Application of List Viterbi Algorithms to Improve the Performance in Space Missions using Convolutional Codes

Application of List Viterbi Algorithms to Improve the Performance in Space Missions using Convolutional Codes

Riccardo Schiavone
Politecnico di Torino
Torino, Italy
riccardo.schiavone@polito.it

Roberto Garello
Politecnico di Torino
Torino, Italy
roberto.garello@polito.it

Gianluigi Liva
German Aerospace Center (DLR)
Wessling, Germany
gianluigi.liva@dlr.de

***Abstract*—Currently, several space missions are still using convolutional codes, which are among the available coding options of the CCSDS telemetry recommendation. When convolutional codes are employed, the CCSDS specification mandates the use of an outer CRC code to perform error detection over the transfer frame. Alternatively, the CRC code may be used, together with list Viterbi decoding of the inner convolutional code, to significantly improve the performance of the coding scheme. In this paper, we first compute the distance spectrum of the concatenation of the outer CRC code and the inner convolutional codes recommended by the CCSDS. By means of a union bound on the block error probability under maximum-likelihood decoding, we estimate the extra coding gain achievable by the concatenation with respect to the use of the Viterbi algorithm applied to the decoding of the inner convolutional code only. The extra coding gain is close to 3 dB. Then, we consider the application of the list Viterbi algorithm and we discuss some techniques useful to reduce its complexity in practical implementations. Results show that it is possible to approach the 3 dB extra coding gain with negligible increase in the decoding complexity with respect to Viterbi decoding of the inner convolutional code.**

***Index Terms*—Channel codes, convolutional codes, CRC codes, list decoders, space communications.**

I. Introduction

Recently, the concatenation of convolutional codes (CCs) with high-rate outer binary linear block codes was introduced [1], showing how the concatenation can be efficiently decoded by means of list Viterbi algorithms (LVAs) [2], mimicking a similar approach used for the decoding of polar codes concatenated with outer codes [3]. In particular, cyclic redundancy check (CRC) codes have been used as outer codes in [1], [4]–[9]. In the short block length regime, the concatenation of an outer CRC code with an inner convolutional code (denoted in the following by the shorthand "CRC+CC") with moderate/small memory was shown to perform remarkably close to finite block length bounds [10] down to low error rates with a manageable decoding complexity [1].

Even if more powerful error correcting codes like low-density parity-check (LDPC) and turbo codes are now available as coding options in Consultative Committee for Space Data Systems (CCSDS) telemetry synchronization and channel coding recommendation [11], CCs are still used by various space missions (e.g., in some European Space Agency (ESA) Earth's observation (EO) satellites missions, as well as in small/cubesat missions), thanks to their simplicity and their reasonably good performance. Note that when CCs are used as the coding option of a telemetry (TM) link, the presence of an outer CRC code is mandatory, according to the TM and the advanced orbiting systems (AOS) space data link protocol [12], [13]. The CRC code is included to provide an error detection capability after Viterbi decoding of the inner CC. In this work, we study the performance of the CRC+CC scheme defined by the CCSDS under list Viterbi decoding of the inner code. The adoption of this decoding technique implies that the outer CRC code is not used anymore for pure error detection, but it becomes part of the error correction mechanism as proposed in [1]. (A discussion on the error correction vs. error detection capability trade-off is provided at the end of Section III). We estimate the extra coding gain achievable by the CRC+CC concatenation with respect to the use of the Viterbi algorithm (VA) applied to the decoding of the inner convolutional code only. The estimation is based on the union bound on the maximum likelihood (ML) block error probability, which is computed based on the distance spectra of the CC and of the CRC+CC concatenation. The extra coding gain is quantified in 3 dB. Simulation results show that this 3 dB gain is approachable with moderate list sizes. The paper is organized as follows. In Section II, we introduce the notation and we recall elements of the CCSDS telemetry recommendation. List Viterbi algorithms are reviewed in Section III. The iterative parallel-list Viterbi algorithm is detailed in Section IV. Numerical results are presented in Section V. Conclusions follow in Section VI.

978-1-6654-8780-1/22/$31.00

II. CCSDS Telemetry Recommendation

A. Transfer Frames

The CCSDS TM [12] and AOS [13] recommendation provides functions for transferring data using the protocol data unit called transfer frame (TF). The synchronization and channel coding sublayer provides additional functions necessary for transferring the TF over the space link. These functions are error-control coding and decoding, TF delimiting and synchronizing, and bit transition generation and removal.

As coding options, different families of channel codes are available: convolutional codes, parallel/serial turbo codes, Reed-Solomon codes, concatenated Reed-Solomon and convolutional codes, and low density parity-check codes. The decoders of the last three families of codes have a native transfer frame validation property i.e., error detection capability, contrary to the case of CCs and turbo codes. For this reason the *Transfer Frame Error Control Field* defined in [12], [13] is mandatory if those two families of codes are used, while the same field is optional for the other three families.

For all codes (except for the CCs), the TF is encoded into a codeword, which is then preceded by an attached synchronization marker (ASM), which is a fixed binary sequence, providing the TF delimiting function needed to perform frame synchronization. For CCs, the case is different, and the TF, before being CC encoded, is preceded by the $32-$bit long ASM pattern $(1ACFFC1D)_{\text{HEX}}$, which is also CC encoded. It is important to note that when no TFs are available, an only idle data (OID) frame which contains dummy data is encoded to maintain the link availability. Then the information sequences are organized as continuous sequences containing data/OID TFs separated by ASM patterns, before entering the convolutional encoder.

B. Convolutional and Cyclic Redundancy Check Codes

The *Transfer Frame Error Control Field* is a $16-$bit vector generated by the recursive and systematic CRC encoder with generator polynomial

$$g(D) = 1 + D^5 + D^{12} + D^{16}.$$

The encoder is initialized to all ones at time $t = 0$, i.e., at the beginning of each TF. The CC encoder is defined in [11] and it is a non-recursive one with memory $\nu = 6$ (64 states) and polynomial generator matrix

$$\boldsymbol{G}(D) = \left[1 + D + D^2 + D^3 + D^6, \right. \\ \left. 1 + D^2 + D^3 + D^5 + D^6\right].$$

In principle the CC encoder is not terminated. However, the start/end states fo the encoder are fixed due to the encoding of the ASM. Let us denote by $\boldsymbol{\sigma}_i$ the i-th state of the trellis. The start/end states (in binary representation) of the CC encoder are forced to $\boldsymbol{\sigma}_{\text{start}} = (101110)$ and $\boldsymbol{\sigma}_{\text{end}} = (110101)$ which are the last and the first 6 bits of the ASM. In the remainder of the paper, we denote by K the number of bits in the TF, prior to CRC encoding. Denoting by $m = 16$ the number of parity bits introduced by the CRC, and by $h = 32$ the ASM length, we have that the code rate is given by

$$R = \frac{K}{K + m + h} R_{\text{cc}}$$

where R_{cc} is the code rate of the convolutional encoder.

Remark 1. Due to the initialization of the CRC encoder to the all-one state, and to the termination conditions imposed by the ASM, the overall CRC+CC strictly speaking does not result in a binary linear code, but its codewords are the coset of the CRC+CC binary linear code.In this paper, we are going to analyze the code performance over the binary-input additive white Gaussian noise (AWGN) channel. Owing to the channel symmetry, the analysis of the affine CRC+CC concatenation reduces to analysis of the linear CRC+CC concatenation obtained by (i) initializing the CRC encoder state to the all-zero vector and (ii) forcing the CC start/end state to the all-zero one. In the remainder of the paper, we will make use of this observation and analyze the linear CRC+CC concatenation.

III. CRC-Aided List Viterbi Algorithms

Let us denote by $\mathcal{I}_i$ the set of the CC trellis state indexes at time t which are connected to state i at time $t+1$. As shown in Fig. 1 (where we focus on rate-$1/2$ CC as the CCSDS code), we further denote by $\lambda_t^{\boldsymbol{\sigma}_j \to \boldsymbol{\sigma}_i}$ the VA state transition metric over the edge connecting state $\boldsymbol{\sigma}_j$ with state $\boldsymbol{\sigma}_i$ in the t-th trellis section. Since we will focus on the binary-input AWGN channel, the state transition metric will be defined as the correlation between the (modulated) edge label and the corresponding observation at the channel output. Finally, $\Lambda_t^{\boldsymbol{\sigma}_i}$ is the cumulative metric for state $\boldsymbol{\sigma}_i$ at time t.

The parallel-list Viterbi algorithm (PLVA) [2] is a simple modification to the VA, which outputs a list $\mathcal{L}$ containing the L trellis paths with largest likelihood. The PLVA works as follows. At a generic state $\boldsymbol{\sigma_i}$ at time t, a sorted list $\mathcal{L}_t^{\boldsymbol{\sigma}_i}$ of size L is stored, which contains the state metrics of the L paths (reaching state $\boldsymbol{\sigma}_i$) with largest likelihood, in decreasing order. Let us denote by $\Lambda_t^{\boldsymbol{\sigma}_i,(\ell)}$ the ℓ-th metric in $\mathcal{L}_t^{\boldsymbol{\sigma}_i}$, i.e., the cumulative metric of the ℓ-th most likely path reaching state $\boldsymbol{\sigma}_i$ at time t. At time $t = 0$ decoding begins from the all-zero state with $\mathcal{L}_0^{\boldsymbol{\sigma}_0}$ containing only the element $\Lambda_0^{\boldsymbol{\sigma}_0,(1)} = 0$. Then, for all the states $\boldsymbol{\sigma}_i$ of the trellis section at time t, we construct $\mathcal{L}_t^{\boldsymbol{\sigma}_i}$ according to

$$\Lambda_{t+1}^{\boldsymbol{\sigma}_i,(\ell)} = \max_{\substack{j \in \mathcal{I}_i \\ z=1\ldots L}}^{(\ell)} \left(\Lambda_t^{\boldsymbol{\sigma}_j,(z)} + \lambda_t^{\boldsymbol{\sigma}_j \to \boldsymbol{\sigma}_i} \right)$$

where the $\max^{(\ell)}$ operator returns the ℓ-th largest argument. The principle is illustrated in Fig. 1. Once the last trellis section is reached, the L paths stored in the final state list are extracted, and each of them is checked with outer CRC. If no path satisfies the CRC constraints, a decoding error is declared. Otherwise, the path with the largest cumulative metric among those satifsying the CRC constraints is output as the final decision.

A possible alternative to the PLVA is the serial-list Viterbi algorithm (SLVA) [2]. The algorithm works iteratively: in

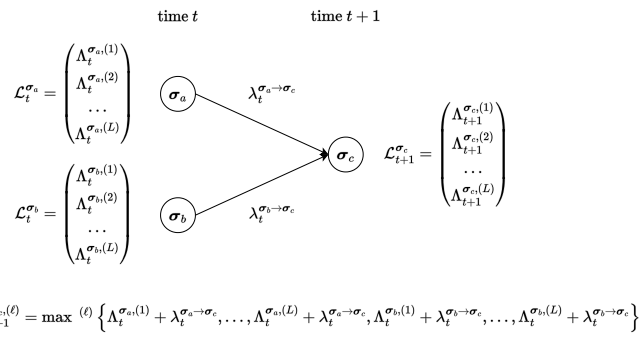

Fig. 1. Construction of the list at a generic state according to the parallel-list Viterbi algorithm.

order to save computations, rather than computing the L most likely paths all at once, the algorithm outputs the ℓ most likely path after the ℓ-th iteration. Hence, decoding stops as soon as a path that satisfies the CRC code constraints is found, possibly halting the search at an early stage. At its first iteration, SLVA is equivalent to the VA. If the path selected by the VA does not satisfy the CRC code constraints, SLVA scans the trellis, in order to find the path with the smallest likelihood difference with respect to the ML path and verify that the CRC code constraints over that path are met, otherwise the algorithm runs another iteration. In general, at the ℓ-th step, SLVA scans the not yet scanned paths with the smallest likelihood difference from the $(\ell - 1)$ most likely paths, and then extracts the ℓ-th most likely path. The procedure is repeated until or the message associated to the ℓ-th most likely best path satisfies the CRC code constraints, or L iterations are reached. Taking advantage of the tree structure of the trellis many computations can be saved and the SLVA can be accelerated as described in [14].

Remark 2. The CRC code was introduced in combination with CCs in the CCSDS telemetry synchronization and channel coding recommendation [11] to provide the receiver with an error detection mechanism. The adoption of the decoding techniques outlined in this Section implies that the outer CRC code is not used anymore for pure error detection, but it becomes part of the error correction mechanism [1]. Still, some error detection capability is retained by this approach: in fact, a decoding error can be declared whenever none of the paths composing the final list satisfies the CRC code constraints. The error detection capability is, in this case, strongly influenced by the list size: the larger L, the higher will be the undetected error probability. Note that for any list size the undetected block error probability is upper bounded by the ML block error probability of the CRC+CC concatenation. A careful analysis of the undetected error probability, which may play an important role in the design of telemetry links, will be addressed in future studies.

IV. Iterative Parallel-List Viterbi Algorithm

In this section, we discuss about a possible implementation based on the PLVA with an average algorithmic complexity of the decoder which decreases, while increasing the signal-to-noise ratio (SNR), up to reach nearly the same complexity of the VA.

A. The Algorithm

The idea behind the iterative PLVA is to run instances of the PLVA, starting from a small list size L, e.g. $L = 1$, and then increasing L every time that the CRC code constraints are not satisfied for all the paths in the list extracted by the PLVA, or L has exceeded a maximum list size $L_{\max}$. Doing so, it is possible to mimic the behaviour of the SLVA and reducing the average algorithmic complexity when increasing the SNR. We define the way of increasing L every time that the check of the CRC conditions is not met as the scheduler of the iterative PLVA. This scheduler can be any function $sched(\cdot)$ that given $L^{(i)}$, the value of L at the $i-$th iteration, it outputs $L^{(i+1)}$, the value of L at the $(i+1)-$th iteration, where $L^{(i+1)} > L^{(i)}$. E.g. we can have a simple constant increase in the list size as in $L^{(i+1)} = sched(L^{(i)}) = L^{(i)} + 1$, or a scheduler which double the list size at every iteration $L^{(i+1)} = sched(L^{(i)}) = 2 \cdot L^{(i)}$. We want to point out that the choice of the scheduler affects both the delay and the complexity of the algorithm in the worst-case scenario, which is the scenario in which L increases up to $L_{\max}$.

The procedure of the iterative PLVA is reported in Algorithm 1, where for a received sequence $\boldsymbol{y}$, and given a maximum list size $L_{\max}$ and the scheduler $sched(\cdot)$, it outputs a list $\mathcal{L}$ containing the list of the messages associated with the L most likelihood paths over the trellis and sorted in decreasing order of their likelihood, NACK is a flag which is 0 in case none of the found messages satisfy the CRC code constraints and it is 1 otherwise, lastly ℓ indicates the position of the more likelihood message which met the CRC conditions, if any, in the list.

Algorithm 1 Procedure of the iterative parallel-list Viterbi algorithm for the received sequence $\boldsymbol{y}$ with the maximum list size $L_{\max}$ constraint and the list increment function $sched(\cdot)$.

procedure ITERATIVE_PLVA($\boldsymbol{y}, L_{\max}, sched(\cdot)$)
  $L \leftarrow 1$
  $\mathcal{L} \leftarrow \emptyset$
  NACK$\leftarrow 0$
  while ($L \leq L_{\max}$ and NACK $= 0$) **do**
    $(\mathcal{L}, \text{NACK}, \ell) = \text{PLVA}(\boldsymbol{y}, L)$
    $L \leftarrow sched(L)$
  end while
  return $(\mathcal{L}, \text{NACK}, \ell)$
end procedure

B. Software and Hardware Sorting

In software, the list sorting at each trellis node for the PLVA with list size L should be done taking into account that each node has two predecessors, each with an already sorted list, and so we need to perform a merge operation. This can be done with minimum complexity using L comparisons in series,

starting from the best element of each list, and then increasing the list index of the node whose element was selected by the previous comparison. The resulting sorting delay is L. Another possibility for sorting the list elements in modern CPU/GPUs relying on several computing cores, which allows to execute several comparisons in parallel, is via the application of a sorting network [15]. The same sorting network can also be used for hardware implementations.

In [9] a sorting network based on the bitonic sorter [16], but with delay $O(\log_2(L))$ is proposed instead of $O(\log_2^2(L))$ of the original bitonic sorter. It requires $\frac{L}{2}\log_2(L)+L$ comparators, instead of $L\log_2^2(L)$ of the bitonic sorter. We sketch in Fig. 2 an example of the sorting network when two nodes with $L=8$ are merged, and we refer to [9, Algorithm 2] for the details of the general sorting network for any list size L.

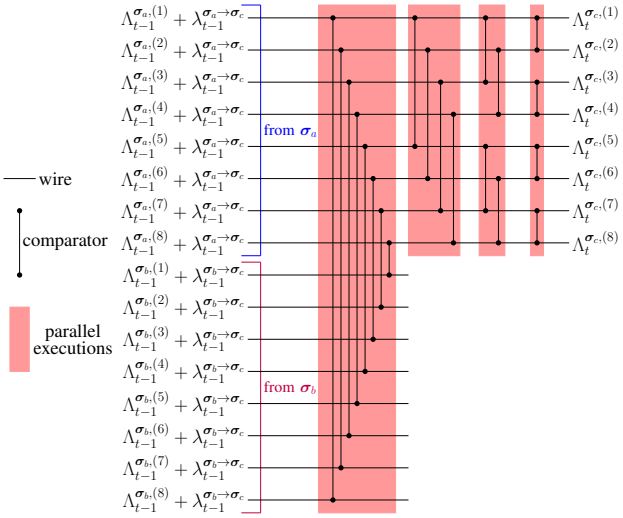

Fig. 2. Example of the proposed sorter in [9] to sort 2 trellis nodes with $L=8$ elements each. The red boxes underline the operations which can run in parallel.

V. Numerical Results

In this section, we report the results on the block error probability P_B of the CRC+CC concatenation of the CCSDS standard under CRC-aided list decoding. The results are obtained via Monte Carlo simulations for various lengths K of the uncoded TF. In particular we use $K \in \{1768,\ 3552,\ 8904\}$ bits (that, when the 16 bits of the CRC are added, corresponds to some of the most typical CCSDS input lengths: 1784, 3568 and 8920 bits). We assume a binary phase-shift keying (BPSK) modulated signal over the AWGN channel, and perfect frame and carrier synchronization at the receiver. At the channel output, the i-th observation is given by

$$y_i = x_i + n_i$$

where $x_i \in \{\pm 1\}$ and $n_i \sim \mathcal{N}(0,\sigma^2)$. The block error probability is estimated as frame error rate (FER), and the SNR is provided in terms of E_b/N_0 with E_b being the energy per information bit, and N_0 the noise single-sided power spectral density.

A. Asymptotic Coding Gain Analysis

We provide next an analysis of the asymptotic coding gain achievable by the CRC+CC concatenation when decoded as described in Section III. The analysis is based on the derivation of a union upper bound on the block error probability of the inner CC and of the CRC+CC concatenation. To proceed with the analysis we first derived the weight enumerator of the inner CC. The derivation is based on standard techniques [17]. The derivation of the weight enumerator for the CRC+CC is based on the techniques described in [9]. The multiplicity A_w of weight-w codewords (limited to the lower tail of the distance spectrum) is provided for various values of K in Table I. Remarkably, the concatenation with the outer code allows to double the minimum distance of the plain CC. Moreover, it is interesting to note that the A_w terms grow linearly with respect to K for the CC only, according to [18], while the same terms have a quadratic increase for the CRC+CC scheme. Owing to the doubled minimum distance, under ML decoding the CRC+CC concatenation will yield an asymptotic coding gain of 3 dB over the plain CC.

TABLE I
Distance spectra of the CC and of the CRC+CC concatenation of the CCSDS standard for various transfer frame lengths.

code	K	$d_{\min}$	$A_{d_{\min}}$, $A_{d_{\min}+1}$, $A_{d_{\min}+2}$, ...
CC	1768	10	19580, 0, 67477, 0, 342205, ...
CRC+CC	1768	20	7431, 0, 28005, 0, 175576, ...
CC	3552	10	39204, 0, 135269, 0, 686517, ...
CRC+CC	3552	20	16351, 0, 91945, 0, 610136, ...
CC	8904	10	98076, 0, 338645, 0, 1719453, ...
CRC+CC	8904	20	59091, 0, 557162, 0, 3581187, ...
CC	16368	10	180180, 0, 622277, 0, 3160005, ...
CRC+CC	16368	20	197358, 0, 1800329, 0, 11847522, ...

Given the distance spectrum, we can upper bound the error probability of the code under ML decoding by using the union bound as

$$P_B \le \frac{1}{2}\sum_{w>0} A_w \operatorname{erfc}\left(\sqrt{wR\frac{E_b}{N_0}}\right). \qquad (1)$$

In Fig. 3 we have depicted truncated versions of the P_B upper bounds for the various lengths K for both the CC and the CRC+CC codes (the curves in Fig. 3 are obtained by truncating the sum in (1) up to $w=120$). Looking at block error rates below 10^{-7}, where the bound is expected to be tight, we can already observe the additional coding gain of ≈ 3 dB achieved by the CRC+CC concatenated scheme.

B. Practical Coding Gains for Space Missions

Figures 4, 5, and 6 report the simulation results for $K=1768$, $K=3552$, and $K=8904$, respectively. The results

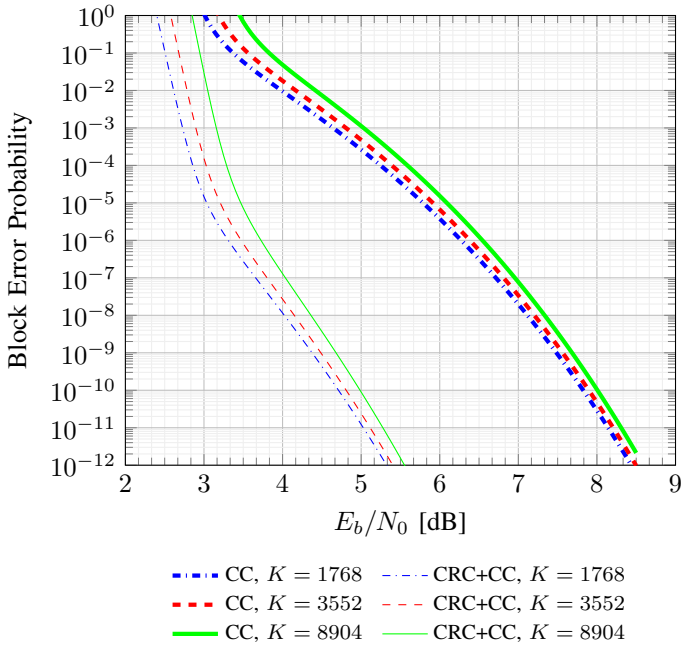

Fig. 3. Truncated union bounds on the ML block error probability of the CCs and CRC+CC concatenations of the CCSDS TM recommendation over the AWGN channel. BPSK modulation.

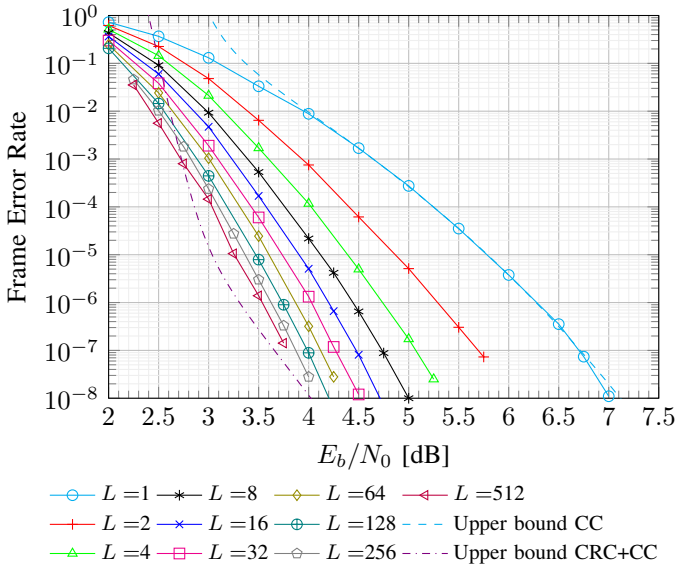

Fig. 4. FER vs. SNR for the CCSDS CRC+CC code with BPSK modulation over the AWGN channel for a TF of length $K = 1768$ and CC encoder with $R_{\text{CC}} = 1/2$.

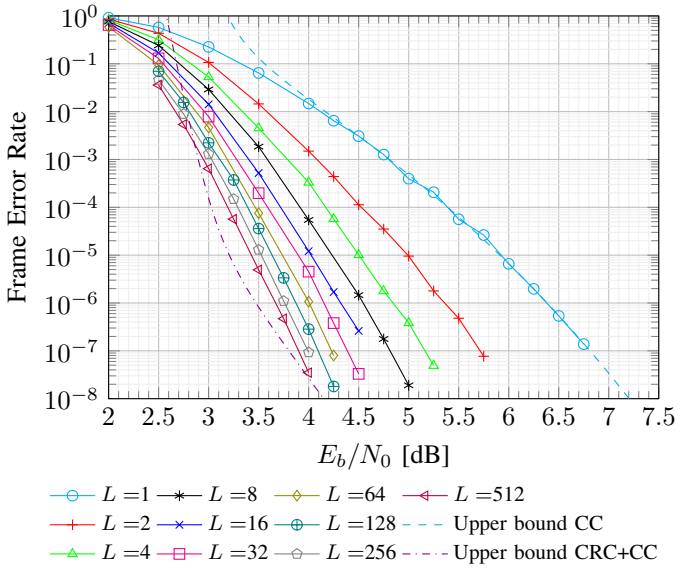

Fig. 5. FER vs. SNR for the CCSDS CRC+CC code with BPSK modulation over the AWGN channel for a TF of length $K = 3552$ and CC encoder with $R_{\text{CC}} = 1/2$.

are provided for various list sizes. Note that for a given list size SLVA and PLVA yield the same performance. Hence, the type of list decoder is left unspecified in the figures. For $L = 1$, both list decoders reduce to the application of standard Viterbi decoder to the inner CC. The simulations are obtained by counting 100 frame errors. On each chart, the (truncated) union bounds on the ML block error probability are provided for both the CC and the CRC+CC concatenation.

Focusing on Fig. 4 ($K = 1768$), we observe that as the SNR increases it is possible to approach the ML decoding bound with decreasing list size L. For instance, at $E_b/N_0 = 4$ dB we need a maximum list size $L = 64$ to be within 0.5 dB from the ML bound for the CRC+CC code, while at $E_b/N_0 = 4.5$ dB the same result is achieved with $L = 32$, with a coding gain of ≈ 2.5 dB over the ML block error probability of the plain CC. Similar results hold for the other lengths of the transfer frame.

C. Complexity Using the Iterative PLVA

In Fig. 7 we show the results of the application of the iterative PLVA algorithm in terms of average complexity normalized with respect to the VA, which we denote by $E[L]$. The maximum list size is $L_{\max}$ and we have used a scheduler which doubles the list size each time the CRC code conditions are not met ($L^{(i+1)} = sched(L^{(i)}) = 2 \cdot L^{(i)}$). To compute $E[L]$, firstly we consider that each iteration of the iterative PLVA algorithm with list L has complexity $\approx L \cdot C_{\text{VA}}$, where C_{VA} is the complexity of the VA. Secondly, we have to keep in mind that the iterative PLVA algorithm runs first the VA, then if it fails it runs PLVA with $L = 2$, then if it fails doubles L to 4, ecc ..., until L overcomes the maximum allowed value. Thus, the complexity of the iterative PLVA $C^{(j)}_{\text{iterative PLVA}}$ for the j^{th} received sequence is given by

$$C^{(j)}_{\text{iterative PLVA}} = \sum_{L \in \mathcal{L}^{(j)}} L \cdot C_{\text{VA}},$$

where $\mathcal{L}^{(j)}$ is the set of list sizes used to correctly decode the i^{th} sent codeword or to reach the maximum allowed list size. E.g., if the algorithm has maximum list size of 8, but the j^{th} received sequence is successfully decoded when $L = 4$, then $C^{(j)}_{\text{iterative PLVA}} = 1 \cdot C_{\text{VA}} + 2 \cdot C_{\text{VA}} + 4 \cdot C_{\text{VA}} = 7 \cdot C_{\text{VA}}$. We can

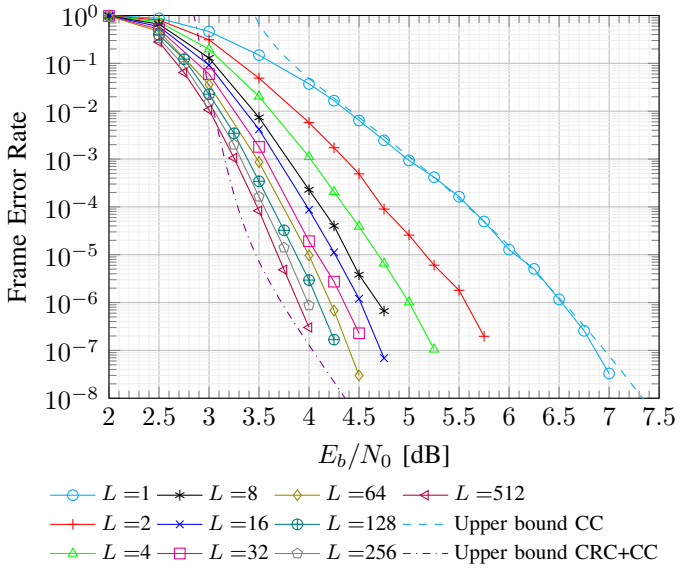

Fig. 6. FER vs. SNR for the CCSDS CRC+CC code with BPSK modulation over the AWGN channel for a TF of length $K = 8904$ and CC encoder with $R_{\mathsf{CC}} = 1/2$.

normalize $C^{(j)}_{\text{iterative PLVA}}$ with respect to the complexity of the VA and we can estimate the average value of its normalized expression, which we call $E[L]$, at a specific value of E_b/N_0 according to,

$$E[L] \approx \frac{1}{N_{tests}} \sum_{j=1}^{N_{tests}} \sum_{L \in \mathcal{L}^{(j)}} L,$$

where N_{tests} is the number of simulated messages to estimate $E[L]$.

Looking at the estimated values of $E[L]$ with a TF of length $K = 1768$, which are depicted in Fig. 7, and various maximum list sizes L, we can see that when E_b/N_0 is low, $E[L]$ corresponds to the sum of all the powers of two list sizes up to L. This means that independently of the list size, nearly no messages can be correctly decoded at early stages. While looking for target FER of 10^{-7} of practical applications (e.g., satellite TM links), where $E_b/N_0 > 4$ dB for the ML decoding of the CRC+CC, using the iterative PLVA decoder $E[L] \to 1$ for all the various list sizes, meaning that nearly all codewords are correctly decoded during the first iteration, and very few requires extra iterations. Also in software simulations, using the iterative PLVA algorithm, the average simulation speed is nearly the same of the Viterbi implementation for $E_b/N_0 \geq 4$ dB.

Focusing on Fig. 4, the case when $K = 1768$, if we look at the FER results when $E_b/N_0 = 4.5$ dB, we can see that Viterbi fails to decode once every ≈ 500 messages. This means that with the use of the iterative PLVA only less than 0.2% of messages require to run a list $L = 2$ PLVA decoder. Going down, keeping fixed E_b/N_0, we see that approximately 1 every 16 000 messages requires to increase L to 4, nearly 1 every 200 000 needs $L = 8$, about 1 every 1.5 million needs $L = 16$ and so on. And this fraction decreases, at higher values of E_b/N_0 for the same L values.

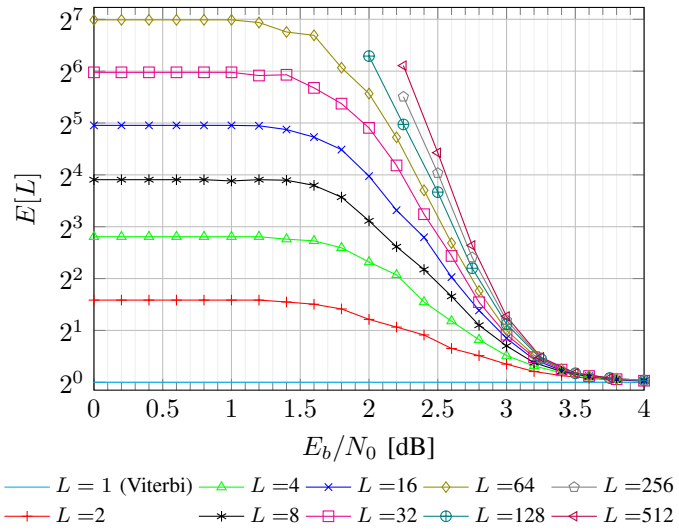

Fig. 7. $E[L]$ vs. SNR for the CCSDS CRC+CC code with BPSK modulation over the AWGN channel for a TF of length $K = 1768$ and CC encoder with $R_{\mathsf{CC}} = 1/2$, and decoded using the iterative PLVA with maximum list size L in the legend.

D. Punctured Codes

In [11] the output of the CC encoder described in Section II-B may be punctured to achieve higher code rates. Results on the FER of several punctured codes are provided in Fig. 8 for $R_{\mathsf{cc}} = 2/3$, Fig. 9 for $R_{\mathsf{cc}} = 3/4$ and Fig. 10 for $R_{\mathsf{cc}} = 5/6$. In all cases, the TF length has been set to 1768. On the same charts, union bounds on the block error probability under ML decoding are provided as reference (the lower tail of the distance spectra used for the evaluation of the union bounds is provided in Table II). The results show already coding gains around 2.75 dB at a FER of 10^{-7}, when list decoding is employed in place of the VA only. Also in this case, the expected list size approaches 1 with increasing SNR.

TABLE II
DISTANCE SPECTRA OF THE PUNCTURED CCS AND OF THE PUNCTURED CRC+CCS OF THE CCSDS STANDARD FOR VARIOUS RATES OF THE ENCODER, WHILE KEEPING FIXED THE TF LENGTH TO $K = 1768$.

code	R_{cc}	$d_{\min}$	$A_{d_{\min}}$, $A_{d_{\min}+1}$, $A_{d_{\min}+2}$, ...
CC	2/3	6	891, 14229, 42607, 139960, ...
CRC+CC	2/3	14	1756, 21066, 76351, 341467, ...
CC	3/4	5	4738, 18328, 94331, 524544, ...
CRC+CC	3/4	10	808, 2646, 15199, 80484, ...
CC	5/6	4	4971, 24449, 230378, 1754473, ...
CRC+CC	5/6	8	787, 4618, 36036, 317668, ...

Similar results holds also for the punctured cases about the expected list size, which approaches 1, when E_b/N_0 increases.

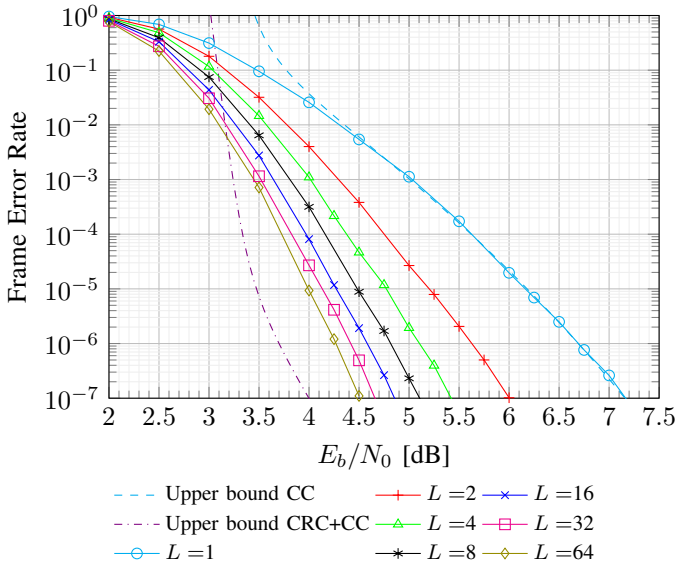

Fig. 8. FER vs. SNR for the CCSDS CRC+CC code with BPSK modulation over the AWGN channel for a TF of length $K = 1768$ and CC encoder with $R_{\mathsf{CC}} = 2/3$.

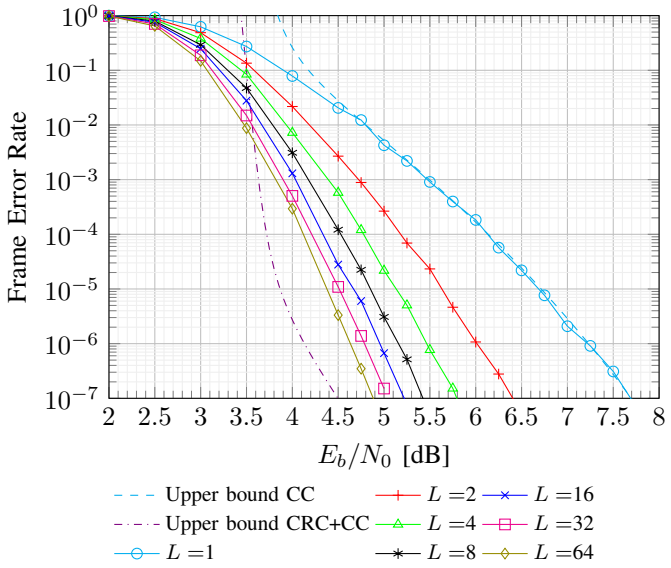

Fig. 9. FER vs. SNR for the CCSDS CRC+CC code with BPSK modulation over the AWGN channel for a TF of length $K = 1768$ and CC encoder with $R_{\mathsf{CC}} = 3/4$.

VI. Conclusions

In this paper, the application of list Viterbi decoding to the concatenation of a CRC code with an inner convolutional code recommended by the CCSDS has been studied. The extra coding gain achievable by the list decoder with respect to the plain Viterbi decoding of the inner convolutional code has been analyzed by means of union bounds on the maximum-likelihood block error probability, showing that the use of list decoding may yield an additional coding gain close to 3 dB. The 3 dB gain was shown to be approachable already with moderate list sizes, with negligible increase in average

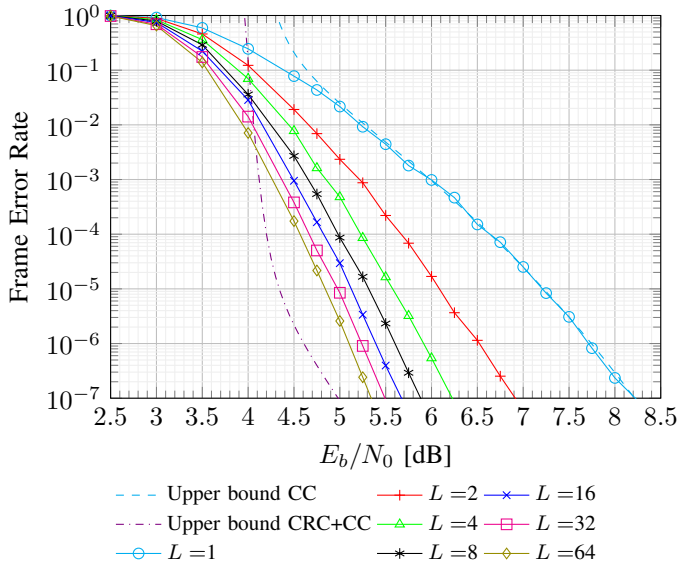

Fig. 10. FER vs. SNR for the CCSDS CRC+CC code with BPSK modulation over the AWGN channel for a TF of length $K = 1768$ and CC encoder with $R_{\mathsf{CC}} = 5/6$.

complexity compared to the Viterbi algorithm. This means that if the ground systems receivers of the space missions which rely on the convolutional codes are enabled to support list Viterbi decoders, it is possible to halve the transmit power while introducing a marginal increase of complexity on ground, or it is possible the use of the extra-gain to counter additional interference, jamming, scintillation and other kind of impairments.

VII. Acknowledgments

The authors wish to thank Dr. Dariush Divsalar, Prof. William Ryan, the UCLA Communications Systems Laboratory group leaded by Richard D. Wesel, Dr. Andrea Modenini of ESA/ESTEC, and Prof. Monica Visintin of Politecnico di Torino for fruitful technical discussions on the topic.

References

[1] 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, 2015.

[2] N. Seshadri and C.-E. Sundberg, "List Viterbi Decoding Algorithms with Applications," *IEEE Trans. Commun.*, vol. 42, no. 234, pp. 313–323, 1994.

[3] I. Tal and A. Vardy, "List Decoding of Polar Codes," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2213–2226, 2015.

[4] H. Yang, S. V. Ranganathan, and R. D. Wesel, "Serial List Viterbi Decoding with CRC: Managing Errors, Erasures, and Complexity," in *Proc. IEEE Global Telecommun. Conf.* IEEE, 2018, pp. 1–6.

[5] H. Yang, E. Liang, and R. D. Wesel, "Joint Design of Convolutional Code and CRC under Serial List Viterbi Decoding," *arXiv preprint arXiv:1811.11932*, 2018.

[6] V. Bioglio, C. Condo, and I. Land, "Design of Polar Codes in 5G New Radio," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 29–40, 2020.

[7] W. Sui, H. Yang, B. Towell, A. Asmani, and R. D. Wesel, "High-Rate Convolutional Codes with CRC-Aided List Decoding for Short Blocklengths," *arXiv preprint arXiv:2111.07929*, 2021.

[8] L. Wang, D. Song, F. Areces, and R. D. Wesel, "Achieving Short-Blocklength RCU bound via CRC List Decoding of TCM with Probabilistic Shaping," *arXiv preprint arXiv:2111.08756*, 2021.
[9] R. Schiavone, "Channel Coding for Massive IoT Satellite Systems," Master's thesis, Politecnico di Torino, 2021.
[10] 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.
[11] CCSDS, "TM Synchronization and Channel Coding," *Recommended standard. Blue Book*, no. 131.0-B-4, April 2022.
[12] ——, "TM Space Data Link Protocol," *Recommended standard. Blue Book*, no. 132.0-B-3, October 2021.
[13] ——, "AOS Space Data Link Protocol," *Recommended standard. Blue Book*, no. 732.0-B-4, October 2021.
[14] M. Roder and R. Hamzaoui, "Fast Tree-Trellis List Viterbi Decoding," *IEEE Trans. Commun.*, vol. 54, no. 3, pp. 453–461, 2006.
[15] D. E. Knuth, *The Art of Computer Programming: Volume 3: Sorting and Searching*. Addison-Wesley Professional, 1998.
[16] K. E. Batcher, "Bitonic Sorting," *Goodyear Aerospace Corp., Rep. GER-11869*, 1964.
[17] J. Wolf and A. Viterbi, "On the Weight Distribution of Linear Block Codes Formed from Convolutional Codes," *IEEE Trans. Commun.*, vol. 44, no. 9, pp. 1049–1051, Sep. 1996.
[18] M. P. Fossorier, S. Lin, and D. J. Costello, "On the Weight Distribution of Terminated Convolutional Codes," *IEEE Trans. Inf. Theory*, vol. 45, no. 5, pp. 1646–1648, 1999.