A List-Decoding Approach to Low-Complexity Soft Maximum-Likelihood Decoding of Cyclic Codes

Hengjie Yang*, Ethan Liang*, Hanwen Yao†, Alexander Vardy†, Dariush Divsalar‡, and Richard D. Wesel*
*University of California, Los Angeles, Los Angeles, CA 90095, USA
†University of California, San Diego, La Jolla, CA 92093, USA
‡Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109, USA
Email: {hengjie.yang, emliang}@ucla.edu, {hwyao, avardy}@ucsd.edu, Dariush.Divsalar@jpl.nasa.gov, wesel@ucla.edu

***Abstract*—This paper provides a reduced-complexity approach to maximum likelihood (ML) decoding of cyclic codes. A cyclic code with generator polynomial $g_{\text{cyclic}}(x)$ may be considered a terminated convolutional code with a nominal rate of 1. The trellis termination redundancy lowers the rate from 1 to the actual rate of the cyclic code. The proposed decoder represents $g_{\text{cyclic}}(x)$ as the product of two polynomials, a convolutional code (CC) polynomial $g_{\text{cc}}(x)$ and a cyclic redundancy check (CRC) polynomial $g_{\text{crc}}(x)$, i.e., $g_{\text{cyclic}}(x) = g_{\text{cc}}(x)g_{\text{crc}}(x)$. This representation facilitates serial list Viterbi algorithm (S-LVA) decoding. Viterbi decoding is performed on the natural trellis for $g_{\text{cc}}(x)$, and $g_{\text{crc}}(x)$ is used as a CRC to determine when the S-LVA should conclude. At typical target frame error rates, the expected list size of S-LVA is small, and the average decoding complexity is dominated by the trellis complexity of $g_{\text{cc}}(x)$ rather than $g_{\text{cyclic}}(x)$. Some high-rate binary Bose-Chaudhuri-Hocquenghem (BCH) examples show that the proposed use of S-LVA via factorization significantly lowers complexity as compared to using the minimum-complexity trellis representation of $g_{\text{cyclic}}(x)$ for soft ML decoding.**

I. Introduction

The Berlekamp-Massey [1], [2] and Euclidean [3] algorithms provide bounded-distance hard decoding for Bose-Chaudhuri-Hocquenghem (BCH) codes [4]. For soft decoding, Guruswami and Sudan developed a ground-breaking (but complex) list decoding approach for Reed-Solomon codes in [5] that identifies all codewords within a defined Hamming distance of the received word.

Separately, maximum-likelihood (ML) decoding can be performed by the Viterbi algorithm [6] on any trellis representation of the cyclic code. The natural trellis implied by the generator polynomial $g_{\text{cyclic}}(x)$ has 2^r states where r is the degree of $g_{\text{cyclic}}(x)$. Decoding on the natural trellis essentially treats the BCH generator polynomial as the generator polynomial of a terminated convolutional code. However, for most cyclic codes of interest, the number of states in the natural trellis induces a complexity that is not practical.

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. The work of H. Yao and A. Vardy was supported in part by the NSF under grants CCF-1719139 and CCF-1764104. Research was carried out in part at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with NASA.

Trellises with lower complexity than the natural trellis can be found using techniques developed in [7]–[9] that identify minimum complexity trellis representations. Such trellises are often time varying and can have a complex structure. Furthermore, at least for the high-rate binary BCH code examples [10] in this paper, the minimum complexity trellis representations turn out to have complexity similar to the natural trellis.

As a main contribution, this paper presents serial list Viterbi algorithm (S-LVA) as an ML decoder for composite cyclic codes that, at least for our high-rate BCH examples, has complexity similar to that of the Viterbi decoder used in communication devices today. The proposed decoding algorithm also supports complete hard decoding as opposed to the bounded distance decoding of the Berlekamp-Massey and Euclidean algorithms.

In [11] the concatenation of a convolutional code with an outer cyclic redundancy check (CRC) was recognized as being a new convolutional code with a larger constraint length. This paper makes the dual observation: when a cyclic code has a composite a generator polynomial $g_{\text{cyclic}}(x)$, it can be factored into a "convolutional encoder" polynomial $g_{\text{cc}}(x)$ and an "outer CRC" polynomial $g_{\text{crc}}(x)$. Factoring $g_{\text{cyclic}}(x)$ into $g_{\text{cc}}(x)$ and $g_{\text{crc}}(x)$ facilitates S-LVA decoding.

The paper also studies the complexity of the S-LVA decoder and shows that for high-rate binary BCH examples S-LVA has lower complexity than soft decoding based on the minimal-complexity trellis of the original cyclic code.

This paper is organized as follows: Section II describes the decoding approach and presents examples and simulations for several high-rate binary BCH codes. Section III explores the complexity of the new decoding approach, showing that decoding complexity for S-LVA depends on the expected list size, which decreases as signal-to-noise ratio improves. At typical frame error rate (FER) operating points, the expected list size is small and the average decoding complexity is dominated by the complexity of the natural trellis for $g_{\text{cc}}(x)$. Section IV investigates the minimal trellis representations for our high-rate BCH examples and shows that these minimal trellises will not have significantly lower complexity than the natural trellises. In our examples, S-LVA achieves a complexity much lower than Viterbi decoding on the minimal trellis. Section V concludes the paper.

TABLE I
EXAMPLE BINARY BCH CODES THAT CAN BE VIEWED AS BINARY CONVOLUTIONAL CODES

m	r	Conjugate	δ	$g_{\text{cyclic}}(x)$	K	N	R
6	6	α	3	$(103)_8$	57	63	0.90
7	7	α	3	$(211)_8$	120	127	0.94
8	8	α	3	$(435)_8$	247	255	0.97
9	9	α	3	$(1021)_8$	502	511	0.98
10	10	α	3	$(2011)_8$	1013	1023	0.99

II. SERIAL LIST VITERBI ON FACTORED CYCLIC CODES

This section describes how a cyclic code can be represented as a rate-1 terminated convolutional code and how factoring the cyclic code generator polynomial into the product of two polynomials facilitates S-LVA decoding. This section also presents examples and simulations for several high-rate binary BCH codes.

A. Cyclic Codes as Rate-1 Convolutional Codes

It is well known that a natural trellis that has 2^r states exists for each degree-r generator polynomial $g_{\text{cyclic}}(x)$. See, for example, [12]. The decoder design proposed in this paper can be best understood by using this result to view the cyclic code as a terminated rate-1 convolutional code. This is presented below as a theorem.

Theorem 1. *Every cyclic code is a terminated convolutional code.*

Proof: Consider the cyclic code with degree-r generator polynomial $g_{\text{cyclic}}(x)$ and codewords $c(x) = w(x)g_{\text{cyclic}}(x)$ where $w(x)$ is the message polynomial. The codeword produced by multiplying the polynomials $w(x)$ and $g_{\text{cyclic}}(x)$ is identical to the codeword produced by passing $w(x)$ followed by r zeros through the rate-1 feedforward convolutional encoder circuit defined by $g_{\text{cyclic}}(x)$. ■

We use the term *BCH convolutional code* to refer to a BCH code represented as a terminated rate-1 convolutional code, i.e., a convolutional code where the number of input symbols k per encoder operation and the number of output symbols n per encoder operation are both 1. The trellis termination redundancy lowers the rate from 1 to the actual rate of the cyclic code. This is distinct from the notion in [13] of a BCH convolutional code as a standard rate-k/n ($k < n$) convolutional code designed using ideas from BCH theory.

We focus on binary BCH codes, but the approach generalizes to non-binary BCH codes such as Reed-Solomon codes. Although we refer to the convolutional code as having rate one, the rate is, of course, actually below one because of the termination bits. For a degree-r binary BCH generator polynomial $g_{\text{cyclic}}(x)$ for a code with blocklength $2^m - 1$ over Galois field $\text{GF}(2^m)$, the actual rate is $R = 1 - \frac{r}{2^m-1}$. Viewing the BCH code as a convolutional code, there exists a natural trellis structure for every BCH code, which facilitates soft Viterbi decoding.

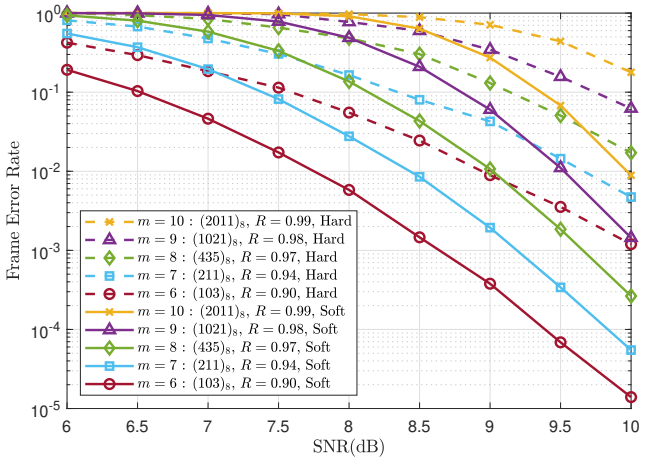

Fig. 1. FER performance of BCH convolutional codes of degree $m = 6, 7, \ldots, 10$ decoded using the Viterbi algorithm. Both soft decoding and complete hard decoding are shown.

Table I presents example high-rate binary BCH codes where each $g_{\text{cyclic}}(x)$ is a minimal polynomial of degree $m = 6, 7, \ldots, 10$. Note that the polynomials are described in octal form with degree decreasing from left to right. For example, $x^6 + x + 1$ is represented by $(103)_8$. The rate R is greater than or equal to 0.9 for all these examples. For each code, m is the base-2 logarithm of the field size over which the binary BCH code was designed. The degree r of $g_{\text{cyclic}}(x)$ for these examples is also equal to m. In each case, $g_{\text{cyclic}}(x)$ is the minimal polynomial for the conjugate element provided in the column identified as "Conjugate", where α is a primitive element that is a root of the primitive polynomial used to define the Galois field. The designed distance of the BCH code is denoted δ.

Fig. 1 provides simulation results from applying the Viterbi algorithm to the BCH convolutional codes of Table I. The system model is as follows: Let $w(x)$ be the K-bit information sequence, and $c(x) = w(x)g_{\text{cyclic}}(x)$ is the N-bit BCH convolutional codeword. After BPSK modulation, the constellation points are transmitted over the additive white Gaussian noise (AWGN) channel. The signal-to-noise ratio (SNR) is defined as the amplitude of a single BPSK signal divided by the variance of a one-dimensional, zero-mean Gaussian.

If hard Viterbi decoding is employed, the noisy constellation points are first hard decoded to the nearest BPSK symbol and the corresponding bits are used by the hard Viterbi decoder. Unlike Berlekamp-Massey or the Euclidean algorithm, this is a complete decoder. If soft Viterbi decoding is employed, the received noisy constellation points are directly used by the soft Viterbi decoder. As expected, soft decoding provides a significant benefit over hard decoding. For example, for $m = 6$ case, the FER can be improved by 2 orders of magnitude at 10 dB.

TABLE II
EXAMPLE BINARY BCH CODES FACTORED SUCH THAT THE GENERATOR POLYNOMIAL $g_{\text{CYCLIC}}(x) = g_{\text{CC}}(x)g_{\text{CRC}}(x)$.

m	r	r_{cc}	r_{crc}	δ	$g_{\text{cc}}(x)$	$g_{\text{crc}}(x)$	K	N	R
6	12	6	6	5	$(103)_8$	$(127)_8$	51	63	0.81
7	14	7	7	5	$(211)_8$	$(217)_8$	113	127	0.89
8	16	8	8	5	$(435)_8$	$(567)_8$	239	255	0.94
9	18	9	9	5	$(1021)_8$	$(1131)_8$	493	511	0.96
10	20	10	10	5	$(2011)_8$	$(2017)_8$	1003	1023	0.98

TABLE III
BCH EXAMPLES WITH $g_{\text{CYCLIC}}(x) = g_1(x)g_2(x)g_3(x)$, $g_{\text{CC}}(x) = g_1(x)$, AND $g_{\text{CRC}}(x) = g_2(x)g_3(x)$

m	δ	r	$g_1(x)$	$g_2(x)$	$g_3(x)$	K	N	R
6	7	18	$(103)_8$	$(127)_8$	$(147)_8$	45	63	0.71
7	7	21	$(211)_8$	$(217)_8$	$(235)_8$	106	127	0.84
8	7	24	$(435)_8$	$(567)_8$	$(763)_8$	231	255	0.91
9	7	27	$(1021)_8$	$(1131)_8$	$(1461)_8$	484	511	0.95
10	7	30	$(2011)_8$	$(2017)_8$	$(2415)_8$	993	1023	0.97

B. Simplifying the Trellis by Factoring out a CRC Polynomial

In [11] the concatenation of a convolutional code with an outer CRC was recognized as being a new convolutional code of a larger constraint length. This new convolutional code is catastrophic because all convolutional polynomials share the CRC polynomial as a factor, but this is not a significant issue for FER performance of finite blocklength codes. This paper simplifies decoding of cyclic codes by applying the dual of this result to the rate-1 convolutional code representation of cyclic codes. This is stated below as a theorem:

Theorem 2. *If a cyclic code generator polynomial $g_{cyclic}(x)$ is the product of multiple minimal polynomials and thus can be factored as*

$$g_{cyclic}(x) = g_{cc}(x)g_{crc}(x)\,, \tag{1}$$

then ML decoding of the cyclic code is achieved by using S-LVA (with a sufficiently large list) to identify a convolutional codeword of $g_{cc}(x)$ that also passes the CRC check using $g_{crc}(x)$.

Proof: ML decoding of the cyclic code is certainly achieved by Viterbi decoding on the natural trellis for $g_{\text{cyclic}}(x)$. The set of admissible codewords for the cyclic code with generator polynomial $g_{\text{cyclic}}(x)$ is the set of paths on the natural trellis for $g_{\text{cyclic}}(x)$. This set is a strict subset of the set of paths on the natural trellis for $g_{\text{cc}}(x)$. Specifically, the set of paths on the natural trellis for $g_{\text{cyclic}}(x)$ is the set of paths on the natural trellis for $g_{\text{cc}}(x)$ whose associated input sequences also pass the CRC check based on $g_{\text{crc}}(x)$. Thus, the most likely codeword according to Viterbi decoding on the trellis for $g_{\text{cc}}(x)$ that also passes the CRC check based on $g_{\text{crc}}(x)$ is also the most likely codeword for Viterbi decoding on the trellis for $g_{\text{cyclic}}(x)$, i.e. it is the ML codeword for the cyclic code. ■

Note that, of course, the codewords produced by $g_{\text{cyclic}}(x)$ and the concatenation of $g_{\text{cc}}(x)$ and $g_{\text{crc}}(x)$ are the same codewords and have the same length. The natural trellis for $g_{\text{cyclic}}(x)$ is traversed by applying the input sequence $w(x)$ plus terminating zeros to the encoder described by $g_{\text{cyclic}}(x)$. The natural trellis for $g_{\text{cc}}(x)$ is traversed by applying input sequences of the lengths of $w(x)g_{\text{crc}}(x)$ plus terminating zeros to the encoder described by $g_{\text{cc}}(x)$.

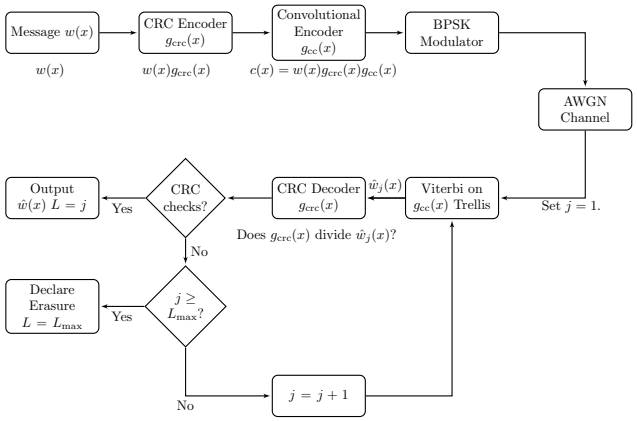

Fig. 2. System model for a cyclic code transmitted over the AWGN channel and decoded using the S-LVA algorithm.

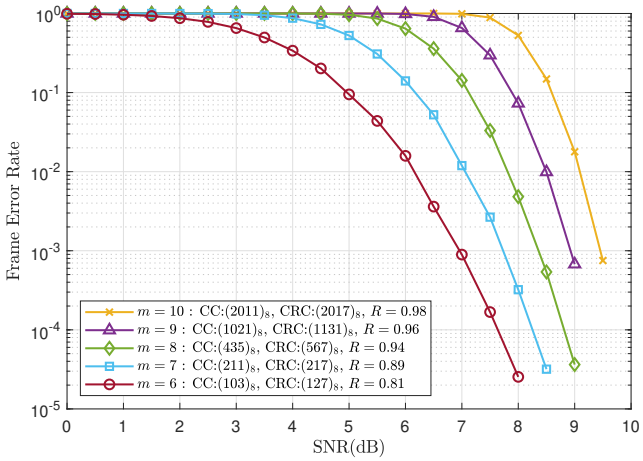

Fig. 3. FER for the codes described in Table II decoded using Theorem 2 with $g_{\text{cyclic}}(x)$ factored into $g_{\text{crc}}(x)$ and $g_{\text{cc}}(x)$ as described in Table II.

C. Binary BCH example

Theorem 2 provides an ML decoder that implements S-LVA by factoring the cyclic code generator polynomial into $g_{\text{crc}}(x)$ and $g_{\text{cc}}(x)$. Tables II and III provide examples of binary BCH codes for which $g_{\text{cyclic}}(x) = g_{\text{cc}}(x)g_{\text{crc}}(x)$ using Theorem 2.

Fig. 2 shows the overall system used for simulation, including the S-LVA decoding algorithm. ML decoding is achieved by performing list decoding [14]–[16] on the convolutional code defined by $g_{\text{cc}}(x)$, which has significantly fewer states than the natural trellis of the generator polynomial $g_{\text{cyclic}}(x)$ of the cyclic code. The polynomial $g_{\text{crc}}(x)$ is used as a CRC polynomial to determine when the S-LVA can be concluded. This approach allows for soft ML decoding (or complete hard decoding) of the cyclic code. Figs. 3 and 4 show the FER

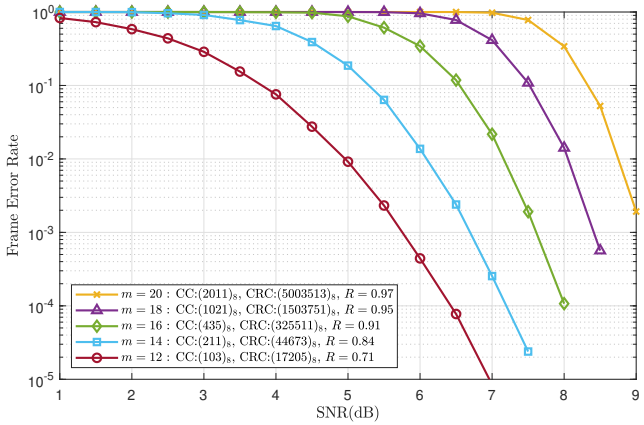

Fig. 4. FER for the codes described in Table III decoded using Theorem 2 with $g_{\text{cyclic}}(x)$ factored into $g_{\text{crc}}(x)$ and $g_{\text{cc}}(x)$ as described in Table II.

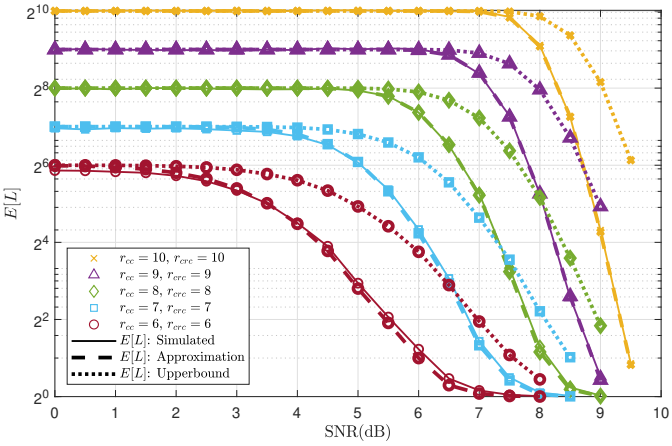

Fig. 5. Expected list size $\mathbb{E}[L]$ as a function of SNR for decoding the binary BCH codes defined in Table II. Also shown for each code are the upper bound of (3) and the approximation of (5).

performance vs. SNR for the codes described in Tables II and III, respectively, decoded according to Theorem 2 and Figure 2.

Define M_{cc} to be the total number of valid terminated convolutional codewords for $g_{\text{cc}}(x)$. As illustrated in Fig. 2, define $L_{\max}$ to be the maximum number of codeword candidates S-LVA will consider before terminating, regardless of whether a valid codeword is identified. There may be times when $L = L_{\max}$ without S-LVA identifying a CC codeword that also passes the CRC. In these cases, the decoder will declare an erasure, and is therefore not a complete decoder.

The phrase "with a sufficiently large list" in Theorem 2 is certainly met when $L_{\max} = M_{\text{cc}}$. The decoder will also be complete if $L_{\max} = M_{\text{cc}}$.

For the decoder to be complete and ML as in Theorem 2, $L_{\max}$ should be When $L_{\max} < M_{\text{cc}}$, Regardless of the size L, if the decoder does not declare an erasure it will identify the ML codeword. $L_{\max}$ is 10^4 in Figs. 3, 4, and 5.

III. Complexity of Serial List Viterbi Decoding

In [17], the authors present tables that compare the time and space complexity for different implementations of the LVA. Although the multiple-list tree-trellis algorithm (ml-TTA) achieves linear time complexity for the backward passes of the S-LVA, the implementation does not support floating point precision without the use of quantization. The red-black tree tree-trellis algorithm (T-TTA) is another implementation of the S-LVA that uses a red-black tree to store the metric differences during the traceback operation. The time complexity results in [17] indicate that T-TTA achieves the best performance among algorithms that support floating point precision. Our simulation results in Section II-C and the complexity analysis in this section both use the T-TTA of [17].

As observed in [17] as well as in [18], [19], the complexity of S-LVA depends on the size L of the list that had to be explored before identifying a codeword that satisfies the CRC. This paper is interested in the average complexity of S-LVA, which depends on $\mathbb{E}[L]$ and is a function of E_s/N_0. As shown in [18], $\mathbb{E}[L]$ converges to $2^{r_{\text{crc}}}$ for very low values of SNR and to one for very large values of SNR. This general behavior is illustrated in Fig. 5, which shows $\mathbb{E}[L]$ as a function of SNR for S-LVA for the binary BCH codes described in Table II.

The overall complexity as a function of list size L is explored more fully in [19] (See especially Fig. 12 of [19]), but there are two primary conclusions. The first conclusion is that when the list size is small, S-LVA adds very little overall complexity to the complexity of the Viterbi decoder for the convolutional code, about 10% for a list size of $L = 10$ and essentially no complexity increase for $L = 1$. The second conclusion is that for typical operating points of target FER, the list size is small enough that the overall complexity is similar to that of Viterbi decoding on the convolutional trellis.

The second conclusion is demonstrated by examining Fig. 5 and Fig. 3 together. From Fig. 3, the operating points of the codes at FER 10^{-3} are in the range of $7 - 9.5$ dB. Examining Fig. 5 we see that the actual expected list size is always below 10 at this operating point.

Let ϵ denote FER, and let ϵ_{cc} denote the FER of the convolutional code defined by $g_{\text{cc}}(x)$, i.e., the FER of using Viterbi decoding on the natural trellis of $g_{\text{cc}}(x)$ with no CRC check. The following theorem provides an upper bound on the average list size $\mathbb{E}[L]$ as a function of ϵ_{cc}:

Theorem 3. *Let L be the random variable describing the list size associated with S-LVA based on a particular factorization*

$$g_{cyclic}(x) = g_{cc}(x)g_{crc}(x)\,. \tag{2}$$

The average list size $\mathbb{E}[L]$ has the following upper bound:

$$\mathbb{E}[L] \leq 1 - \epsilon_{cc} + \epsilon_{cc}2^{r_{crc}} \tag{3}$$

$$\leq 1 + \epsilon_{cc}2^{r_{crc}}. \tag{4}$$

Proof: The set of admissible codewords using the natural trellis for $g_{\text{cyclic}}(x)$ is a strict subset of the set of admissible

codewords using the natural trellis for $g_{\text{cc}}(x)$. Whenever the Viterbi decoder using the natural trellis for $g_{\text{cc}}(x)$ identifies the correct codeword on the first attempt, i.e. it does not make a frame error, the CRC checks immediately and the list size is $L = 1$. If the first attempt using the natural trellis for $g_{\text{cc}}(x)$ causes a frame error, the list size is greater than 1 but the conditional expected list size in that case is upper bounded by $2^{r_{\text{crc}}}$ as shown in [18]. This proves the bound. ■

Theorem 3 is illustrated in Fig. 5, but turns out to be somewhat loose in practice. A tighter approximation, for which an analytic justification remains a topic of ongoing investigation, uses ϵ_{cyclic}, the overall FER of the cyclic code:

$$\mathbb{E}[L] \approx 1 - \epsilon_{\text{cyclic}} + \epsilon_{\text{cyclic}} 2^{r_{\text{crc}}} . \tag{5}$$

As seen in Fig. 5 and in all of our other simulations, (5) is an excellent approximation of actual expected list size.

Using (5), the factorization of $g_{\text{cyclic}}(x)$ can be chosen in practice so that the expected list size is small, e.g. by choosing r_{crc} so that the approximation of (5) is close to two at the target value ϵ^* of ϵ_{cyclic}, i.e. by choosing $r_{\text{crc}} \approx -\log_2 \epsilon^*$ This ensures that the average complexity at the operating point ϵ^* is close to the complexity of Viterbi decoding on the trellis of $g_{\text{cc}}(x)$. In the examples of Table II, the trellis of $g_{\text{cc}}(x)$ has $2^{\frac{r}{2}}$ states when the BCH generator polynomial has a natural trellis with 2^r states.

IV. Minimal Trellises for Cyclic Codes

The value of the complexity reduction benefit of the S-LVA approach described in Secs. II and III depends on how much complexity reduction it provides as compared to other approaches. The natural trellis associated with the generator polynomial $g_{\text{cyclic}}(x)$ may not be the lowest complexity trellis representation. Thus, to properly assess the advantage of the proposed S-LVA approach over its alternatives, it should be compared with Viterbi decoding on the lowest-complexity trellis representation for the original cyclic code [20], [21].

This section analyzes the trellis complexity for the example BCH codes in Table II. The results of this analysis show that the S-LVA approach has significantly lower complexity than that of Viterbi algorithm using the minimal trellis.

It is well known that for linear block codes, a reduction in trellis complexity may be achieved by permuting the code coordinates [12]. For any given order of the code coordinates, one can construct the canonical *minimal trellis* representing the code, using one of several methods described in [12, Section 4.2]. Such trellises are usually time-varying. That is, unlike the natural trellis, they do not have the same number of states at all times. Let V_i denote the set of states in the minimal trellis at time i, for $i = 0, 1, ..., N$, where N is the dimension of the code. It is well known [12] that, for any linear block code, $|V_i|$ is a power of 2 for all i. The sequence $s_0, s_1, \ldots, s_N$, where $s_i = \log_2 |V_i|$, is called the *state-complexity profile* of the trellis.

The computational complexity of Viterbi decoding on a trellis is governed by its state-complexity profile $s_0, s_1, \ldots, s_N$. In turn, the values of $s_0, s_1, \ldots, s_N$ depend on the order in which the code coordinates are arranged. Unfortunately, the problem of finding the optimal permutation of code coordinates is wide open — there are precious few codes for which such permutations are known [12, Section 5.2]. In fact, most of the work in the literature is concerned with finding the permutation that minimizes the *maximum number of states* in the trellis, namely

$$s_{\max} \stackrel{\text{def}}{=} \max\{s_0, s_1, ..., s_N\},$$

rather than the entire state-complexity profile. This is still a difficult problem. For general linear codes, finding a coordinate permutation that minimizes $s_{\max}$ is NP-hard [22], [23]. For most structured codes (including most BCH codes) only upper and lower bounds on $s_{\max}$ are known.

TABLE IV
The set of codeword weights $\mathcal{D}(C^\perp)$ for $C^\perp$ and number of states $s_{\max}$ in the minimal trellis for BCH codes in Table II

m	$g_{\text{cyclic}}(x)$	$K^\perp$	$\mathcal{D}(C^\perp)$	$s_{\max}$
6	$(103)_8(127)_8$	12	$\{0, 24, 28, 32, 36, 40\}$	12
7	$(211)_8(217)_8$	14	$\{0, 56, 64, 72\}$	14
8	$(435)_8(567)_8$	16	$\{0, 112, 120, 128, 136, 144\}$	16
9	$(1021)_8(1131)_8$	18	$\{0, 240, 256, 272\}$	18
10	$(2011)_8(2017)_8$	20	$\{0, 480, 496, 512, 528, 544\}$	20

However, in this section, we determine the *precise value of* $s_{\max}$ for all the BCH codes in Table II. In fact, we prove that for every one of these codes $s_{\max} = N - K = K^\perp$. That is, for these BCH codes, the maximum number of states in the minimal trellis is equal to the number of states in the natural trellis. In order to establish this result, we make use of the following theorem of Ytrehus [24].

Theorem 4 (Ytrehus)**.** *For an (N, K, d) binary linear code, define its distance set (or weight set) as follows:*

$$\mathcal{D}(C) = \{0 \le w \le N \ : \ \exists \ c, c' \in C \text{ such that } d(c, c') = w\}$$

If $\mathcal{D}(C) \subseteq \{0\} \cup \{d, d+1, \cdots, 2d-3\}$ then $s_{\max} = K$ under all permutations of code coordinates.

It is well known that the minimal trellis of any linear code and the minimal trellis of its dual code have the same number of states at all times (this result was first established by Forney in [25]). For the BCH codes in Table II, the dimensions $K^\perp$ of their dual codes and the distance sets of the dual codes are given in Table IV. Notably, all the distance sets in Table IV satisfy the condition of Theorem 4. It follows that the trivial upper bound $s_{\max} \le K^\perp$ holds with equality for all these codes, under all permutations of the code coordinates. This suggests that the reduction in trellis complexity that can be obtained for these BCH codes by permuting their coordinates is very limited.

Let $|V|$ and $|E|$ denote, respectively, the number of states and the number of branches in a trellis. Then the complexity of Viterbi decoding on this trellis (the number of operations required by the Viterbi algorithm) is given by

TABLE V
VITERBI DECODING COMPLEXITIES FOR NATURAL TRELLISES VS. OUR PERMUTED TRELLISES FOR BCH CODES IN TABLE II

m	$g_{\text{cyclic}}(x)$	$Q_{\text{Viterbi,natural}}$	$Q_{\text{Viterbi,permuted}}$
6	$(103)_8(127)_8$	5.0×10^5	3.2×10^5
7	$(211)_8(217)_8$	4.9×10^6	2.8×10^6
8	$(435)_8(567)_8$	4.4×10^7	2.4×10^7
9	$(1021)_8(1131)_8$	3.7×10^8	2.0×10^8
10	$(2011)_8(2017)_8$	3.1×10^9	1.6×10^9

$2|E| - |V| + 1$ (see [12, Section 3.3]). For comparison, in Table V we construct the minimum-complexity (permuted) trellises we could find for the BCH codes in Table II, and compare the resulting Viterbi decoding complexity with that of the natural trellis. Here $Q_{\text{Viterbi,natural}}$ and $Q_{\text{Viterbi,permuted}}$ are the number of operations required by the Viterbi algorithm on the natural trellis and the trellis we found by permuting the code coordinates, respectively.

From Table V, we see that by permuting the coordinates of the example BCH codes, we can reduce the Viterbi decoding complexity by a factor of about 2 at most. Thus the proposed S-LVA decoding approach is significantly more efficient than conventional Viterbi decoding on the minimal trellis these BCH codes.

Moreover, we could also apply coordinate permutations to optimize the trellis complexity of the trellis used by Viterbi decoding of $g_{\text{cc}}(x)$. In this way, both the S-LVA decoding approach and the trellis theory of linear block codes, in combination, can be used to reduce the decoding complexity even further.

V. CONCLUSION

This paper presents a low complexity approach for ML decoding of cyclic codes that is based on factoring the generator polynomial $g_{\text{cyclic}}(x)$ into a convolutional encoder polynomial $g_{\text{cc}}(x)$ and a CRC polynomial $g_{\text{crc}}(x)$. This factorization facilitates S-LVA decoding which, with r_{crc} chosen appropriately for the target FER, has a complexity that is similar to the complexity of performing Viterbi decoding on the natural trellis for $g_{\text{cc}}(x)$.

When the two factors have equal degree, the trellis used for Viterbi decoding has a number of states equal to the square root of the number of states in the natural trellis for the cyclic code. For the binary BCH examples in this paper, the minimal trellis associated with the cyclic code was not significantly smaller than the natural trellis so that the factoring approach described by Theorem 2 and Figure 2 leads to a significant practical complexity reduction.

REFERENCES

[1] E. R. Berlekamp, *Algebraic Coding Theory*. New York: McGraw-Hill, 1968.

[2] J. Massey, "Shift-register synthesis and bch decoding," *IEEE Trans. Inf. Theory*, vol. 15, no. 1, pp. 122–127, Jan 1969.

[3] Y. Sugiyama, M. Kasahara, S. Hirasawa, and T. Namekawa, "A method for solving key equation for decoding goppa codes," *Information and Control*, vol. 27, no. 1, pp. 87–99, January 1975.

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

[5] V. Guruswami and M. Sudan, "Improved decoding of reed-solomon and algebraic-geometric codes," *Proc. 39th Annu. Symp. on Foundations of Comput. Sci. (Cat. No.98CB36280)*, 1998.

[6] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimal decoding algorithm," *IEEE Trans. Inf. Theory*, vol. 13, pp. 260–269, 1967.

[7] J. Wolf, "Efficient maximum likelihood decoding of linear block codes using a trellis," *IEEE Trans. Inf. Theory*, vol. 24, no. 1, pp. 76–80, Jan 1978.

[8] D. Muder, "Minimal trellises for block codes," *IEEE Trans. Inf. Theory*, vol. 34, no. 5, pp. 1049–1053, 1988.

[9] A. Vardy and Y. Be'ery, "Maximum-likelihood soft decision decoding of bch codes," *IEEE Trans. Inf. Theory*, vol. 40, no. 2, pp. 546–554, Mar 1994.

[10] R. C. Bose and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Information and Control*, vol. 3, pp. 68–79, 1960.

[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] A. Vardy, *Handbook of Coding Theory*. Elsevier, 1998, ch. Trellis structure of codes.

[13] J. Rosenthal and F. York, "Bch convolutional codes," *IEEE Trans. Inf. Theory*, vol. 45, no. 6, pp. 1833–1844, 1999.

[14] 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.

[15] 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.

[16] R. Johannesson and K. S. Zigangirov, *Fundamentals of Convolutional Coding*, J. B. Anderson, Ed. New Jersey, USA: IEEE Press, 1999.

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

[18] 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.

[19] H. Yang, E. Liang, and R. D. Wesel, "Joint design of convolutional code and crc under serial list viterbi decoding." [Online]. Available: http://arxiv.org/abs/1811.11932v1

[20] A. B. Kiely, S. Dolinar, L. Ekroot, R. J. McEliece, and W. Lin, "Minimal trellises for linear block codes and their duals," in *Telecommunications and Data Acquisition Report*, J. H. Yuen, Ed., May 1995.

[21] A. B. Kiely, S. J. Dolinar, R. J. McEliece, L. L. Ekroot, and Wei Lin, "Trellis decoding complexity of linear block codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 6, pp. 1687–1697, Nov 1996.

[22] F. R. Kschischang and G. B. Horn, "A heuristic for ordering a linear block code to minimize trellis state complexity," in *32nd Allerton Conf. on Communication, Control, and Comput.*, 1994, pp. 75–84.

[23] N. Kashyap, "Matroid pathwidth and code trellis complexity." [Online]. Available: http://arxiv.org/abs/0705.1384v1

[24] O. Ytrehus, "On the trellis complexity of certain binary linear block codes," *IEEE Trans. Inf. Theory*, vol. 41, pp. 559–560, 1995.

[25] J. G. D. Forney, "Coset codes ii: Binary lattices and related codes," *IEEE Trans. Inf. Theory*, vol. 34, pp. 1152–1187, 1988.