On the Encoding of the Multi-Non-Binary Convolutional Codes

Balta Horia¹, Alexandre De Baynast², Kovaci Maria¹

¹ University “Politehnica” of Timișoara, Faculty of Electronics and Telecommunications, V. Parvan 2, 30223 Timișoara, Romania, e-mail: horia.balta@etc.upt.ro, maria.kovaci@etc.upt.ro
² Department of Wireless Networks RWTH Aachen University, Aachen, Germany, email: ade@gollum.mobnets.rwth-aachen.de

Abstract - Recently, Douillard et al. proposed a new family of multi-binary turbo-codes based on the parallel concatenation of two constituent convolutional codes with multiple inputs that has better global performance than classical turbo-codes. The encoder is based on an r-inputs linear feedback shift register (LFSR). In this paper, we show that the encoder can also be represented by the observer canonical configuration. This configuration is essential to reduce the computational complexity of the code design procedure especially for moderate codeword sizes. Indeed, in this context, an exhaustive search usually provides better results than the EXIT chart or similar optimization tools. We show that the second configuration reduces the computational complexity of the search up to 300%. Based on this strategy, we were able to design a rate-1/2 turbo-code with two inputs and memory m=3 which outperforms the turbo-code with same characteristics proposed by Douillard et al. by 0.25 decibels for a frame error rate of 10⁻⁴.

Keywords: recursive and systematic multi-binary convolutional code, generator matrix, turbo-code.

I. INTRODUCTION

The multi-binary turbo-codes (MBTC) also referred as non-binary turbo-codes ([1]-[2]) have several advantages compare to the initial binary turbo-codes in [3] like faster convergence and lower error floor. We refer to [4] for a more detailed analysis. The design of the MBTC generally requires an exhaustive search through large sets of feasible codes [5]-[6] and interleavers [7]. This is particularly true for moderate codeword lengths. In that case, the asymptotic threshold determined by EXIT chart or by similar optimization tools may not be accurate due to the lack of randomness of the structure of the interleaver and an exhaustive search provides often better results. In the case of turbo-codes with multiple binary and non-binary inputs (MBTC [8] and MNBTC [9], respectively), this search is a major burden for their optimization since the computational complexity of the search is exponential with the number of inputs r. Therefore, restricting the search to a limited set of ‘good’ M(N)BTC is essential.

The MBTC proposed in [8] consist of the parallel concatenation of two rate-r/(r+1) recursive systematic convolutional codes (RSC) where r represents the number of inputs of the code. The encoders are based on multiple-input linear feedback shift registers (LFSR). An alternative encoding method for M(N)BTC consists in a generalization of the Fibonacci encoder ([10]-[14]). Whereas both encoding methods are equivalent for r=1, i.e., for the binary single input turbo-codes with and without puncturing, they give two different sets of encoder in the case of multiple inputs (r>1). In this paper, we compare both sets according several criteria commonly used in the system theory: cardinal number of both sets, element-to-element equivalence between both sets. This comparison is important in order to select the best encoding method for the design of the MBTC and MNBTC based on an exhaustive search.

The rest of the paper is organized as follows. Section II describes the encoding method proposed in [8]. In the third section, an alternative encoding method based on the generalization of a Fibonacci encoder is presented. In Section IV, the conditions of equivalence between both methods are determined. Finally, in Section 5, we present our optimization results and compare the packet error rate performance with the MBTC from [1].

II. CANONICAL FORM OF CONVOLUTIONAL ENCODERS BASED ON LINEAR FEEDBACK SHIFT REGISTER

Fig.1 shows the general structure of a multiple-input recursive systematic convolutional encoder that is used in [8] for each of both constituent codes. The encoder is based on an r-input linear feedback shift register (LFSR). This encoder is generally not decomposable into r single-input encoders, i.e., it cannot be represented by the controller canonical
configuration [1]. Throughout the paper, in order to simplify the notations, this configuration is simply referred as the canonical form of type ‘H’.

The encoder has \( r \) inputs \( u_1, u_2, \ldots, u_r \) and \( r+1 \) outputs corresponding to the \( r \) inputs and one redundant bit \( u_0 \) also referred as \( c \). The current encoder state is given by the outputs of the \( m \) shift registers \( s_1, s_2, \ldots, s_m \). The inputs \( u_i, i=1 \ldots r \) are physically connected to the \( j \)-th adder if \( h_{ij} \neq 0 \). \( S_t = [s_{1,t}, s_{2,t}, \ldots, s_{m,t}] \) and \( U_t = [u_{1, t}, u_{2, t-1}, \ldots, u_{r, t}] \) describe the encoder state at the time \( t \) and the input vector of size \( r \times 1 \), respectively where \( x^T \) denotes the transpose of the vector \( x \). The input/current state and output/current state relations of the encoder at the time \( t \) can be expressed in the compact form:

\[
S_{t+1} = H_0 \cdot U_t + T \cdot S_t
\]

\[
c^t = H_E \cdot S_t + W \cdot S_t
\]

where the generator matrix \( H_0 \) and the matrix \( T \) are defined as follows:

\[
H_0 = \begin{bmatrix}
h_{r,m} & \cdots & h_{1,m} \\
\cdots & \cdots & \cdots \\
h_{r,2} & \cdots & h_{1,2} \\
h_{r,1} & \cdots & h_{1,1}
\end{bmatrix}
\]

and

\[
T = \begin{bmatrix}
0_{(m-1) \times 1} & I_{m-1}
\end{bmatrix}
\]

with

\[
H_E = [h_{r+1,1}, \ldots, h_{r+1,2}, h_{r+1,1}]
\]

In order to have a decodable code [10], we assume that \( H_0 \) is full rank. In (2), the vector \( W \) is equal to \( [0 \ 0 \ \ldots \ 0 \ 1]_s \cdot u_r \) and:

\[
H_E = [h_{r+1,1}, \ldots, h_{r+1,2}, h_{r+1,1}]
\]

In order to compare this canonical form based on LSFR with the observer canonical form presented in next section, we determine the transfer function matrix of the encoder. Let define the Laurent series of the sequence \( x_i \) as \( X(D) = \sum_{j=0}^{\infty} x_i \cdot D^j \). Assuming that the matrix \( I_m + D \cdot T \) is invertible, i.e. \( \det(I_m + D \cdot T) = h_0(D) \neq 0 \) with \( h_0(D) = \sum_{j=0}^{m} h_{ij} \cdot D^j + 1 \), the transfer function which corresponds to the redundant sequence \( C(D) \) of the encoder and the global transfer function matrix of the code \( M_0(D) \) are respectively equal to:

\[
C(D) = M_0(D) \cdot U(D)
\]

\[
M_0(D) = D \cdot H_E \cdot (I_m + D \cdot T)^{-1} \cdot H_0 + W \cdot (I_m + D \cdot T)^{-1} \cdot H_0
\]

After some algebraic manipulations, it can be shown that:

\[
(I_m + D \cdot T)^{-1} = \frac{1}{h_0(D)} \cdot P_m(D) \cdot H_R \cdot \Delta_m(D) + D^{-1} \cdot \Delta_m(D)
\]

where \( P_m(D) = [D^{m-1} \ldots D \ 1] \) and \( \Delta_m(D) \) is a Toeplitz matrix with first row \( [D \ D^2 \ldots D^{m-1} \ D^m] \) and first column \( [D^0 \ldots 0 \ 1]' \). Therefore, the transfer function matrix \( M_0(D) \) can be simplified as:

\[
M_0(D) = \begin{bmatrix} \frac{h_{r+1,1}(D)}{h_0(D)} & H_R + H_E \cdot \Delta_m(D) \cdot H_0 \end{bmatrix}
\]

\[
+ [h_{r,1} \ldots h_{2,1} h_{1,1}]
\]

In the next section, we introduce a second canonical form for multiple-input encoders based on Fibonacci representation.

III. OBSERVER CANONICAL FORM OF CONVOLUTIONAL ENCODERS WITH MULTIPLE INPUTS

A recursive and systematic convolutional encoder with multiple inputs (MIRSC) is generally not decomposable into \( r \) single-input encoders, i.e., this encoder can generally not be represented by an equivalent structure with one shift register for each input. However, we show in this section that a realizable structure consists to have one shift register for the single output \( c \) as shown in Fig. 2. In system theory, this canonical form is referred as the observer canonical form. [10] Throughout the paper, for sake of simplicity, we refer this scheme as the canonical form of type ‘G’.

![Fig. 1](image1.png)

![Fig. 2](image2.png)
Moreover, manipulations and by defining decodable code, we assume that \( G_r \) and \( W \) are defined as:

\[
G_r = \begin{bmatrix} g_{r,m} & g_{r,m-1} & \cdots & g_{1,m} \\
               g_{r,m-1} & g_{r,m-2} & \cdots & g_{1,m-1} \\
               \cdots & \cdots & \cdots & \cdots \\
               g_{1,1} & g_{1,2} & \cdots & g_{1,1} \\
\end{bmatrix}
\]

(14)

and \( G_r = [g_{0,m} g_{0,m-1} \cdots g_{0,1}]^T \). After some algebraic manipulations and by defining \( g_0(D) \) as

\[
g_0(D) = \sum_{j=0}^{m} g_{j} D^j
\]

it can be shown that the transfer function \( C(D) \) of the encoder is equal to:

\[
C(D) = G_L \cdot U(D) + W \cdot S(D)
\]

(15)

\[
M_g(D) = \frac{1}{g_0(D)} \{ D \cdot P_m(D) \cdot G_0 + G_L \}
\]

(16)

After deriving the transfer function matrices \( M_h(D) \) and \( M_g(D) \) for both canonical forms \( H \) and \( G \) in (9) and (16) respectively, we investigate in the next section the relations of equivalence between the canonical forms \( H \) and \( G \).

IV. EQUIVALENCE BETWEEN ENCODERS DEFINED WITH THE CANONICAL FORMS \( H \) AND \( G \)

First, we define the relation of equivalence between the canonical forms \( H \) and \( G \):

**Definition 1**: The canonical forms \( H \) and \( G \) are equivalent if for any data input \( U(D) \), both canonical forms give the same output \( C(D) \), i.e.:

\[
M_h(D) = M_g(D)
\]

(17)

where \( M_h(D) \) and \( M_g(D) \) are defined in (9) and (16), respectively.

A. Single input classical binary case \((r=1 \text{ and } g_{10}=1)\):

We first investigate the equivalence between both methods in the single input case and when \( g_{10}=1 \) as in the classical case [3].

**Theorem 1** ([12] p.1220): In the case of single input \((r=1)\), both configurations are equivalent.

**Proof**: For \( r=1 \), \( H_0=W^T \). By multiplying both sides of (10) by \( h_0(D) \), we have:

\[
h_0(D) \cdot M_h(D) = (h_2(D) \cdot h_0(D) + h_1(D) - h_0(D)) \cdot h_0(D) = h_2(D).
\]

Since \( G_0 \) is a column vector and \( G_r \) is equal to 1 in the single input case, (16) can be written as:

\[
h_0(D) \cdot M_h(D) = g_0(D) \cdot g_0(D) = g_0(D) \cdot g_0(D) = g_0(D).
\]

Since \( h_2(D) = g_0(D) \) and \( h_0(D) = g_0(D) \) both configurations \( H \) and \( G \) for the classical binary case are one-to-one equivalent.

![Diagram](image)

Fig. 3 The two canonical forms for the 13/158 convolutional RSC

We illustrate this result with the following example. Suppose the RSC with the output and feedback polynomials equal to 158+1+D+D^2 and 138+1+D+D^2, respectively. By defining the full generator matrices \( H \) and \( G \) as follows:

\[
H = [H_L^T \ H_0 \ H_E^T]
\]

(18)

\[
G = [G_0 \ G_R \ 1]
\]

(19)

\[
H \text{ and } G \text{ are equal to } [6 \ 1 \ 5] \text{ and } [13 \ 11] \text{ in this example. The two canonical forms } H \text{ and } G \text{ for the encoder are shown in Figures 3.a and 3.b, respectively.}

Since we have a strict equivalence between both canonical forms \( H \) and \( G \) in the classical single input case, i.e. \( r=1 \) and \( g_{10}=1 \), the optimization of the turbo-codes can be performed either using the canonical
form G or H. However, we show next that it is not true in the case of multiple inputs.

B. Multiple input case (r>1):

We start with the following remark. Let G and H denote the sets of the encoders with canonical forms G and H, respectively. The sizes of the matrices H and G defined in (18) and (19) are $m \times (r+2)$ and $(m+1) \times (r+1)$, respectively. Assuming $g_{0,0}=1$ and $m \geq r$, the matrix H has more entries than G. Therefore the set H is larger than the subset G, i.e. the canonical forms H and G are generally not one-to-one equivalent in the multiple input case.

The conditions for which both canonical forms are equivalent are summarized in the next two theorems.

**Theorem 2:** For any generator matrix H of canonical form H, it exists a unique equivalent matrix G of canonical form G, which is solution of the following system:

\begin{align*}
\alpha) & \quad g_{0,0}=1, \\
\beta) & \quad g_{i,0}=h_{i,1} \quad \text{for any } i=1, ..., r, \\
\gamma) & \quad g_{0,i}=h_{0,i} \quad \text{for any } i=1, ..., m, \\
\delta) & \quad D \cdot P(D) \cdot G_0 = (h_{r+1}(D) \cdot H_k + h_0 \cdot D(D) \cdot H_k \cdot \Delta(D) \cdot H_0 + (h_0 \cdot D)^{+} - [h_{1,1} \ldots h_{1,1} h_{1,1}].
\end{align*}

**Proof:** Equation (\alpha) imposes the structure to be recursive. After some basic manipulations, it can easily be shown that (\beta), (\gamma) and (\delta) are equivalent to (9), (16) and (17). In order to complete the proof, we have to show that (\delta) has a unique solution G_0.

**Lemma 1:** Equation (\delta) is equivalent to the equation: $G_r = A \cdot H_0$ where the $m \times m$ matrix A is determined recursively as follows:

- Initialization: $H_{ER} = [H_E 1]^T \cdot H_R + [H_R 1]^T \cdot H_E$ and $A(1, 1) = H_{ER}(1, 1)$;
- Iteration: $i=2, 3, ..., m; A(i, i) = H_{ER}(i, 1) + A(i-1, 1) \cdot I_m(2)$; where $I_m(p)$ is the $m \times m$ identity matrix shifted by $p-1$ positions to the right. $M(i, i)$ denotes the row vector of the entries of the $i$-th row in the matrix M (Matlab notations).

**Proof:** We have by definition: $h_{r+1}(D) \cdot H_k + h_0 \cdot D(D) \cdot H_k = [D^m D^{m-1} \ldots D \cdot 1] \cdot (H_E 1)^T \cdot H_R + [H_R 1]^T \cdot H_E \cdot \Delta(D) = I_m(1) \cdot D + I_m(2) \cdot D^2 + ... + I_m(m) \cdot D^m$. Using (\beta), (\gamma) and (12), we can now write:

$$m \sum_{i=1}^{m} D^i \cdot G_T(i, :) = \sum_{i=1}^{m} \sum_{p=1}^{m} D^{i+p-1} \cdot H_{ER}(i, :) \cdot I_m(p) \cdot H_0$$

or equivalently term by term:

$$G_T(i, :) = \sum_{k=1}^{m} H_{ER}(i, :) \cdot I_m(i-k+1) \cdot H_0 =$$

$$\left[ H_{ER}(i, :) + \sum_{k=1}^{i-1} H_{ER}(i, :) \cdot I_m(i-k) \cdot I_m(2) \right] \cdot H_0.$$  \hfill (21)

We define A as: $A(i, i) = \sum_{k=1}^{i} H_{ER}(i, :) \cdot I_m(i-k+1)$.

Using (21), the $i$-th row of A can be expressed as:

$$A(i, :) = H_{ER}(i, :) + a(i-1, :) \cdot I_m(2)$$

and $G_T(i, :) = A(i, :) \cdot H_0$, $i=1, 2, ..., m$ or equivalently: $G_T = A \cdot H_0$.

According to the Lemma 1, for any encoder defined by its generator matrix $H_0$, it exists a unique generator matrix $G_T$ and therefore a unique equivalent encoder with a canonical form G which completes the proof of Theorem 2.

**Theorem 2** shows that for any matrix H there is an equivalent matrix G. We propose next to investigate if the converse is true. We start with the following example: Assume $m=r=1$ and a generator polynomial $G = [2 3]$. The corresponding encoder is depicted in Fig.4. This example does not belong to the classical binary case because $g_{10} \neq 0 \neq 1$. We have successively: $G_r = [1] = H_{ER}, G_r = [0] = H_{0}, H = [H_E^T H_0^T H_R^T]$ and $G_T = [x 0]$. It is easy to show that the condition (\delta) is $\{1 \equiv 1+x \equiv 0\}$ which cannot be satisfied for at least one value of $x \in \{0, 1\}$. Thus, the encoder presented in Fig.4 cannot be represented in the canonical form H.

**Theorem 3:** For any generator matrix in G, it exists i) none, ii) one or iii) several equivalent generator matrices in H.

**Proof:** A generator matrix H in H which has an equivalent G verifies the system of equations: ($\beta$) $[h_{1,1} h_{1,1} ... h_{1,1}] = G_2$; (\gamma) $H_0 = G_0$; (\delta) $A \cdot H_0 = G_T$ (Lemma 1). In order to find the vector $H_0$ and the matrix $H_0$ that verified (\delta), a exhaustive computational search is performed through the $2^{m-1} r^m$ possible pairs {$H_0, H$}. Equation (\delta) has i) none, ii) one or iii) several solutions:

i) We have shown in the previous example that an encoder in G may not have an equivalent in H. In section 5.A, we provide several examples.

**ii** There is a unique solution for the single input classical binary case as shown in Theorem 2. Moreover, for any $r>1$, one could find a unique solution $H_0$ to (\delta) for particular values of $G_T$.

**iii** Clearly we have: $|H| \geq |G|$, where $|.|$ denotes the cardinal number of a set. However, from Theorem 2, we know that any generator matrix in H has a unique equivalent in G. Thus, there is at least two encoders in H with the same equivalent encoder in G.

According to Theorem 3, the system of equations (\alpha), (\beta), (\gamma) and (\delta) establishes a function of equivalence $\hat{\xi}: H \rightarrow G$, which is neither injective nor subjective.

Theorem 3 has a huge impact for the design of TC. Indeed, searching a good encoder in G instead of H is not only faster according to the condition (iii) but can furthermore lead to a better solution which does not exist in H according to the condition (i). Furthermore, we show in the following theorem that there are no distinct encoders in G that are equivalent to each other.
Theorem 4: \( \forall \{G^1, G^2 \neq G^1\} \in G, G^1 \) and \( G^2 \) are not equivalent i.e. two distinct matrices in \( G \) cannot be equivalent.

Proof: Assume that two matrices \( G^1 \) and \( G^2 \) in \( G \) are equivalent. According to (16) and (17), it implies that:

\[
g_\Omega^1(D) = g_\Omega^2(D), \quad G_R^1 = G_R^2 \quad \text{and} \quad D \cdot P_u(D) \cdot G_Q^1 + G_L^1 = D \cdot P_u(D) \cdot G_Q^2 + G_L^2.
\]

The last equality is verified if only if: \( G_Q^1 = G_Q^2 \) and \( G_L^1 = G_L^2 \), so \( G^1 = G^2 \).

C. Extension to the Multi-Non-Binary Convolutional Codes:

It is worth noting that all previous calculations are valid for binary and non-binary inputs. Therefore, all the results can be generalized for \( (u_n, s_j, h_{ij}, g_{il}) \) in \( GF(2^q) \), \( q > 1 \). They can directly be used for the design of the MNBTC [9]. In that case, each connection corresponds to a \( q \) bits-wide bus; the coefficients \( h_{ij} \) and \( g_{ij} \) correspond to a multiplication circuit over \( GF(2^q) \); the shift registers buffer packets of \( q \) bits, and the adders perform modulo 2 symbol-wise additions.

V. SIMULATION RESULTS

In this section, we present the optimization results of an exhaustive search among all encoders in \( G \), i.e., all encoders that can be represented with observer canonical form.

In order to reduce the computational complexity of the MBTC based on an exhaustive search, we perform the exhaustive search in both following steps: in the first step, we select both codes and interleavers based on a quick estimate of the ratio between the average number of iterations to decode the MNBTC and the number of packets that were transmitted without error for a fixed number of packets. Only few packets, say roughly 1000 transmitted packets for each code, are needed to accurately estimate this ratio. Moreover, we observe in our simulations that the best code in all cases belongs to the first per cent only of the codes with fastest convergence which dramatically reduces the code optimization. In the second step, we are simulating FER at a SNR such that the FER results approximatively match with a targeted FER. In our case, we target a FER around 10^{-4} and are using 500000 codewords to estimate it: The best code is the code with the smallest FER.

For implementation considerations, we focus on double-binary turbo-codes (MBTC with two inputs) with memory \( m=2 \) and 3. In all cases except if it is notified, we use an S-interleaver of length 752 the trellis-termination technique using tail-bits to drive the encoder to the all-zero state; moreover, the coding rate is 1/2 for all MBTC.

For \( m=2 \), the best code that we found has for generator matrix \( G=[7 \ 6 \ 5] \) which does not have an equivalent in canonical form.

For \( m>2 \), the codes that belong to \( G/H \) do not have a primitive feedback connection polynomial \( g_\Omega(D) \) of...
maximal degree \( m \). Since the best MBTC are based on a primitive feedback polynomial, it is not necessary to consider the codes that belong to \( G/H \) in the MBTC design. The canonical form \( G \) is still interesting for the MBTC design in order to simplify the exhaustive search based on Theorem 4.

For \( m=3 \) and an S-interleaver of parameter 22, the best MBTC that we found is the code proposed in [8] with generator matrix \( G=[13 \ 15 \ 11] \). Whereas the S-interleaver is asymptotically optimal in the codeword size [15], we also consider the family of interleaver proposed in [8] which is simpler to implement in hardware. These interleavers are defined by 4 parameters: \( P, P_1, P_2, \) and \( P_3 \). For more details on the construction of such interleavers, we refer to [8]. Through an exhaustive search, we found two best codes with generator matrices \( G=[15 \ 9 \ 11] \) and \( G=[15 \ 9 \ 13] \) and with interleavers defined by the parameters \( P=19, \ P_1=376, \ P_2=328, \ P_3=196 \) and \( P=19, \ P_1=376, \ P_2=203 \) and \( P_3=677 \), respectively. Both codes have similar Bit Error Rate and Frame Error Rate performance and outperforms the MBTC proposed by [8]. For a frame error rate of \( 10^{-4} \), the gain is approximatively equal to 0.3 decibels. Finally, for \( m=4 \), the best code that we found has for generator matrix \( G=[21 \ 23 \ 25] \) which is the same that Douillard proposed in [8].

VI CONCLUSIONS

The design of the turbo-codes with multiple inputs generally requires an exhaustive search through large sets of feasible codes and interleavers. This search is a major burden in their optimization since the computational complexity of the search is exponential with the number of inputs \( r \).

In this paper, we showed that the encoder of such codes could be represented with the observer canonical configuration. This configuration is essential for reducing the computational complexity of the search (up to 300\% for the codes that we considered). Based on this strategy, we were able to design a rate-1/2 turbo-code with two inputs and memory \( m=3 \) which outperforms the equivalent turbo-code proposed by Douillard et al. by 0.25 decibels for a frame error rate of \( 10^{-4} \).

ACKNOWLEDGMENT

The authors are grateful to Professors C. Douillard, Ph.D and M. Jézéquel., Ph. D, from ENST-Bretagne for the Turbo Codes seminars sustained in Timișoara, Romania.

REFERENCES