ELF Codes: Concatenated Codes with an Expurgating Linear Function as the Outer Code

Richard Wesel*, Amaael Antonini*, Linfang Wang*, Wenhui Sui*, Brendan Towell*, and Holden Grissett*
*Department of Electrical and Computer Engineering, University of California, Los Angeles, Los Angeles, CA 90095, USA
Email: {wesel, amaael, lfwang, wenhui.sui, brendan.towell, holdengs}@ucla.edu

***Abstract*—An expurgating linear function (ELF) is an outer code that disallows low-weight codewords of the inner code. ELFs can be designed either to maximize the minimum distance or to minimize the codeword error rate (CER) of the expurgated code. A list-decoding sieve can efficiently identify ELFs that maximize the minimum distance of the expurgated code. For convolutional inner codes, this paper provides analytical distance spectrum union (DSU) bounds on the CER of the concatenated code.**

For short codeword lengths, ELFs transform a good inner code into a great concatenated code. For a constant message size of $K = 64$ bits or constant codeword blocklength of $N = 152$ bits, an ELF can reduce the gap at CER 10^{-6} between the DSU and the random-coding union (RCU) bounds from over 1 dB for the inner code alone to 0.23 dB for the concatenated code. The DSU bounds can also characterize puncturing that mitigates the rate overhead of the ELF while maintaining the DSU-to-RCU gap. List Viterbi decoding guided by the ELF achieves maximum likelihood (ML) decoding of the concatenated code with a sufficiently large list size. The rate-$K/(K+m)$ ELF outer code reduces rate and list decoding increases decoder complexity. As SNR increases, the average list size converges to 1 and average complexity is similar to Viterbi decoding on the trellis of the inner code. For rare large-magnitude noise events, which occur less often than the FER of the inner code, a deep search in the list finds the ML codeword.

***Index Terms*—Expurgation, Tail-biting Convolutional Codes, Convolutional Codes, Cyclic Redundancy Code, List Viterbi Decoding, List Decoding**

I. Introduction

Expurgating a code decreases the number of message symbols without changing the length [1]. Expurgation strengthens a code by removing weaker codewords at the expense of a slight reduction in rate. For example, in his proof of channel capacity, Gallager [2] removes the half of the randomly selected codewords for which error probability is largest to bound maximum rather than average probability of error. For a linear code, where the minimum distance $d_{\min}$ is the minimum weight of a nonzero codeword [1], expurgation that removes the lowest-weight codewords will increase $d_{\min}$ and thus reduce the probability of a codeword error.

Practical expurgation requires a function, i.e. an outer code, that selects which codewords to remove. This paper develops the paradigm of using an expurgating linear function (ELF) to remove the lowest weight codewords of a linear code or, more generally, to remove codewords so as to minimize a union bound on codeword error rate (CER). The improved performance comes at the expense of a reduction in rate by the rate-$K/(K+m)$ ELF outer code, but well-designed ELFs move CER performance closer to an appropriately rate-adjusted random coding union (RCU) bound [3], [4].

Lou *et al.* in [5] significantly improved error detection performance over traditional cyclic redundancy codes (CRCs) used with zero-terminated convolutional codes by specifically designing CRCs that remove low-weight codewords and thus reduce the undetected error rate. Subsequent papers [6]–[17] used this approach to improve the minimum distance or CER performance of the overall concatenated code, which is decoded using list decoding. These subsequent papers characterized the new designs as CRCs, but we call them ELFs since they are not constrained to be a cyclic code of any particular length and their primary function is expurgation rather than error detection.

For a convolutional inner code with an ELF as the outer code, ML decoding is achieved by serial or parallel list Viterbi decoding with a sufficiently large list size [18]. As observed in [10], as SNR increases, the expected list size converges to one. The average list size is often small at the desired CER.

BCH codes [7] and legacy codes that include CRCs can be reconsidered as ELF codes and decoded using list decoding to decrease CER. Schiavone *et al.* [19] provide a compelling example. The ELF paradigm applies to any inner code. Building on [20], ELFs designed to maximize the minimum distance of the expurgated polar code are described in [11], [21]. An ELF for a trellis code appears in [17]. This paper focuses on ELFs concatenated with tail-biting convolutional codes (TBCCs) [22] as an example.

A. Contributions

This paper presents generating function techniques for distance spectrum union (DSU) upper bounds on the CER of TBCCs concatenated with ELFs under maximum-likelihood (ML) decoding. These bounds are extended to include puncturing for rate compatibility. Such bounds can characterize the CER of every ELF for a given redundancy m and can be used to select the ELF (and the puncturing) that minimizes CER. Alternatively, this paper provides a sieve approach that uses list decoding from the zero-message codeword to identify the ELF that maximizes the minimum distance of the expurgated

This research is supported by National Science Foundation (NSF) grant CCF-2008918. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect views of NSF.

code. The DSU bounds show that an ELF can improve the gap between the DSU and RCU bounds from over 1 dB for the inner code alone to 0.23 dB for the concatenated code.

B. Organization

Sec. II presents (DSU) upper bounds on the CER of convolutional inner codes concatenated with ELF outer codes with or without puncturing. Sec. III presents a sieve approach to identify the ELF that maximizes the minimum distance of the expurgated code. As an example, the best ELFs for expurgating a (152, 76) block code created from a $\nu = 8$ tail-biting convolutional code are identified for ELF redundancies of $0 \leq m \leq 12$. Sec. IV uses the tools of Sec. II and Sec. III to design an example punctured concatenation of an ELF and a tail-biting convolutional code. Sec. V concludes the paper.

II. Distance Spectrum Union Bounds

Generating functions provide upper bounds on the bit error rate [23] and the CER [5], [13] of convolutional codes. This section reviews the generating function approach to computing distance spectrum union (DSU) bounds on the CER of block codes constructed using convolutional codes and describes how such bounds may be applied to zero-terminated and tail-biting convolutional codes with ELFs and with puncturing.

A. DSU Bounds for Zero Termination and Tail Biting

Consider an (N, K) binary block code transmitted over the binary input AWGN channel where $+\sqrt{E_s}$ is transmitted for binary 0 and $-\sqrt{E_s}$ is transmitted for binary 1. Let A_w be the number of codewords with Hamming weight w. A DSU bound on codeword error rate P_{cw} is shown below:

$$P_{cw} \leq \sum_{w=d_{\text{min}}}^{N} A_w Q\left(\sqrt{\frac{wE_s}{\sigma^2}}\right). \tag{1}$$

Define the weight enumerator polynomial as

$$A(w) = \sum_{w=d_{\text{min}}}^{N} A_w W^w. \tag{2}$$

Using the bound $Q(\sqrt{x+y}) \leq Q(\sqrt{x}e^{-y/2})$ [23], a slightly looser bound than (1) may be computed from the weight enumerator polynomial as follows:

$$P_{cw} \leq Q\left(\sqrt{\frac{d_{\text{min}}E_s}{\sigma^2}}\right) e^{\frac{d_{\text{min}}E_s}{2\sigma^2}} A\left(e^{-\frac{E_s}{2\sigma^2}}\right). \tag{3}$$

A transition matrix $T(W)$ facilitates computation of $A(W)$ for either a zero-terminated or tail-biting convolutional code. For the example of a rate $1/n$ convolutional code with ν memory elements, $T(W)$ is an $s \times s$ matrix, where $s = 2^\nu$.

The entry of $T(W)$ at row j and column i represents the transition from state i to state j in the convolutional encoder. If there is no transition from state i to state j, then the entry is zero. Otherwise, the entry is $W^{w_{i,j}}$ where $w_{i,j}$ is the Hamming weight of the n-bit symbol transmitted for that state transition.

Define the i^{th} basic row vector e_i as a length-s vector of all-zeros except a one in position i. We will index the entries

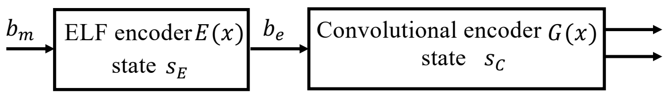

Fig. 1. Convolutional encoder $G(x)$ with an ELF $E(x)$ as an outer code.

in both $T(W)$ and e_i from zero to $s-1$. For example, with $\nu = 2$ and thus $s = 4$, $e_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}$.

For an $(N = n(K+\nu), K)$ block code implemented by sending K information bits through a ν-state, rate-$1/n$, zero-terminated convolutional code,

$$A(W) = e_0 T(W)^{K+\nu} e_0^T - 1. \tag{4}$$

Similarly, for an $(N = nK, K)$ block code implemented by sending K information bits through a 2^ν-state tail-biting convolutional code,

$$A(W) = \sum_{i=0}^{2^\nu - 1} e_i T(W)^K e_i^T - 1. \tag{5}$$

B. DSU Bound for a Convolutional Code with an ELF

A degree-m ELF $E(x)$ can be concatenated with a zero-terminated rate-$1/n$ convolutional code with encoder polynomial matrix $G(x)$. The m ELF redundancy bits reduce the rate from $\frac{K}{(K+\nu)n}$ to $\frac{K}{(K+\nu+m)n}$. Eq. (4) can be applied to the $T(W)$ with $s = 2^{(\nu+m)}$ for the zero-terminated convolutional code $E(x)G(x)$ so that

$$A(W) = e_0 T(W)^{K+\nu+m} e_0^T - 1. \tag{6}$$

For the case of an ELF $E(x)$ concatenated with a rate-$1/n$ tail-biting convolutional code, the rate reduction is from $\frac{K}{Kn}$ to $\frac{K}{(K+m)n}$. To derive the DSU bound, separately consider the state s_E of the ELF encoder in Fig. 1 and the state s_C of the tail-biting convolutional encoder in Fig. 1. Note that

$$0 \leq s_E \leq 2^m - 1 \tag{7}$$

$$0 \leq s_C \leq 2^\nu - 1 \tag{8}$$

Define the overall encoder state of the concatenated code as $s = s_C + 2^\nu \times s_E$. To compute an entry in $T(W)$, the following computations are performed (referencing Fig. 1):
1) The message input bit b_m is processed by the ELF encoder with the origin ELF state $s_E^{(o)}$ to compute the destination ELF state $s_E^{(d)}$ and ELF output bit b_e. 2) The ELF output bit b_e is the input to the convolutional encoder which updates its origin state $s_C^{(o)}$ to the destination state $s_C^{(d)}$ and produces an n-bit output with Hamming weight $w_{i,j}$. 3) The weight term $W^{w_{i,j}}$ is written to $T(W)$ at row j and column i where $j = s_C^{(d)} + 2^\nu \times s_E^{(d)}$ and $i = s_C^{(o)} + 2^\nu \times s_E^{(o)}$. Using this $T(W)$, the weight enumerator polynomial for an $(N = 2 \times (K+m), K)$ block code implemented by sending K information bits through an outer ELF code with 2^m states and encoding the ELF output with a 2^ν-state tail-biting convolutional code is

$$A(W) = \sum_{i=0}^{2^\nu - 1} e_i T(W)^{K+m} e_i^T - 1. \tag{9}$$

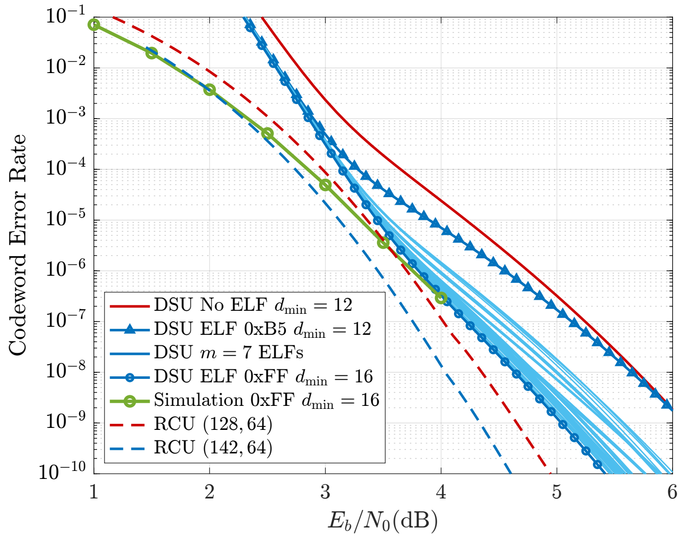

Fig. 2. DSU bounds for the $\nu = 8$ tail-biting convolutional code with $K = 64$ message bits with no ELF (red) and with each possible $m = 7$ ELF (blue). Also shown is a simulation of list Viterbi decoding of the best ELF 0xFF (green) and, for reference, the (142,64) and (128,64) RCU bounds (dashed).

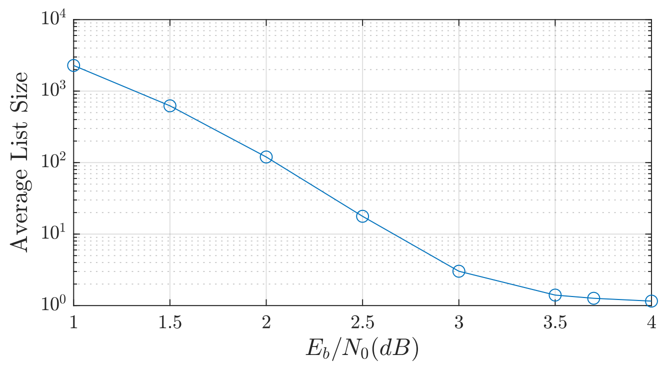

Fig. 3. Average list size vs. E_b/N_0 for the list Viterbi decoding simulation of a $\nu = 8$ TBCC concatenated with ELF 0xFF shown in Fig.2

In (9), $T(W)$ is an $2^{m+\nu} \times 2^{m+\nu}$ matrix, but the only paths that contribute to $A(W)$ are the 2^ν paths that start and end at the same value of s_C and that start and end with $s_E = 0$.

Fig. 2 shows the DSU bounds computed according to (9) and (3) for all of the 64 possible (142, 64) codes resulting from an $m = 7$ ELF concatenated with the $\nu = 8$ tail-biting convolutional code (561,753), where 561 is octal for $1 + x^4 + x^5 + x^6 + x^8$. The RCU bound for (142,64) codes is shown for comparison. For reference, the DSU bound for the (128,64) code that results from using the tail-biting convolutional code without an ELF and its corresponding RCU bound are also shown. The CER curve from simulating list decoding of the tail-biting code used with the best ELF 0xFF shows that this DSU bound is tight for CER $\leq 10^{-6}$.

Fig. 2 shows how the DSU bound on CER varies with the choice of ELF. The best ELF is 0xFF whose DSU bound is 0.35 dB from the (142,64) RCU bound at CER=10^{-6}. The worst ELF is 0xB5 which is 1.10 dB from the RCU bound at CER=10^{-6}. The original (128,64) TBCC is 1.05 dB from the (128,64) RCU bound. Thus, in this case a well-designed 7-bit ELF reduced the gap from the RCU bound by 0.65 dB while the poorest choice had a slightly larger gap.

Fig 3 shows average list size as a function of E_b/N_0 for the list decoding simulation shown in Fig. 2. The maximum list size was set at 2^{20}, but the average list size is 1.26 at E_b/N_0=3.7 dB, where the CER is 1.1×10^{-6}. Thus the average complexity is similar to regular Viterbi decoding on a 256-state trellis but the gap to the RCU bound is only 0.35 dB.

C. DSU Bound for Punctured Convolutional Code with ELF

This section extends the DSU bounding technique to handle puncturing. The techniques below can be applied to any puncturing scheme. For simplicity of exposition, our description only considers puncturing at most one of the n bits in each convolutional encoder symbol. Let p be the puncturing index with range $0 \leq p \leq n$, which indicates either that no bit is punctured ($p = 0$) or which bit is punctured $1 \leq p \leq n$.

Let $T_p(W)$ be the transition matrix of a trellis stage with puncturing according to puncturing index p. The matrix $T_0(W)$ is the same as the $T(W)$ discussed above. The matrices $T_p(W)$ for $1 \leq p \leq n$ have the power of W reduced by one for the entries where the p^{th} output bit, i.e. the punctured bit, is a 1. With this description of punctured transition matrices $T_p(W)$, let p_i be the puncturing index that describes the puncturing of the i^{th} trellis stage. The weight enumerator polynomial for the block code implemented by sending K information bits through an outer ELF code with 2^m states and then a 2^ν-state tail-biting convolutional code and then puncturing some number of bits is expressed as follows:

$$A(W) = \sum_{i=0}^{2^\nu-1} e_i \prod_{i=1}^{K+m} T_{p_i}(W) e_i^T - 1. \tag{10}$$

Often, puncturing is designed according to a periodic pattern. If the period includes q trellis stages, then we can define a transition matrix $T_\pi(W)$ for the puncturing period as follows:

$$T_\pi(W) = \prod_{i=1}^{q} T_{p_i}(W)\,. \tag{11}$$

If the $K+m$ trellis stages are an integer number of puncturing periods, the weight enumerator of the punctured tail-biting convolutional code with an ELF can be expressed as

$$A(W) = \sum_{i=0}^{2^\nu-1} e_i T_\pi(W)^{(K+m)/q} e_i^T - 1. \tag{12}$$

III. A List Decoding Sieve to find the best ELF

This section provides a list decoding sieve method to find the ELFs that maximize the $d_{\min}$ of the concatenated code. This approach is computationally more efficient than the error event construction method of Yang [9]. The sieve performs serial list Viterbi decoding [18] with the (noiseless) all-zeros codeword as the received word. Codewords join the list in order of increasing Hamming weight.

TABLE I
BEST ELFS $E(x)$ FOR $m = 0$ TO $m = 12$ REDUNDANCY BITS THAT MAXIMIZE MINIMUM DISTANCE FOR THE $\nu = 8$ TBCC (561,753) FOR $K = 64$ MESSAGE BITS, $N = 2 \times (64 + m)$ TRANSMITTED BITS AND FOR $N = 2 \times 76$ TRANSMITTED BITS, $K = 76 - m$ MESSAGE BITS.

	$K = 64, N = 2 \times (64 + m)$			$N = 2 \times 76, K = 76 - m$		
m	$E(x)$	$d_{\min}$	$A_{d_{\min}}$	$E(x)$	$d_{\min}$	$A_{d_{\min}}$
0	0x1	12	704	0x1	12	836
1	0x3	12	260	0x3	12	304
2	0x5	12	66	0x5	12	76
3	0xF	12	4	0xF	14	380
4	0x11	14	68	0x11	14	76
5	0x33	14	11	0x33	14	4
6	0x7F	16	210	0x55	14	2
7	0xFF	16	86	0x81	16	24
8	0x1AB	18	360	0x195	16	6
9	0x301	18	146	0x325	18	297
10	0x4F5	18	17	0x53D	18	21
11	0x9AF	20	300	0xE0D	18	2
12	0x1565	20	47	0x1565	20	47

TABLE II
EXPURGATED DISTANCE SPECTRA FOR $m = 0$ (NO EXPURGATION) TO $m = 12$ FOR THE $\nu = 8$ (N=152,K=76) TAIL-BITING CONVOLUTIONAL MOTHER CODE DESCRIBED BY POLYNOMIAL (561,753).

		Expurgated Distance Spectrum for $w \leq 20$				
m	$E(x)$	A_{12}	A_{14}	A_{16}	A_{18}	A_{20}
0	0x1	836	3800	21736	123880	732564
1	0x3	304	1900	11324	61788	367764
2	0x5	76	988	5776	32300	177840
3	0xF	0	380	3344	15656	90060
4	0x11	0	76	1824	8056	43320
5	0x33	0	4	752	4040	22854
6	0x55	0	2	214	2210	11569
7	0x81	0	0	24	1341	5910
8	0x195	0	0	6	461	2932
9	0x325	0	0	0	297	1449
10	0x53D	0	0	0	21	742
11	0xE0D	0	0	0	2	393
12	0x1565	0	0	0	0	47

To find the best ELF polynomial $E(x)$ of degree m, codewords are grouped into sets S_{w_i} according to their Hamming weight w_i. The first set has weight w_1 equal to the $d_{\min}$ of the inner code, and for each i, $w_i > w_{i-1}$. For each set S_{w_i}, the sieve method removes from contention all polynomials $E(x)$ that divide any message polynomial $m(x)$ that produces a codeword in S_{w_i}. The remaining ELF polynomials correspond to concatenated codes with $d_{\min} > w_i$. This continues until a weight w^* is reached that would cause all the remaining ELF polynomials to be removed from contention. From among the remaining polynomials $E(x)$ at weight w^*, select the $E(x)$ that achieves w^* with the smallest number of neighbors.

Table I shows ELFs found by the list decoding sieve for TBCCs, e.g., ELF 0x301 indicates $E(x) = 1 + x^8 + x^9$. The left half of the table holds K constant at 64 bits while the right half holds N constant at 152 bits. ELFs for $0 \leq m \leq 12$ are designed using the sieve approach. The $K = 64$ results for $3 \leq m \leq 10$ match the ELFs reported in [10]. When codeword length is held constant, all the ELFs are expurgating the same mother code. Table II shows how the the low-weight distance spectrum of the (152,76) mother code thins out as m increases.

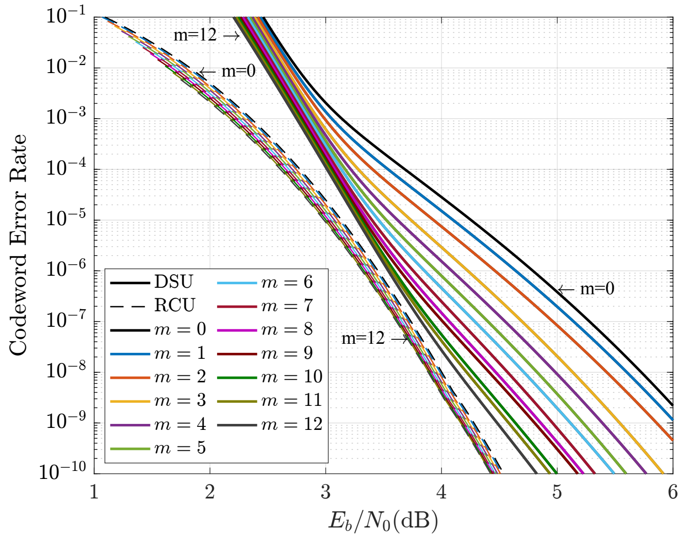

Fig. 4. DSU and RCU bounds for ELF codes of Table II.

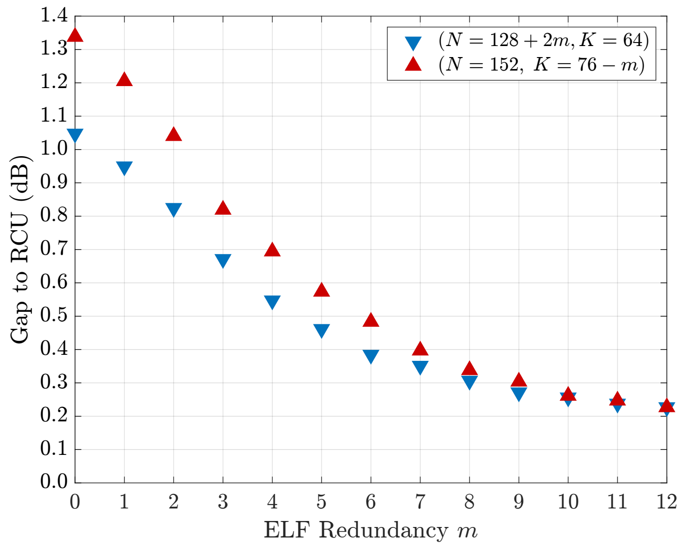

Fig. 5. Gap at 10^{-6} between DSU and RCU bounds vs. m for Table I ELFs.

Fig. 4 uses (9) and (3) to compute the DSU bounds for the expurgated tail-biting convolutional codes of Table II. The corresponding RCU bounds are shown for reference. As m increases the DSU bound on CER performance steadily improves. Meanwhile the slight rate reduction does not significantly improve the CER performance of the RCU bound. As a result, the gap between the DSU and RCU bounds decreases as m increases. Fig. 5 further explores how ELFs reduce the gap between DSU and RCU bounds by showing the gap between DSU and RCU bounds at CER $= 10^{-6}$ vs. m for all the ELFs in Table I. We can see that in both cases the gap decreases as m increases culminating in a gap of 0.227 dB for the $m = 12$ ELF. For the $m = 12$ case, the two codes are actually identical (152, 64) codes. Recall from Fig. 3 that this increase in performance requires a minimal increase in average complexity as described in, e.g, [8], [10].

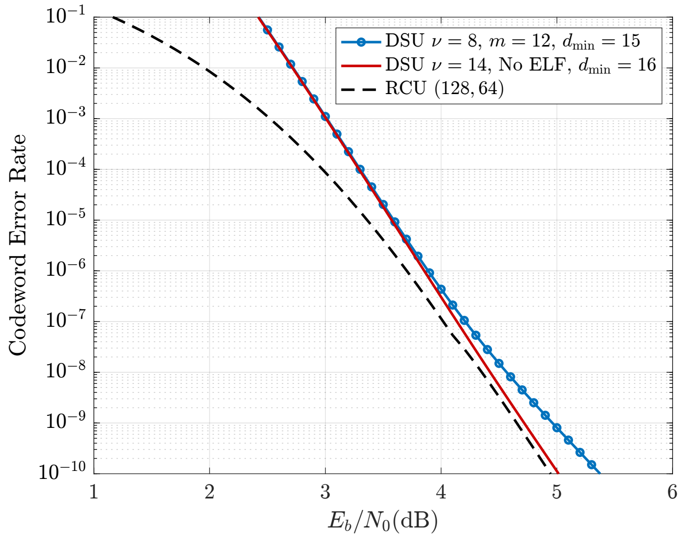

Fig. 6. DSU bounds for two (128,64) codes and the (128,64) RCU bound. One code is the standard $\nu = 14$ tail-biting convolutional code (75063,56711) with no ELF and no puncturing. The other is the $\nu = 8$ tail-biting convolutional code (561,753) with ELF 0x1565 from Tables I and II with 24 bits punctured.

IV. A Puncturing Example: Rate-1/2 $K = 64$

This section applies Sec. II-C to explore a (128, 64) code created by puncturing 24 bits from the (152,64) block code formed by concatenating the $m = 12$ ELF 0x1565 from Tables I and II with the $\nu = 8$ tail-biting convolutional code. This example uses a periodic puncturing pattern with period $q = 19$, where six bits are punctured from each of the four periods that comprise the $K + m = 76$ trellis stages. The 19 puncturing indices for this periodic puncturing pattern are as follows:

$$[0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 2\ 0\ 1\ 0\ 0\ 2\ 0\ 0\ 2]$$

where puncturing index $p_i = 0$ indicates no puncturing, $p_i = 1$ indicates puncturing the output of the convolutional encoder polynomial 561 and $p_i = 2$ indicates puncturing the output of the convolutional encoder polynomial 753. We did not perform a fully exhaustive search even of puncturing patterns with period $q = 19$, but this pattern gives reasonable performance.

Fig. 6 shows the DSU bound for the $\nu = 8$ tail-biting convolutional code (561,753) with ELF 0x1565 and 24 bits punctured as described above. Also shown for comparison are the (128,64) RCU bound and the DSU bound for the standard $\nu = 14$ tail-biting convolutional code (75063,56711) with no ELF and no puncturing. The $\nu = 14$ code has DSU to RCU gap of 0.15 dB at CER 10^{-6}. The $\nu = 8$ solution has a gap of 0.18 dB at CER 10^{-6}. At an operating point of CER $= 10^{-6}$ the $\nu = 8$ solution requires less average decoder complexity.

V. Conclusions

For short block lengths, expurgating linear functions (ELFs) transform a good inner code into a great concatenated code with a minimal increase in average complexity. This paper presented DSU bounding techniques that allow tight bounds on codeword error rate for ELF codes with and without puncturing and a sieve method for finding good ELFs efficiently.

References

[1] R. E. Blahut, *Algebraic Codes for Data Transmission*. Cambridge University Press, 2003.

[2] R. Gallager, “A simple derivation of the coding theorem and some applications,” *IEEE Trans. on Information Theory*, vol. 11, no. 1, pp. 3–18, 1965.

[3] Y. Polyanskiy, H. V. Poor, and S. Verdu, “Channel coding rate in the finite blocklength regime,” *IEEE Trans. on Information Theory*, vol. 56, no. 5, pp. 2307–2359, May 2010.

[4] J. Font-Segura, G. Vazquez-Vilar, A. Martinez, A. Guillén i Fàbregas, and A. Lancho, “Saddlepoint approximations of lower and upper bounds to the error probability in channel coding,” in *2018 52nd Annual Conf. on Information Sciences and Systems (CISS)*, 2018, pp. 1–6.

[5] C.-Y. Lou, B. Daneshrad, and R. D. Wesel, “Convolutional-code-specific CRC code design,” *IEEE Trans. on Communications*, vol. 63, no. 10, pp. 3459–3470, 2015.

[6] H. Yang, S. V. S. Ranganathan, and R. D. Wesel, “Serial list Viterbi decoding with CRC: Managing errors, erasures, and complexity,” in *2018 IEEE Global Comm. Conf. (GLOBECOM)*, 2018, pp. 1–6.

[7] H. Yang, E. Liang, H. Yao, A. Vardy, D. Divsalar, and R. D. Wesel, “A list-decoding approach to low-complexity soft maximum-likelihood decoding of cyclic codes,” in *2019 IEEE Global Comm. Conf. (GLOBECOM)*, 2019, pp. 1–6.

[8] E. Liang, H. Yang, D. Divsalar, and R. D. Wesel, “List-decoded tail-biting convolutional codes with distance-spectrum optimal CRCs for 5G,” in *2019 IEEE Global Comm. Conf. (GLOBECOM)*, 2019, pp. 1–6.

[9] H. Yang, L. Wang, V. Lao, and R. D. Wesel, “An efficient algorithm for designing optimal CRCs for tail-biting convolutional codes,” in *2020 IEEE Int. Sym. Inf. Theory (ISIT)*, June 2020, pp. 1–6.

[10] H. Yang, E. Liang, M. Pan, and R. D. Wesel, “CRC-aided list decoding of convolutional codes in the short blocklength regime,” *IEEE Trans. on Information Theory*, vol. 68, no. 6, pp. 3744–3766, 2022.

[11] J. King, A. Kwon, H. Yang, W. Ryan, and R. D. Wesel, “CRC-aided list decoding of convolutional and polar codes for short messages in 5G,” in *ICC 2022 - IEEE Int. Conf. on Comm.*, 2022, pp. 92–97.

[12] J. King, W. Ryan, and R. D. Wesel, “CRC-aided short convolutional codes and RCU bounds for orthogonal signaling,” in *GLOBECOM 2022 - 2022 IEEE Global Comm. Conf.*, 2022, pp. 4256–4261.

[13] D. Song, F. Areces, L. Wang, and R. Wesel, “Shaped TCM with list decoding that exceeds the RCU bound by optimizing a union bound on fer,” in *GLOBECOM 2022 - 2022 IEEE Global Comm. Conf.*, 2022, pp. 4262–4267.

[14] J. King, “CRC-aided list decoding of short convolutional and polar codes for binary and nonbinary signaling,” Master’s thesis, University of California, Los Angeles (UCLA), 2022.

[15] W. Sui, H. Yang, B. Towell, A. Asmani, and R. D. Wesel, “High-rate convolutional codes with CRC-aided list decoding for short block-lengths,” in *ICC 2022 - IEEE Int. Conf. on Comm.*, 2022, pp. 98–103.

[16] L. Wang, D. Song, F. Areces, and R. D. Wesel, “Achieving short-blocklength RCU bound via CRC list decoding of TCM with probabilistic shaping,” in *ICC 2022 - IEEE Int. Conf. on Comm.*, 2022, pp. 2906–2911.

[17] L. Wang, D. Song, F. Areces, T. Wiegart, and R. D. Wesel, “Probabilistic shaping for trellis-coded modulation with CRC-aided list decoding,” *IEEE Trans. on Communications*, vol. 71, no. 3, pp. 1271–1283, 2023.

[18] N. Seshadri and C. E. W. Sundberg, “List Viterbi decoding algorithms with applications,” *IEEE Trans. on Communications*, vol. 42, no. 234, pp. 313–323, Feb. 1994.

[19] R. Schiavone, R. Garello, and G. Liva, “Application of list Viterbi algorithms to improve the performance in space missions using convolutional codes,” in *2022 9th Int. Workshop on Tracking, Telemetry and Command Systems for Space Applications (TTC)*, 2022, pp. 1–8.

[20] I. Tal and A. Vardy, “List decoding of polar codes,” *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2213–2226, 2015.

[21] J. King, H. Yao, W. Ryan, and R. D. Wesel, “Design, performance, and complexity of CRC-aided list decoding of convolutional and polar codes for short messages.” [Online]. Available: https://arxiv.org/abs/2302.07513

[22] H. Ma and J. Wolf, “On tail biting convolutional codes,” *IEEE Trans. on Communications*, vol. 34, no. 2, pp. 104–111, February 1986.

[23] A. Viterbi, “Convolutional codes and their performance in communication systems,” *IEEE Trans. on Communication Technology*, vol. 19, no. 5, pp. 751–772, 1971.