List-Decoded Tail-Biting Convolutional Codes with Distance-Spectrum Optimal CRCs for 5G

Ethan Liang*, Hengjie Yang*, Dariush Divsalar†, and Richard D. Wesel*
*Department of Electrical and Computer Engineering, University of California, Los Angeles, Los Angeles, CA 90095, USA
†Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109, USA
Email: emliang@ucla.edu, hengjie.yang@ucla.edu, Dariush.Divsalar@jpl.nasa.gov, wesel@ucla.edu

***Abstract*—This paper uses convolutional codes (CCs) with distance-spectrum optimal (DSO) cyclic redundancy checks (CRCs) and the serial list Viterbi algorithm (S-LVA) to approach the random coding union (RCU) bound with low decoding complexity at the target FER. We show, for example, that a 64-state zero-terminated CC with a DSO CRC can achieve performance within 0.5 dB of the RCU bound for information blocklength $k = 64$ at FER of 10^{-3}. We also show that a tail-biting CC with a DSO CRC can achieve even better performance, within 0.05 dB of the RCU bound at FER of 10^{-4} for a 256-state CC with $k = 64$. This paper provides analysis of decoding complexity, which for S-LVA depends on the expected list size. We show that if the target FER is low enough, the expected list size approaches one so that the average complexity of S-LVA approaches that of standard soft Viterbi on the CC, i.e., with no list decoding. We also provide DSO CRCs for CCs with $k = 64$ and rates of $1/2, 1/3, 1/6$ and $1/12$ for the 5G new radio control channel and compare their performance with polar codes.**

I. Introduction

For 3GPP LTE [1] a 64-state ($v = 6$), rate-$1/3$ tail-biting convolutional code (CC) supports various rates with puncturing or repetition. Throughout this paper, v indicates the number of memory elements in the convolutional encoder and m indicates the degree of a cyclic redundancy check (CRC). Tail-biting [2] avoids the overhead of zero-termination.

For 5G, polar coding [3] has received attention as a solution for short-blocklength communication. Polar codes with standard successive cancellation decoding exhibit poor performance at short blocklengths, but list decoding significantly improves their performance [4]. The successive cancellation list (SCL) decoder [4] maintains a list of L polar codewords and selects the most likely codeword that passes a CRC.

A technical study group report [5] generalized the SCL approach with a new class of polar codes for 5G known as parity-check polar codes. These codes outperform the previous CRC-aided polar codes [6]. Parity-check polar codes also outperform the 3G tail-biting CCs at rates $1/2, 1/3, 1/6$ and $1/12$ for information blocklength $k = 64$, as shown in [7], but the 3G codes did not have the benefit of list decoding.

CCs are good short-blocklength codes on their own. As shown by Coskun *et al.* [8], a tail-biting CC with $v = 14$ decoded using the wrap-around Viterbi algorithm (WAVA) can *achieve* the random coding union (RCU) bound of Polyanskiy [9] at FER of 10^{-3}, but the complexity is significant. List decoding significantly improves the performance of CCs, as shown in [10] and [8]. Coskun *et al.* [8] demonstrated performance within 0.6 dB of the RCU bound at FER of 10^{-3} using a tail-biting CC with $v = 11$ and a standard CRC of $m = 16$ and a soft decoder that flips unreliable bits. In [10], distance-spectrum-optimal (DSO) CRCs designed according to Lou *et al.* [11] are concatenated with zero-terminated CCs and decoded using the serial list Viterbi algorithm (S-LVA). [12]–[16]. DSO CRCs significantly improve performance as compared to off-the-shelf CRCs from, e.g., [17].

This research is supported in part by National Science Foundation (NSF) grant CCF-1618272 and a grant from the Physical Optics Corporation (POC). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF or POC. Research was carried out in part at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with NASA.

A. Contributions and Organization

This paper demonstrates that short-blocklength CCs concatenated with a DSO CRC and decoded by S-LVA approach the RCU bound with low average complexity.

For zero-terminated CCs, Sec. II develops metrics validated by actual decoder run-time to compare the complexity of standard soft Viterbi (SSV) to that of S-LVA. For example, these metrics show that for FERs below 10^{-2}, the average complexity of using S-LVA for a zero-terminated CC with 2^v states and a CRC of $m = 10$ is within a factor of 2 of the complexity of using SSV for the 2^v-state zero-terminated CC. These metrics show that a CC with a DSO CRC for $k = 64$ can approach the RCU bound with significantly less complexity than [8]. For rate-$1/2$ zero-terminated CCs with $v \geq 5$, list decoding using a DSO CRC of $m = 10$ achieves performance within 0.5 dB of the RCU bound for FER of 10^{-3} with complexity similar to SSV on the 2^v-state trellis.

For tail-biting CCs, Sec. III extends the list decoding of [18] to accommodate a CRC and employs a DSO CRC to approach the RCU bound even more closely. For example, a tail-biting CC with $v = 7$ and DSO optimal CRC of $m = 10$ can perform within 0.05 dB of the RCU bound for FER of 10^{-3}. S-LVA complexity for tail-biting can be significantly higher than for zero-termination, but in both cases average complexity is less than twice that of SSV on the 2^v-state trellis for sufficiently low FER. A good approximation for the expected list size for both zero-termination and tail-biting indicates the CRC size m for which S-LVA complexity is close to that of SSV on the 2^v-state trellis at the target FER.

TABLE I
POPULAR RATE-1/2 CCS AND CORRESPONDING DISTANCE-SPECTRUM-OPTIMAL CRC CODES WITH $k = 64$

v	Conv. Code g_v	Distance-Spectrum-Optimal CRC Gen. Poly. $m = 3$	4	5	6	7	8	9	10
3	$(13, 17)$	9	1B	2D	43	B5	107	313	50B
4	$(27, 31)$	F	15	33	4F	D3	13F	2AD	709
5	$(53, 75)$	9	11	25	49	EF	131	23F	73D
6	$(133, 171)$	F	1B	23	41	8F	113	2EF	629
7	$(247, 371)$	9	13	3F	5B	E9	17F	2A5	61D
8	$(561, 753)$	F	11	33	49	8B	19D	27B	4CF
9	$(1131, 1537)$	D	15	21	51	B7	1D5	20F	50D
10	$(2473, 3217)$	F	13	3D	5B	BB	105	20D	6BB

Section IV provides tail-biting CCs with DSO CRCs for $k = 64$ that outperform the parity-check polar coding results from [5], [7] for rates of $1/2, 1/3, 1/6$, and $1/12$. Section V concludes the paper.

II. ZERO-TERMINATED CC WITH DSO CRC VIA S-LVA

For rate-$1/2$ zero-terminated CCs from [19] with DSO CRCs designed according to [11], this section explores the trade-off between complexity and the gap between the SNR required to achieve a target FER and the SNR of the RCU bound of Polyanskiy *et al.* [9]. SNR is defined as the squared amplitude of a BPSK signal divided by the variance of a one-dimensional, zero-mean Gaussian noise. Table I shows the combinations of rate-$1/2$, zero-terminated CCs and DSO-CRCs, with both m and v ranging from 3 to 10. The DSO CRC can vary as a function of k, and Table I presents DSO CRCs specifically for $k = 64$.

A. Complexity Increase of S-LVA over SSV

There are a variety of implementations of list decoding of CCs as described in, e.g., [15], [20]–[23]. The implementation of S-LVA studied in this paper maintains a list of path metric differences by using a red-black tree as described in [15], which provided the fastest run-time among the data structures that support full floating-point precision.

Numerous papers including [15], [20]–[23] also analyze either the number of bit operations or the asymptotic complexity of the algorithms they present, but the complexity metrics are not directly connected with actual run-time. To explore how the additional complexity of the S-LVA relative to the SSV varies as a function of SNR, this paper develops a complexity expression that closely approximates empirical run-times.

Three components comprise the complexity of S-LVA as shown in (1):

$$C_{\text{S-LVA}} = C_{\text{SSV}} + C_{\text{trace}} + C_{\text{list}} \,. \qquad (1)$$

The first component C_{SSV} is the complexity required to perform the add-compare-select (ACS) operations on the zero-terminated CC trellis and perform the initial traceback associated with SSV:

$$C_{\text{SSV}} = 5(2^v - 1) + 3(k + m - v) \cdot 2^v \qquad (2)$$
$$+ c_1 \cdot [2(k + m + v) + 1.5(k + m)] \,. \qquad (3)$$

The complexity of ACS operations in the SSV algorithm for a zero-terminated CC is represented by (2). Equation (3) approximates the complexity of the traceback operation, assigning 2 units of complexity for accessing the parent node per trellis stage and 1.5 units of complexity per codeword symbol for the CRC check on the decoded message.

The second component, C_{trace}, is the complexity of the traceback operations required by S-LVA. For a zero-terminated CC,

$$C_{\text{trace}} = c_1(\mathbb{E}[L] - 1)\left[2(k + m + v) + 1.5(k + m)\right] , \qquad (4)$$

where L is the random variable describing the number of traceback operations for S-LVA and $\mathbb{E}[L]$ is its expectation.

An ordered list of path metric differences must be maintained to select each candidate path for traceback. The third constituent C_{list} represents the normalized complexity of inserting new elements to maintain the ordered list. Assuming the use of a zero-terminated CC,

$$C_{\text{list}} = c_2 \mathbb{E}[I] \log(\mathbb{E}[I]) \,. \qquad (5)$$

In the above equations, c_1 and c_2 are constants that characterize implementation-specific differences in the implemented complexity of traceback and list insertion (respectively) as compared to the ACS operations of Viterbi decoding. For our implementation we found $c_1 = 1.5$ and $c_2 = 2.2$. $\mathbb{E}[I]$ denotes the expected number of insertions to maintain the sorted list of path metric differences.

The additional complexity of the S-LVA over SSV decoding is completely characterized by the additional tracebacks along the trellis and the maintenance of an ordered list of path metric differences. We define the *normalized complexity* $\bar{C}$ as the complexity divided by the complexity required to perform the SSV, i.e.,

$$\bar{C}_{\text{S-LVA}} = \frac{C_{\text{S-LVA}}}{C_{\text{SSV}}} = 1 + \bar{C}_{\text{trace}} + \bar{C}_{\text{list}} \,. \qquad (6)$$

The normalized complexity provides a measure for the additional complexity of operations associated with the S-LVA relative that of the SSV algorithm.

We recorded the run-time $T_{\text{S-LVA}}$, T_{SSV}, T_{trace}, and T_{list} on an Intel i7-4720HQ using Visual C++. We then divided all of these terms by T_{SSV} to compute a normalized run-time $\bar{T}$. Fig. 1 shows normalized complexity based on equation (6) and normalized run-times. In both cases, the normalization is computed by dividing by the complexity or run-time associated with SSV, i.e. performing all the add-compare-select (ACS) operations on the trellis and a traceback from the state with the best metric. The normalized complexity and normalized run-time curves are indistinguishable. Fig. 1 also shows that the additional complexity of S-LVA is primarily from maintaining an ordered list of path metric differences.

Fig. 2 shows complexity calculated using (1) as a function of SNR for the rate-$1/2$ CC with $v = 7$ in Table I for DSO CRCs from $m = 0$ (no CRC) to $m = 10$. The complexity at target FERs of 10^{-2}, 10^{-3}, and 10^{-4} are indicated, respectively, by squares, diamonds, and stars for

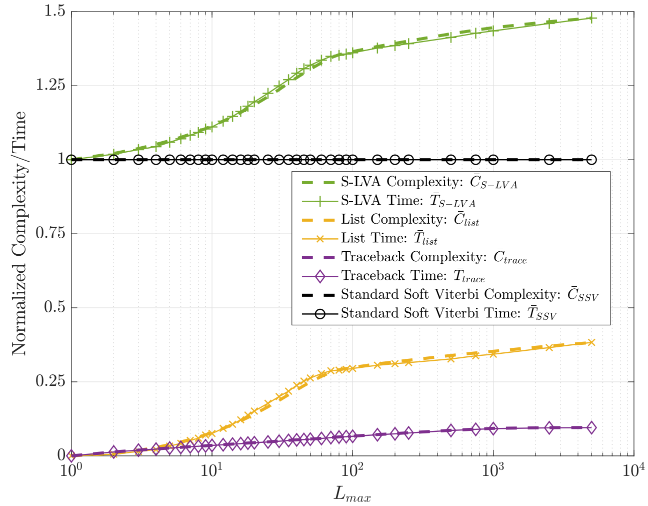

Fig. 1. The complexity of S-LVA with different list sizes for $(27, 31)$ CC, and 0x709 CRC code, with $k = 64$ at SNR $= 2$ dB. All variables are normalized by the time or complexity of the SSV algorithm. In the simulation setting, $c_1 = 1.5$ and $c_2 = 2.2$.

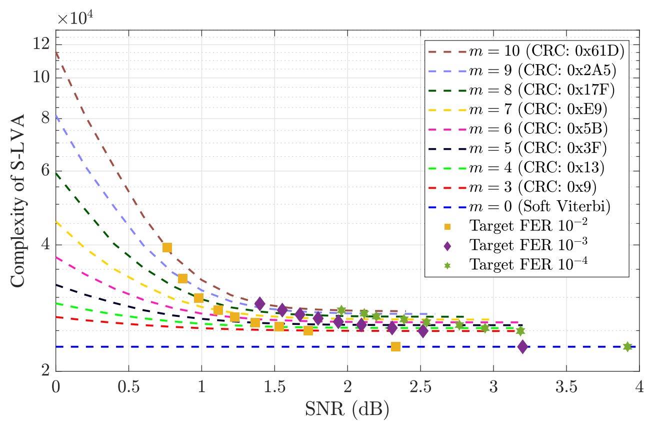

Fig. 2. The decoding complexity vs. SNR for $(247, 371)$ CC with its corresponding distance-spectrum-optimal CRC codes. The CC with no CRC using soft Viterbi decoding is also given as a reference.

each choice of CRC. For each of these three operating points and CRCs, the complexity for S-LVA is within a factor of 2 of the complexity for $m = 0$, i.e., SSV with no list decoding.

B. Comparison with Random Coding Union Bound

To assess performance in the short-blocklength regime, we use the approach of [8] utilizing the RCU bound [9]. We compute the RCU bound using a numerically efficient saddle-point approximation [24]. Fig. 3 shows the gap from the RCU bound on finite blocklength transmission computed as described in [9] as a function of the complexity expression derived in Sec. II-A.

Fig. 3 shows that increasing m can provide significant reduction in the gap from the RCU bound for a relatively small complexity cost. As v becomes larger, the complexity cost for increasing m becomes less significant. For CCs with

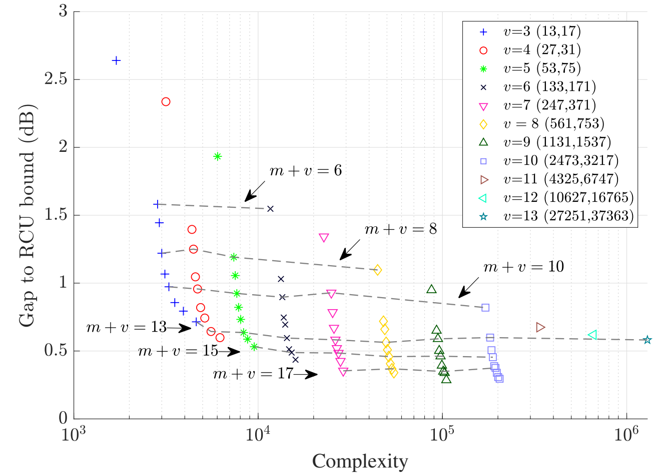

Fig. 3. The SNR gap to RCU bound vs. decoding complexity for various CC-CRC pairs with $k = 64$ and target FER 10^{-3}. Each color corresponds to a specific CC shown in parenthesis. Markers from top to bottom with the same color correspond to soft Viterbi decoding with no CRC at the very top and then $m = 3, 4, \cdots, 10$ distance-spectrum-optimal CRC codes, respectively. CCs with $v = 11, 12, 13$ using soft Viterbi decoding are also provided.

$v \geq 5$ the complexity cost of increasing m from $m = 0$ (no CRC) to $m = 10$ is within a factor of 2. This is consistent with Fig. 1 where the complexity increased by a factor less than 1.5 even for a very large list size.

III. Tail-Biting CC with CRC and S-LVA

A tail-biting CC can be decoded by performing SSV from each initial state and constraining the SSV to terminate in the initial state. Choosing the ending state with the lowest metric selects the overall ML codeword. This has complexity $2^v N_s N_b$ where N_s is the number of trellis stages and N_b is the number of branches in one stage of the trellis.

The wrap-around Viterbi algorithm (WAVA) is an alternative that performs a single SSV in which all initial states are active and initialized with a zero metric. If the lowest-metric final state matches the initial state that results from traceback, WAVA concludes by selecting the corresponding codeword. If not, WAVA "wraps around" by again performing an SSV with the initial states initialized to have the metric of the corresponding final state from the previous SSV. WAVA has not been proven to be ML, but its performance is empirically very close to ML. Its complexity is $\mathbb{E}[M_w] N_s N_b$ where $\mathbb{E}[M_w]$ is the expected number of times SSV is performed. Since $\mathbb{E}[M_w] < 2^v$ at typical target FERs, WAVA typically requires significantly less complexity in practice than performing SSV from each initial state.

A. List decoding for tail-biting CC

List decoding is an alternative to WAVA. An efficient iterative, recursive decoding algorithm for tail-biting CCs is introduced in [18]. We extend the decoding algorithm of [18] to include a CRC by adding an additional step. If a traceback identifies a tail-biting path, the CRC of the corresponding codeword is checked. If the codeword passes the CRC check,

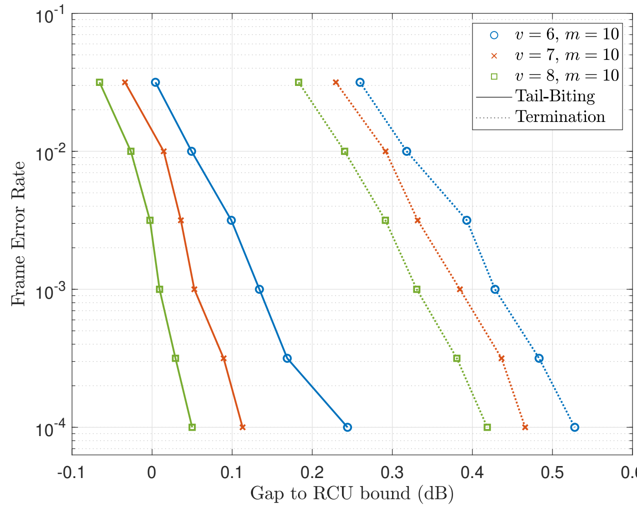

Fig. 4. FER vs. gap from the RCU bound. These rate-$1/2$ CCs with $v = 6, 7, 8$ and a CRC with $m = 10$ are between 0.4 and 0.55 dB from the RCU bound for FER of 10^{-4} when terminated with v zeros. Tail-biting removes the v overhead bits so that the gap form the RCU bound is less than 0.1 dB for $v = 8$.

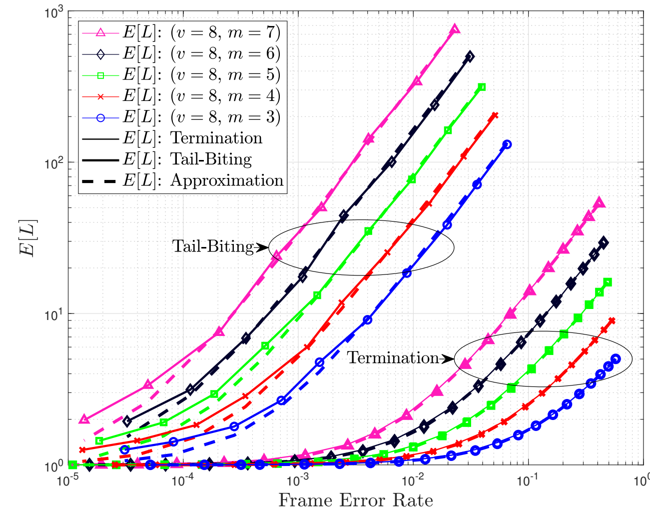

Fig. 5. $\mathbb{E}[L]$ vs. FER for TBCC and zero-terminated CC. The $\mathbb{E}[L]$ for tail-biting CC and zero-terminated CC can be approximated with a function that depends only on FER, v, and m.

the algorithm terminates. If it fails, the algorithm locates the next lowest path metric difference and continues the list decoding process of [18].

The first codeword that satisfies both the tail-biting constraint and the CRC check is the maximum likelihood codeword. This is because the algorithm considers all possible paths, regardless of starting and ending state, in ascending order of total path metric differences. Thus, the first codeword that checks both constraints must be the codeword with the lowest total metric difference from the channel observations.

Figure 4 shows the improvement in decoder performance relative to RCU bound using a DSO CRC with tail-biting rather than zero-termination for the CC. Although tail-biting generally has a higher FER at the same SNR as compared to the zero-terminated CC, tail-biting operates at a higher rate than the zero-terminated CC. Using RCU gap as our metric of relative performance, the FER increase from tail-biting is outweighed by the rate saved through tail-biting.

B. Expected List Size of S-LVA

As defined in Sec. II-A, the random variable L is the "list size" required by S-LVA to find a codeword that passes the CRC for a zero-terminated CC or that both satisfies the tail-biting condition and passes the CRC for tail-biting CC. Defining ϵ as the FER, we propose the following approximations for $\mathbb{E}[L]$ for zero-terminated CC and tail-biting CC for the case where there is no maximum list size, i.e. $L_{\max} = \infty$:

$$\mathbb{E}[L_{\text{ZTCC}}] \approx 1 - \epsilon + 2^{m}\epsilon \tag{7}$$

$$\mathbb{E}[L_{\text{TBCC}}] \approx 1 - \epsilon + 2^{v+m}\epsilon. \tag{8}$$

Figure 5 shows that for a zero-terminated CC with $v = 8$ and $m = 3, 4, \cdots, 7$, (7) is a good approximation for $\mathbb{E}[L]$ over the entire FER range. In the case of a tail-biting CC, Figure 5 also shows that (8) is accurate when $\mathbb{E}[L] > 10$. However, in the case of a tail-biting CC, the approximation falls slightly below $\mathbb{E}[L]$ when $\mathbb{E}[L] < 10$.

Note that (7) and (8) assume that the decoding occurs with no limit on the number of tracebacks. When $L_{\max}$ limits the number of tracebacks such that NACKs are declared, the observed $\mathbb{E}[L]$ can decrease below (7) and (8).

Designers can use (7) and (8) and a target ϵ to choose the CRC size m and to decide between ZTCC and tail-biting CC. Assuming $\epsilon \ll 1$, $\mathbb{E}[L] \approx 2$ can be achieved by selecting m for a ZTCC such that $2^m \approx \epsilon^{-1}$ or choosing m and v such that $2^{m+v} \approx \epsilon^{-1}$ for tail-biting CC. Specifically, by choosing $m + v = 17$, $\mathbb{E}[L] \approx 2$ can be achieved for tail-biting CCs with DSO CRCs for ϵs below 10^{-5}.

IV. Designs for 5G Rates

In this section, we design punctured tail-biting CCs and companion DSO CRCs for $k = 64$ information bits targeting rates $1/2, 1/3, 1/6$, and $1/12$ to compare with parity check polar codes at these rates. For target rates $1/2$ and $1/3$, we use periodic puncturing as in [25] followed by a DSO CRC designed for the punctured CC following [11]. For target rates $1/6$ and $1/12$, the "puncturing" of the CC simply yields a higher-rate un-punctured CC.

Consider a rate-$1/n$ tail-biting CC described by polynomial vector g with v memory elements and a CRC polynomial of degree m. Puncturing $B = nm$ bits compensates for the rate loss introduced by the CRC to obtain an effective rate $R = k/(n(k + m) - B) = 1/n$. For periodic puncturing, as in [25], let T be the puncturing period and t be the number of punctured bits in each period. Thus, $B = \lfloor (k+m)/T \rfloor t$. The puncturing pattern of [25] is described by an $n \times T$ binary matrix $\boldsymbol{P}$ for which $\boldsymbol{P}_{i,j} = 1$ if the j-th bit in each period produced by the i-th column of g is transmitted and $\boldsymbol{P}_{i,j} = 0$ if it is punctured, $1 \le i \le n, 1 \le j \le T$.

For a compact representation exploiting the sparsity of 0 entries in $\boldsymbol{P}$, define the set of zero-entry locations in $\boldsymbol{P}$ as

$$\mathcal{I} = \{b \mid \exists\, i, j, \boldsymbol{P}_{i,j} = 0, b = (j-1)r + i\}\,. \tag{9}$$

The design procedure is as follows: First, we design the periodic puncturing scheme either by exhaustive search or by repetitive construction. Then we find the CRC for the punctured (terminated) CC using Lou *et al.*'s approach [11] which selects the CRC polynomial that maximizes the weight of all possible undetected error events. This is a suboptimal CRC design because it ignores the effect of tail-biting. Incorporating tail-biting effects into the DSO CRC design is a subject for future research.

Table II summarizes our designs. The degree-10 CRC $(2303)_8$ in octal is $p(x) = x^{10} + x^7 + x^6 + x + 1$, and $d_{\min}$ denotes the minimum distance of the punctured CC assuming zero termination. Similarly, d_{CRC} denotes the minimum distance for the concatenated CRC and punctured (terminated) CC. The subsections below describe the specific design procedures for each target rate.

A. Rate-1/2 and Rate-1/3

Consider the rate-1/2 tail-biting CC in which the mother CC is described by the polynomial $g_6 = (133, 171)$ with $v = 6$ memory elements. Following the discussion that concludes Sec. III-B, we choose $m = 11$ for CRC degree since $m + v = 17$ yields complexity close to that of SSV on the mother CC for FERs below 10^{-5}. Similarly, for the mother CC $g_7 = (247, 371)$ with $v = 7$ memory elements, we choose $m = 10$ as the CRC degree.

For $g_6 = (133, 171)$ and $m = 11$, $B = nm = 22$ parity check bits are added as a result of the CRC. Thus, choosing $t = 2$ and solving for T, we have $(k+m)t/B = 75/11 \approx 7$ so that $T = 7$. Then $B = \lfloor (k+m)/T \rfloor t = 20 < 22$. One way to obtain the required 2 bits is to puncture in the last incomplete period according to $\mathcal{I}$ after $\mathcal{I}$ is found.

After determining T and t, we find one optimal puncturing pattern $\mathcal{I} = \{1, 6\}$ which gives a maximum $d_{\min} = 7$ using exhaustive search. After finding the optimal (periodically) punctured terminated CC. We apply Lou *et al.*'s approach to obtain the optimal degree-11 CRC polynomial. When applying Lou *et al.*'s design scheme, the error events considered should be the *punctured* error events. The design process for rate-1/3 is the same as for rate-1/2 and we omit the details.

Figure 6 shows the performance of tail-biting CCs with DSO CRCs relative to that of other rate-1/2 codes with $k = 64$. The code with $v = 7$ and $m = 10$ performs nearly as well as a tail-biting CC with $v = 11$ decoded using WAVA and the code with $v = 8$ and $m = 10$ performs within 0.1 dB of a $v = 14$ tail-biting CC decoded using WAVA at an FER of 10^{-5}. In addition, our method using $v = 7$, $m = 10$ and $v = 8$, $m = 10$ outperforms the iterative bit-flipping decoding method used in [8] despite the code of [8] having a much larger $m = 16$ for its off-the-shelf CRC and a much larger $v = 11$. The tail-biting CCs with DSO CRCs with $v = 7$, $m = 10$ and with $v = 8$, $m = 10$ both outperform the

TABLE II
CODES WITH TAIL-BITING CC AND DSO CRC FOR $k = 64$.
$g_6 = (133, 171)$, $g_7 = (247, 371)$, $g_8 = (561, 753)$,
$f_6 = (133, 171, 165)$, $f_7 = (225, 331, 367)$, $f_8 = (575, 623, 727)$,
$T = 7$ FOR PUNCTURED CODES.

R_{tb}	tail-biting CC		v	$d_{\min}$	m	CRC	d_{CRC}
1/2	g_6	$\mathcal{I} = \{1, 6\}$	6	7	11	6373	15
1/2	g_7	$\mathcal{I} = \{1, 3\}$	7	7	10	2303	15
1/2	g_8	$\mathcal{I} = \{2, 6\}$	8	9	10	2275	16
1/3	f_6	$\mathcal{I} = \{1, 2, 10\}$	6	12	10	3667	24
1/3	f_7	$\mathcal{I} = \{1, 4, 7\}$	7	13	10	2071	25
1/3	f_8	$\mathcal{I} = \{3, 6, 12\}$	8	15	10	2357	26
64/370	(f_6, g_6)		6	25	10	2715	48
64/375	(f_6, g_6)		6	25	11	4517	51
64/370	(f_7, g_7)		6	27	10	3463	53
64/375	(f_7, g_7)		6	27	11	4123	56
64/370	(f_8, g_8)		6	30	10	3155	55
64/375	(f_8, g_8)		6	30	11	7471	58
64/760	(f_6, g_6, f_6, g_6)		6	50	12	15721	108
64/770	(f_6, g_6, f_6, g_6)		6	50	13	25467	110
64/770	(f_7, g_7, f_7, g_7)		6	54	12	14661	116
64/770	(f_7, g_7, f_7, g_7)		6	54	13	35233	122
64/760	(f_7, g_7, f_7, g_7)		6	60	12	11435	120
64/770	(f_7, g_7, f_7, g_7)		6	60	13	23173	126

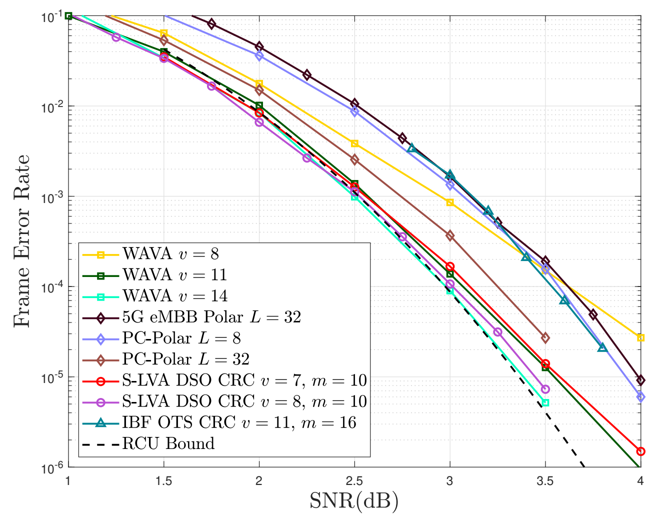

Fig. 6. FER vs. SNR comparison between serial-list Viterbi decoded TBCC with DSO CRC and other candidate codes for URLLC. All simulations are for $k = 64$, $n = 128$, rate=1/2. WAVA and 5G eMBB simulations are located in [26], parity-check polar (PC-Polar) code simulations are located in [7], and iterative bit flipping (IBF) simulations with an off-the-shelf (OTS) CRC are located in [8].

5G eMBB polar and parity-check polar codes with $L = 32$ at rate-1/2.

B. Rate-1/6 and Rate-1/12

The design procedures for rate-1/6 and rate-1/12 codes involves repeating a higher rate CC. The construction scheme combines two low-rate convolutional generator matrices and then deletes columns to leave space for the CRC. For example, the for $v = 6$ case, we first construct a rate-1/6 mother CC by repeating $(133, 171, 165)$ twice and then deleting the last column to obtain the rate 1/5 code $(133, 171, 165, 133, 171)$. We then apply Lou *et al.*'s design method as described in Sec.

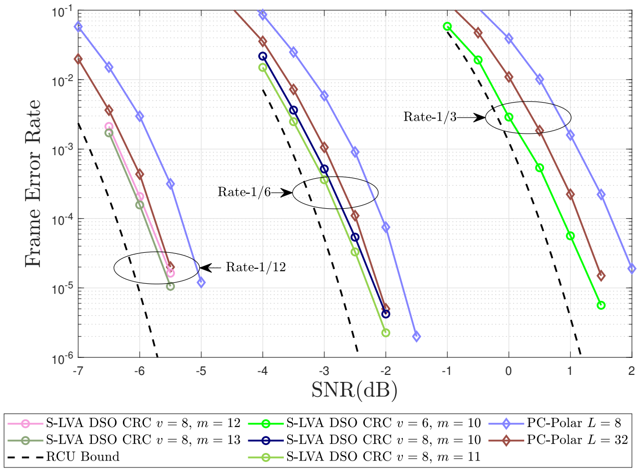

Fig. 7. FER vs. SNR comparison between serial-list decoded TBCC with DSO CRC and other candidate codes for URLLC. The target rates are $1/3, 1/6$ and $1/12$. $k = 64$ is used for all simulations. The PC-Polar simulations are located in [7].

IV-A. The resulting concatenated code is approximately rate-$1/6$ (or $64/384$). In the design process for a rate-$1/12$ code, we repeat the rate-$1/5$ code to obtain the rate-$1/10$ code. The resulting concatenated code is approximately rate-$1/12$ (or $64/768$).

Figure 7 shows the performance of the designed tail-biting CCs with DSO CRCs at rates $1/3$, $1/6$ and $1/12$. At all three rates, our designed codes outperform parity-check polar codes with $L = 32$.

V. Conclusion

This paper develops complexity metrics for S-LVA and uses them to show that the average complexity of using S-LVA for a zero-terminated CC with CRC is within a factor of 2 of SSV for that zero-terminated CC for FERs below 10^{-2}. The paper showed that $k = 64$ CCs with a DSO CRC and S-LVA can achieve FER performance very close to the RCU bound with significantly less complexity than [8]. This paper also provides a good approximation for expected list size for both zero termination and tail-biting. This allows a designer to control complexity by selecting a CRC size m that does not induce a large expected list size. It can also inform the selection between zero termination and tail-biting based on available complexity and desired FER. Finally, the paper shows that tail-biting CCs with DSO CRCs decoded with S-LVA outperform the parity-check polar codes of [7]. A careful complexity comparison between polar approaches and CC with S-LVA approaches remains as future work.

References

[1] "ETSI TS 136.212 V15.3.0 LTE; Evolved universal terrestrial radio access (E-UTRA) multiplexing and channel coding (3GPP TS 36.212 version 15.3.0 release 15)," Jan. 2018.

[2] H. Ma and J. Wolf, "On tail biting convolutional codes," *IEEE Trans. Commun.*, vol. 34, no. 2, pp. 104–111, February 1986.

[3] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, July 2009.

[4] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.

[5] R1-1608862, "Polar code construction for NR," 3GPP TSG RAN WGI Meeting 86, Lisbon, Portugal, November 2016.

[6] H. Zhang, R. Li, J. Wang, S. Dai, G. Zhang, Y. Chen, H. Luo, and J. Wang, "Parity-check polar coding for 5G and beyond," in *IEEE Int. Conf. on Commun. (ICC)*. IEEE, 2018, pp. 1–7.

[7] R1-1611257, "Performance of channel coding scheme for control channel," 3GPP TSG RAN WGI Meeting 87, Reno, Nevada, November 2016.

[8] M. C. Coşkun, G. Durisi, T. Jerkovits, G. Liva, W. Ryan, B. Stein, and F. Steiner, "Efficient error-correcting codes in the short blocklength regime," *Physical Communication*, vol. 34, pp. 66 – 79, 2019.

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

[10] 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 Commun. Conf. (GLOBECOM)*, Dec 2018, pp. 1–6.

[11] C. Y. Lou, B. Daneshrad, and R. D. Wesel, "Convolutional-code-specific CRC code design," *IEEE Trans. Commun.*, vol. 63, no. 10, pp. 3459–3470, Oct. 2015.

[12] F. K. Soong and E. F. Huang, "A tree-trellis based fast search for finding the n-best sentence hypotheses in continuous speech recognition," in *Proc. Int. Conf. Acoust., Speech, and Signal Process. (ICASSP)*, Apr. 1991, pp. 705–708 vol.1.

[13] N. Seshadri and C. E. W. Sundberg, "List Viterbi decoding algorithms with applications," *IEEE Trans. Commun.*, vol. 42, no. 234, pp. 313–323, Feb. 1994.

[14] C. Nill and C. E. W. Sundberg, "List and soft symbol output Viterbi algorithms: Extensions and comparisons," *IEEE Trans. Commun.*, vol. 43, no. 2/3/4, pp. 277–287, Feb. 1995.

[15] M. Roder and R. Hamzaoui, "Fast tree-trellis list Viterbi decoding," *IEEE Trans. Commun.*, vol. 54, no. 3, pp. 453–461, Mar. 2006.

[16] R. Wang, W. Zhao, and G. B. Giannakis, "CRC-assisted error correction in a convolutionally coded system," *IEEE Trans. Commun.*, vol. 56, no. 11, pp. 1807–1815, Nov. 2008.

[17] P. Koopman and T. Chakravarty, "Cyclic redundancy code (CRC) polynomial selection for embedded networks," in *Int. Conf. Dependable Syst. and Networks*, Jun. 2004, pp. 145–154.

[18] P. Shankar, P. N. A. Kumar, K. Sasidharan, B. S. Rajan, and A. S. Madhu, "Efficient convergent maximum likelihood decoding on tail-biting trellises," *CoRR*, vol. abs/cs/0601023, 01 2006.

[19] S. Lin and D. J. Costello, *Error Control Coding: fundamentals and applications*. New Jersey, USA: Pearson Prentice Hall, 2004.

[20] C. Bai, B. Mielczarek, W. A. Krzymien, and I. J. Fair, "Efficient list decoding for parallel concatenated convolutional codes," in *2004 IEEE 15th Int. Symp. on Personal, Indoor and Mobile Radio Communc. (IEEE Cat. No.04TH8754)*, vol. 4, Sep. 2004, pp. 2586–2590 Vol.4.

[21] M. Shirvanimoghaddam, M. S. Mohammadi, R. Abbas, A. Minja, C. Yue, B. Matuz, G. Han, Z. Lin, W. Liu, Y. Li, S. Johnson, and B. Vucetic, "Short block-length codes for ultra-reliable low latency communications," *IEEE Commun. Mag.*, vol. 57, no. 2, pp. 130–137, February 2019.

[22] L. Lijofi, D. Cooper, and B. Canpolat, "A reduced complexity list single-wrong-turn (SWT) Viterbi decoding algorithm," in *2004 IEEE 15th Int. Symp. on Personal, Indoor and Mobile Radio Communc. (IEEE Cat. No.04TH8754)*, vol. 1, Sep. 2004, pp. 274–279 Vol.1.

[23] J. Kim, J. Tak, H. Kwak, and J. No, "A new list decoding algorithm for short-length TBCCs with CRC," *IEEE Access*, vol. 6, pp. 35 105–35 111, 2018.

[24] 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 Annu. Conf. on Inf. Sciences and Syst. (CISS)*, March 2018, pp. 1–6.

[25] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Trans. Commun.*, vol. 36, no. 4, pp. 389–400, Apr 1988.

[26] G. Liva and F. Steiner, "pretty-good-codes.org: Online library of good channel codes," http://pretty-good-codes.org, Oct. 2017.