallel-concatenated SPC and zigzag codes with different interleaver size. Fixed number of constituent encoders $K = 4$ and rate = $1/2$. $D_{\max} = 500$.

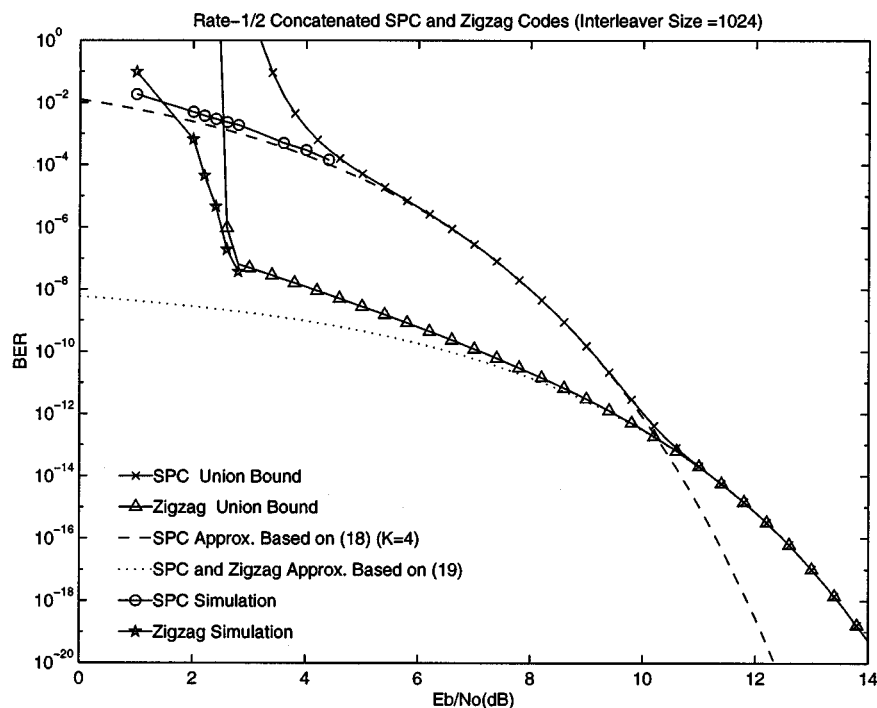


Fig. 6. Union bounds, approximations of the union bounds, and MLA simulation results of concatenated SPC and zigzag codes. $(I, J, K) = (256, 4, 4)$, $(n, k) = (2048, 1024)$, rate = $1/2$, and $D_{\max} = 500$.

normally improve the performance in this range, since the larger IJ , the smaller $A_{2,0}^{C,K}$ (assuming that J and K are fixed and I is increased).

Notice that (18) does not apply to the concatenated zigzag code. In fact, for the latter case, a weight-1 input sequence

results in a parity vector of the form $0, 0, 0, \dots, 1, 1, \dots, 1$. The nonzero section starts from the segment containing the only nonzero input bit. This implies that for weight-1 input sequences, the codeword weight increases with high probability as the interleaver size increases. Consequently, the performance

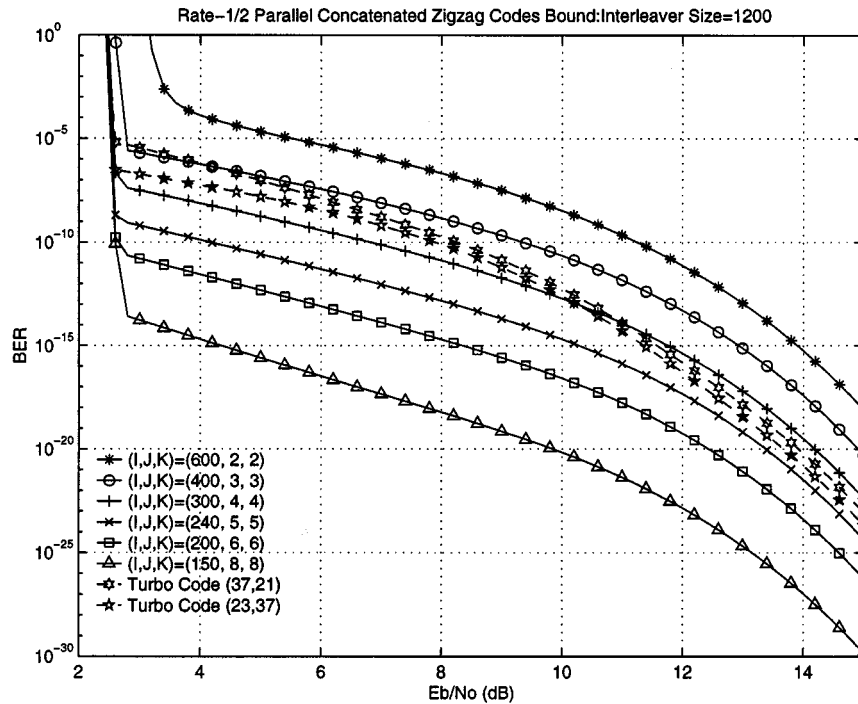


Fig. 7. Upper bounds to the bit error probability of parallel-concatenated zigzag codes with different number of constituent encoders K and (37, 21) and (23, 37) turbo codes. $(n, k) = (2400, 1200)$, rate = 1/2, and $D_{\max} = 500$.

of concatenated zigzag code is only dominated by (19). This results in much better performance for the zigzag-code-based schemes in medium E_b/N_0 range, as seen in Fig. 6. At very high E_b/N_0 , the performances of the concatenated SPC array and zigzag code converge since they are both dominated by (19).

Fig. 7 compares the BER upper bounds for parallel-concatenated zigzag codes with different K , fixed interleaver size ($IJ = 1200$), and fixed code rate 1/2. This figure illustrates that as the number of constituent encoders K increases, the code potential increases, assuming optimal (ML) sequence decoding. This phenomenon can be attributed to the constant $\binom{IJ}{w} K^{-I}$ in (10) which grows exponentially in K . Similar results were found for other interleaver sizes and code rates. This finding suggests that to achieve a low BER with a small interleaver (hence low latency), a code with a large number of constituent encoders can be used. However, it is worthwhile to mention that the suboptimality of the iterative decoding algorithms described in [1], [2], [4], and in Section IV becomes more enhanced as K increases. Furthermore, the decoding cost increases approximately linearly with K (see Section V). From our simulation experiments, $K = 4$ seems to be a good choice for rate-1/2 codes.

In Fig. 7, we also show the union bound of the (37, 21) [1], [2] and (23, 37) [15] turbo codes with the same interleaver length (1200 bits). All the codes shown are not terminated. The rate-1/2 turbo code is achieved by puncturing all the parity bits with odd indexes (the last parity bit is not punctured). This figure illustrates that the concatenated zigzag codes with four or more constituent encoders can achieve lower error floor than turbo codes (with two constituent encoders) for the range

of median E_b/N_0 . It is noted that the union bound analysis yields an accurate assessment of the code performance only at reasonably high E_b/N_0 . For very low E_b/N_0 , the simulation results in Section VI show that the (23, 37) turbo code is better than the zigzag code with four constituent encoders by 0.2–0.3 dB at 10^{-4} BER.

IX. CONCLUSION

We present a class of graphical codes called zigzag codes. The structural properties of these codes result in a very low-cost iterative decoding. The performances of the zigzag codes are within 1.0–1.5 dB of the Shannon limit. These codes perform well for medium to high rates. Finally, the error floors of the concatenated zigzag codes with multiple constituent encoders are lower than that of turbo codes.

REFERENCES

- [1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *Proc. 1993 IEEE Int. Conf. Communications*, May 1993, pp. 1064–1070.
- [2] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo codes," *IEEE Trans. Commun.*, vol. 44, pp. 1261–1271, Oct. 1996.
- [3] 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.
- [4] L. Ping, S. Chan, and K. L. Yeung, "Iterative decoding of multi-dimensional concatenated single parity check codes," in *Proc. 1998 IEEE Int. Conf. Communications*, June 1998, pp. 131–135.
- [5] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *Proc. 1995 IEEE Int. Conf. Communications*, June 1995, pp. 1009–1013.

- [6] F. R. Kschischang and B. J. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 219–230, Feb. 1998.
- [7] R. G. Gallager, "Low density parity check codes," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 21–28, Jan. 1962.
- [8] L. Ping, W. K. Leung, and N. Phamdo, "Low density parity check codes with semi-random parity check matrix," *IEE Electron. Lett.*, vol. 35, no. 1, pp. 38–39, Jan. 1999.
- [9] L. Ping, "Modified turbo codes with low decoding complexity," *IEE Electron. Lett.*, vol. 34, no. 23, pp. 2228–2229, Nov. 1998.
- [10] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor graphs and the sum-product algorithm," , vol. 47, pp. 498–519, Feb. 2001.
- [11] L. Ping and N. Phamdo, "Zigzag codes and concatenated zigzag codes," in *Proc. IEEE Information Theory and Networking Workshop*, Metsovo, Greece, June/July 1999.
- [12] 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.
- [13] X. Huang, "Turbo codes and zigzag codes: Performance analysis and simulation," Ph.D. dissertation, State Univ. New York at Stony Brook, Stony Brook, NY, Dec. 2000.
- [14] S. Benedetto and G. Montorsi, "Unveiling turbo codes: Some results on parallel concatenated coding schemes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 409–428, Mar. 1996.
- [15] S. Benedetto, R. Garelo, and G. Montorsi, "A search for good convolutional codes to be used in the construction of turbo codes," *IEEE Trans. Commun.*, vol. 46, pp. 1101–1105, Sept. 1998.
- [16] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 1992.
- [17] X. Huang, N. Phamdo, and L. Ping, "BER bounds on parallel concatenated single parity check arrays and zigzag codes," in *Proc. 1999 IEEE GLOBECOM*, Brazil, Dec. 1999, pp. 2436–2440.