An analysis of the Channel Coding Performance in DVB-RCS Systems

Teodor B. Iliev[1]

Abstract – **The convolutional turbo codes are very flexible codes, easily adaptable to a large range of data block sizes and coding rates. This is the main reason for their being adopted in the DVB standard for Return Channel via Satellite (DVB-RCS). The paper presents the turbo coding/decoding scheme specified in this standard. Simulation results show the performance of the coding scheme chosen, in particular for the transmission of ATM cells and MPEG transport stream packets.**

Keywords – **Digital video broadcast-return channel via satellite (DVB-RCS), turbo code, MAP algorithm**

I. INTRODUCTION

The Digital Video Broadcasting (DVB) Project was founded in 1993 by the European Telecommunications Standards Institute (ETSI) with the goal of standardizing digital television services. Its initial standard for satellite delivery of digital television, dubbed DVB-S, used a concatenation of an outer (204,188) byte shortened Reed Solomon code and an inner constraint length 7, variable rate (R ranges from 1/2 to 7/8) convolutional code [1].

The same infrastructure used to deliver television via satellite can also be used to deliver Internet and data services to the subscriber. Because DVB-S only provides a downlink, an uplink is also needed to enable interactive applications such as web browsing [2].

In the DVB-RCS (Return Channel via Satellite) to transmit an uplink signal back to the satellite is used the same antenna like for receiving the downlink signal. However, given the small antenna aperture and requirement for a low-cost, low-power amplifier, there is very little margin on the uplink. Therefore, strong FEC coding is desired. For this reason, the DVB Project has adopted turbo codes for the satellite return channel in its DVB-RCS standard [3].

The DVB-RCS turbo code was optimized for short frame sizes and high data rates. Twelve frame sizes are supported raging from 12 bytes to 216 bytes, including a 53 byte frame compatible with ATM and a 188 byte frame compatible with both MPEG-2 and the original DVB-S standard. The return link supports data rates from 144 kbps to 2 Mbps and is shared among terminals by using muti-frequency time-division multiple-access (MF-TDMA) and demand-assigned multiple-access (DAMA) techniques. Eight code rates are supported, ranging from R=1/3 to R=6/7.

Like the turbo codes used in other standards, a pair of constituent RSC encoders is used along with log-MAP or max-log-MAP decoding [4]. The decoder for each constituent code performs best if the encoder begins and ends in a known state, such as the all-zeros state. This can be accomplished by independently terminating the trellis of each encoder with a tail which forces the encoder back to the all-zeros state. However, for the small frame lengths supported by DVB-RCS, such a tail imposes a non-negligible reduction in code rate and is therefore undesirable. As an alternative to terminating the trellis of the code, DVB-RCS uses circular recursive systematic convolutional (CRSC) encoding [5], which is based on the concept of tailbiting [6]. CRSC codes do not use tails, but rather are encoded in such a way that the ending state matches the starting state.

II. ENCODING IN DVB-RCS

The CRSC constituent encoder used by DVB-RCS is shown in Fig. 1. The encoder is fed blocks of k message bits which are grouped into $N=k/2$ couples. The number of couples per block can be $N \in \{48, 64, 212, 220, 228, 424, 432, 440, 752, 848, 856, 864\}$. The number of bytes per block is N=4. In Fig. 1, the symbol A represents the first bit of the couple, and B represents the second bit. The two parity bits are denoted W and Y [7].

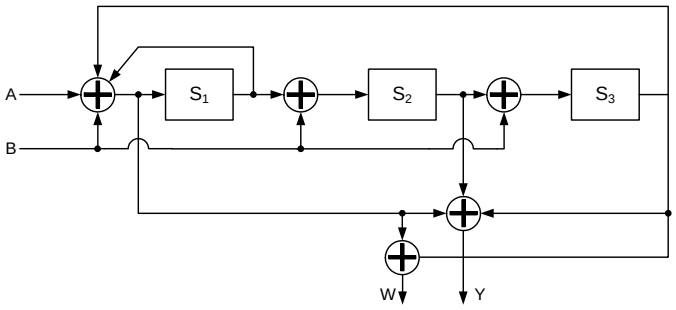

Fig.1 Duobinary CRSC constituent encoder used by DVB-RCS

Let the vector $S_k=[S_{k,1}\ S_{k,2}\ S_{k,3}]^T$, $S_{k,m} \in \{0,1\}$ denote the state of the encoder at time k. The inputs and outputs of the encoder are defined over $GF(4)$, only binary values are stored within the shift register and thus the encoder has just eight states. The encoder state at time k is related to the state at time k–1 by:

$$S_{k+1} = GS_k + X_k\,, \tag{1}$$

where

$$X_k = \begin{bmatrix} A_k + B_k \\ B_k \\ B_k \end{bmatrix} \tag{2}$$

[1]Assis. Prof. Ph.D Eng. Teodor B. Iliev is with the Department of Communication Systems and Technologies, 8 Studentska Str., 7017 Rousse, Bulgaria, E-mail: tiliev@ecs.ru.acad.bg

and

$$G = \begin{bmatrix} 101 \\ 100 \\ 010 \end{bmatrix} \tag{3}$$

Because of the tailbiting nature of the code, the block must be encoded twice by each constituent encoder. During the first pass at encoding, the encoder is initialized to the all-zeros state $S_0 = [0\ 0\ 0]^T$. After the block is encoded, the final state of the encoder S_N is used to derive the circulation state:

$$S_c = \left(I + G^N\right)^{-1} S_N \tag{4}$$

Where the above operations are over $GF(2)$. The matrix $I+G^N$ is not invertible if N is a multiple of the period of the encoder's impulse response. The circulation state S_c can be found from S_N by using a lookup table [3]. When the circulation state is found, the data is encoded again. This time, the encoder is set to start in state S_c and will be guaranteed to also end in state S_c.

The first encoder operates on the data in its natural order, yielding parity couples $\{W_{k,1}, Y_{k,1}\}$. The second encoder operates on the data after it has been interleaved. Interleaving is performed on two levels. First, interleaving is performed within the couples, and second, interleaving is performed between couples. Let $\{A_k', B_k'\}$ denote the sequence after the first level of interleaving and $\{A_k'', B_k''\}$ denote the sequence after the second level of interleaving. In the first level of interleaving, every other couple is reversed in order, i.e. $(A_k', B_k') = (B_k, A_k)$ if k is even, otherwise $(A_k', B_k') = (A_k, B_k)$. In the second level of interleaving, couples are permuted in a pseudorandom fashion. The exact details of the second level permutation can be found in the standard [3].

After the two levels of interleaving, the second encoder (which is identical to the first) encodes the sequence $\{A_k'', B_k''\}$ to produce the sequence of parity couples $\{W_{k,2}, Y_{k,2}\}$. As with the first encoder, two passes of encoding must be performed, and the second encoder will have its own independent circulation state. To create a rate R=1/3 turbo code, a codeword is formed by first transmitting all the uninterleaved data couples $\{A_k, B_k\}$, then transmitting $\{Y_{k,1}, Y_{k,2}\}$ and finally transmitting $\{W_{k,1}, W_{k,2}\}$. The bits are transmitted using QPSK modulation, so there is a one-to-one correspondence between couples and QPSK symbols. Alternatively, the code word can be transmitted by exchanging the parity and systematic bits, i.e. $\{Y_{k,1}, Y_{k,2}\}$, followed by $\{W_{k,1}, W_{k,2}\}$ and finally $\{A_k, B_k\}$.

Code rates higher than R=1/3 are supported through the puncturing of parity bits. To achieve R=2/5, both encoders maintain all the Y_k but delete odd-indexed W_k. For rate 1/2 and above, the encoders delete all W_k. For rate R=1/2, all the Y_k bits are maintained, while for rate R=2/3 only the even-indexed Y_k are maintained, and for rate R=4/5 only every fourth Y_k is maintained. Rates R=3/4 and 6/7 maintain every third and sixth Y_k respectively, but are only exact rates if N is a multiple of three, otherwise the rates are slightly lower.

III. Decoding in DVB-RCS

Decoding of the DVB-RCS code is complicated by the fact that the constituent codes are duobinary and circular. As with conventional turbo codes, decoding involves the iterative exchange of extrinsic information between the two component decoders. While decoding can be performed in the probability domain, the log-domain is preferred since the low complexity Max-Log-MAP algorithm can then be applied [4]. Unlike the decoder for a binary turbo code, which can represent each binary symbol as a single log-likelihood ratio, the decoder for a duobinary code requires three log-likelihood ratios. For example, the likelihood ratios for message couple (A_k, B_k) can be represented in the form:

$$\Lambda_{a,b}(A_k, B_k) = \log \frac{P(A_k = a, B_k = b)}{P(A_k = 0, B_k = 0)} \tag{5}$$

where (a, b) can be (0, 1), (1, 0), or (1, 1).

An iterative decoder that can be used to decode the DVB-RCS turbo code is shown in Fig. 2. The goal of each of the two constituent decoders is to update the set of log-likelihood ratios associated with each message couple. In the figure and in the following discussion, $\{\Lambda_{a,b}^{(i)}(A_k, B_k)\}$ denotes the set of LLRs corresponding to the message couple at the input of the decoder and $\{\Lambda_{a,b}^{(o)}(A_k, B_k)\}$ is the set of LLRs at the output of the decoder. Each decoder is provided with $\{\Lambda_{a,b}^{(i)}(A_k, B_k)\}$ along with the received values of the parity bits generated by the corresponding encoder (in LLR form). Using these inputs and knowledge of the code constraints, it is able to produce the updated LLRs $\{\Lambda_{a,b}^{(o)}(A_k, B_k)\}$ at its output.

As with binary turbo codes, extrinsic information is passed to the other constituent decoder instead of the raw LLRs. This prevents the positive feedback of previously resolved information. Extrinsic information is found by simply subtracting the appropriate input LLR from each output LLR, as indicated in Fig. 2 [7].

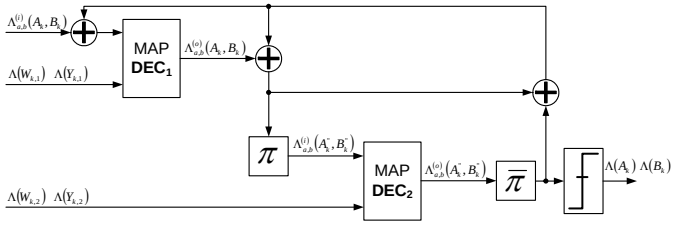

Fig.2 A decoder for the DVB-RCS code

The extrinsic information that is passed between the two decoders must be interleaved (π) or deinterleaved ($\bar{\pi}$) so that it is in the proper sequence at the input of the other decoder. Interleaving and deinterleaving between the two constituent

decoders must be done on a symbol-wise basis by assuring that the three likelihood ratios $\{\Lambda_{0,1}(A_k, B_k), \Lambda_{1,0}(A_k, B_k), \Lambda_{1,1}(A_k, B_k)\}$ belonging to the same couple are not separated.

The trellis for the duobinary constituent code is as shown in Fig. 3 and contains eight states, with four branches entering and exiting each state. The trellis contains two 4 by 4 butterflies, and because these two butterflies are independent, they can be processed in parallel. In the following, the ith state is denoted by S_i where $i \in \{0,...,7\}$ for DVB-RCS. The numbers on the left indicate the labels (*A, B, W, Y*) of the branches exiting each state. From left to right, the groups of numbers correspond to the exiting branches from top to bottom.

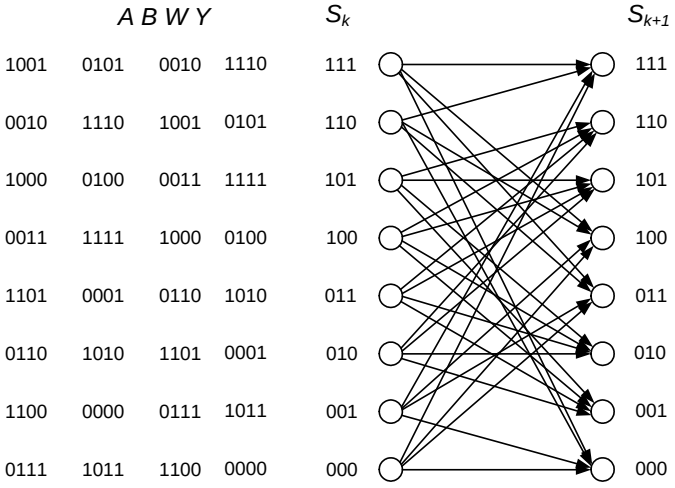

Fig.3 Trellis associated with the duobinary CRSC constituent encoder used by DVB-RCS

The extension of the Log-MAP and Max-Log-MAP algorithms [4] to the duobinary case is fairly straightforward. Each branch must be labeled with the log-likelihood ratios corresponding to the systematic and parity couples associated with that branch. Because QPSK modulation is orthogonal, the LLR of message couple (*A*, *B*) can be initialized prior to being fed into the first decoder as $\Lambda_{a,b}^{(i)}(A_k, B_k) = a\Lambda(A_k) + b\Lambda(B_k)$, where $\Lambda(C) = \log[P(C=1)/P(C=0)]$. Because extrinsic information about the parity bits is not exchanged, the parity bits can always be decomposed in a similar manner. For these reasons the systematic bits, the three likelihood ratios defined in (5) must be calculated during each iteration and exchanged between the decoders.

With $\gamma_k(S_i \to S_j)$ we denote the branch metric corresponding to state transition $S_i \to S_j$ at time *k*. The branch metric depends on the message and parity couples that label the branch along with the channel observation and extrinsic information at the decoder input. If transition $S_i \to S_j$ is labeled by (A_k, B_k, W_k, Y_k) = (*a*, *b*, *w*, *y*) then

$$\gamma_k(S_i \to S_j) = \Lambda_{a,b}^{(i)}(A_k, B_k) + \omega\Lambda(W_k) + y\Lambda(Y_k) \qquad (6)$$

As with binary codes, the constituent decoder must perform a forward and a backward recursion. Let $\alpha_k(S_i)$ denote the normalized forward metric at trellis stage *k* and state S_i, while $\alpha'_{k+1}(S_j)$ is the forward metric at trellis stage *k*+1 and state S_j prior to normalization. The forward recursion is

$$\alpha'_{k+1}(S_j) = \max_{S_i \to S_j}{}^{*} \{\alpha_k(S_i) + \gamma_k(S_i \to S_j)\} \qquad (7)$$

where the max* operation is performed over the four branches $S_i \to S_j$ leading into state S_j at time *k*+1. While the log-MAP algorithm uses the exact definition of max*, the max-log-MAP algorithm uses the approximation $\max^*(x, y) \approx \max(x, y)$.

After computing $\alpha'_{k+1}(S_j)$ for all S_j at time *k*+1, the forward metrics are normalized with respect to the metric stored in state zero:

$$\alpha_{k+1}(S_j) = \alpha'_{k+1}(S_j) - \alpha'_{k+1}(S_0) \qquad (8)$$

Similarly, let $\beta_{k+1}(S_j)$ denote the normalized backward metric at trellis state *k*+1 and state S_j and $\beta'_k(S_i)$ denote the backward metric at trellis state *k* and state S_i prior to normalization. The backward recursion is

$$\beta'_k(S_i) = \max_{S_i \to S_j}{}^{*} \{\beta_{k+1}(S_j) + \gamma_k(S_i \to S_j)\} \qquad (9)$$

where max* is over the four branches $S_i \to S_j$ exiting state S_i at time *k*. As with α, β are normalized with respect to the metric stored in state zero

$$\beta_k(S_i) = \beta'_k(S_i) - \beta'_k(S_0) \qquad (10)$$

After the forward and backward recursions have been completed, a full set of $\{\alpha_k\}$ and $\{\beta_k\}$ metrics will be stored in memory. The next step is for the decoder to use these metrics to compute the LLRs given by (5). This is accomplished by first computing the likelihood of each branch

$$Z_k(S_i \to S_j) = \alpha_k(S_i) + \gamma_k(S_i \to S_j) + \beta_{k+1}(S_j) \qquad (11)$$

The likelihood that message pair (A_k, B_k)=(*a*, *b*) is calculated using

$$t_k(a,b) = \max_{S_i \to S_j:(a,b)}{}^{*} \{Z_k\}, \qquad (12)$$

where the max* operator is over the eight branches labeled by message couple (*a*, *b*). Finally, the LLR at the output of the decoder is found as

$$\Lambda_{a,b}^{(o)}(A_k, B_k) = t_k(a,b) - t_k(0,0), \qquad (13)$$

where $(a,b) \in \{(0,1),(1,0),(1,1)\}$.

After the turbo decoder has completed a fixed number of iterations or met some other convergence criterion, a final

decision on the bits must be made [8]. This is accomplished by computing the LLR of each bit in the couple (A_k, B_k) according to

$$\begin{aligned}\Lambda(A_k) &= \max^*\left\{\Lambda_{1,0}^{(o)}(A_k,B_k),\Lambda_{1,1}^{(o)}(A_k,B_k)\right\}\\ &\quad-\max^*\left\{\Lambda_{0,0}^{(o)}(A_k,B_k),\Lambda_{0,1}^{(o)}(A_k,B_k)\right\}\\ \Lambda(B_k) &= \max^*\left\{\Lambda_{0,1}^{(o)}(A_k,B_k),\Lambda_{1,1}^{(o)}(A_k,B_k)\right\}\\ &\quad-\max^*\left\{\Lambda_{0,0}^{(o)}(A_k,B_k),\Lambda_{1,0}^{(o)}(A_k,B_k)\right\}\end{aligned}, \tag{14}$$

where $\Lambda_{0,0}^{(o)}=(A_k,B_k)=0$. The hard bit decisions can be found by comparing each of these likelihood ratios to a threshold.

IV. Simulation and Results

The performance of the DVB-RCS turbo code with QPSK modulation in an additive white Gaussian channel (AWGN), applying Max-Log-MAP and Log-MAP decoding algorithm are shown in Fig. 4 and Fig. 5. Fig. 4 shows the frame error rate (*FER*) when using blocks of *N*=212 message couples (53 bytes), code rate *R*=1/3 and the number of iteration is 10.

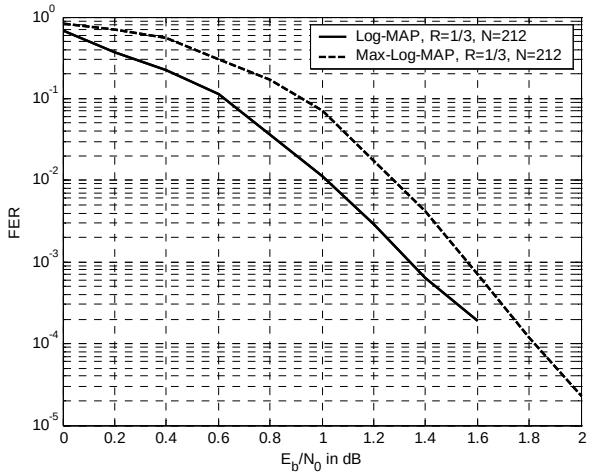

Fig.4 Performance of DVB-RCS turbo code with frame length *N*=212, code rate *R*=1/3 and 10 iteration

The two curves in Fig. 4 show the differences between Max-Log-MAP and Log-MAP decoding algorithms when using ten iterations.

Fig. 5 is illustrated the influence of the block size. Frame error rate results are shown for blocks of *N*={48, 64, 212, 432, 752} message couples, or correspondingly {12; 16; 53; 108; 188} bytes. In each case, the code rate is *R*=1/3, the circulation state is unknown at the decoder, and eight iterations of Max-Log-MAP decoding are performed. The SNR required to achieve a FER of 10^{-4} is E_b/N_0= {3.02, 2.77, 1.86, 1.65, 1.44}dB for *N*={48, 64, 212, 432, 752} respectively.

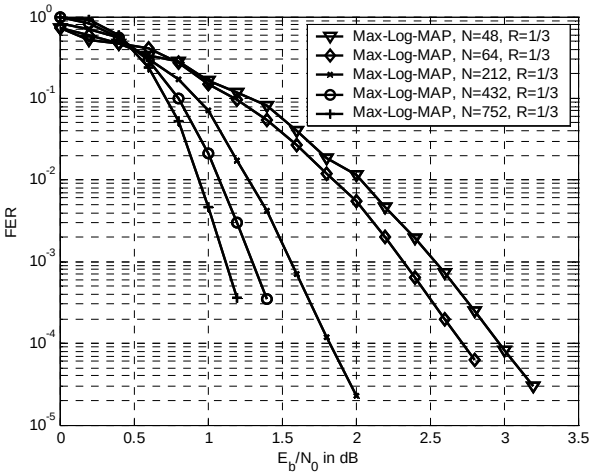

Fig.5 Performance of the DVB-RCS turbo code with code rate *R*=1/3, block size is *N* message couples, and ten iterations of Max-Log-MAP decoding algorithm

V. Conclusion

In this paper, we have highlighted some of the main features and advantages of a DVB-RCS system. From the conducted simulation we can see that the turbo code which was proposed for DVB-RCS applications is powerful, very flexible and can be implemented with reasonable complexity. This code could also be easily adjusted to many other applications, for various configurations of block sizes and code rates while retaining excellent coding gains.

References

[1] Digital Video Broadcasting (DVB); Interaction Channel for Satellite Distribution Systems; Guidelines for the Use of EN 301 790, ETSI TR 101 790 (V1.2.1), 2001.

[2] T. Iliev, I. Lokshina, D. Radev, G. Hristov, “Analysis and Evaluation of Reed–Solomon Codes in Digital Video Broadcasting Systems”, WTS’2008, Conference Proceedings, Pomona, CA, USA, 2008 (under press).

[3] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near Shannon limit error-correcting coding and decoding: turbo-codes,” ICC’93, IEEE Conference Proceedings, pp. 1064–1070, May 1993.

[4] P. Robertson, P. Hoeher, and E. Villebrun, “Optimal and sub-optimal maximum a posteriori algorithms suitable for turbo decoding”, European Trans. on Telecommun., pp. 119–125, 1997.

[5] C. Berrou, C. Douillard, and M. Jezequel, “Multiple parallel concatenation of circular recursive convolutional (CRSC) codes”, Annals of Telecommunication, pp. 166–172, 1999.

[6] H. H. Ma and J. K. Wolf, “On tail biting convolutional codes”, IEEE Trans. Commun., pp. 104–111, 1986.

[7] M. R. Soleymani, Y. Gao, and Y. Vilaipornsawai, *Turbo Coding for Satellite and Wireless Communications*, Kluwer Academic Publishers, Dordrecht, Netherlands, 2002.

[8] J. B. Anderson and S. M. Hladik, “Tailbiting MAP decoders”, IEEE Select. Areas Commun., pp. 297–302, 1998.