8th International Conference on DEVELOPMENT AND APPLICATION SYSTEMS
Suceava, Romania, May 25 – 27, 2006

SERIAL CONCATENATED CONVOLUTIONAL CODES DESIGN AND ANALYSIS

Luminita SCRIPCARIU
Technical University "Gh.ASACHI" of Iaşi, ROMANIA
Bd. Copou 11, RO – 700506, IASI
[1]luminita.scripcariu@gmail.com

Abstract. *Serial Concatenated Convolutional Codes (SCCC) also named Turbo Codes, obtained with different constituent trellis codes are presented with a simple and quick mode of data interleaving. A SCC code has higher error-correction performances than the constituent codes. Optimal convolutional codes are deduced for SCCC. A comparison of the SCCC performances is made using weight enumerating functions.*
Keywords: *serial concatenated convolutional code, error-correction, weight enumerating function.*

Serial Concatenated Convolutional Codes Principles

A Serial Concatenated Convolutional Code (SCCC) also named a Turbo Code is obtained based on a simple constituent trellis code applied twice on the data input, first on the direct data sequence and second on the interleaved one [1]. When two trellis coding techniques are applied on some interleaved data symbols, a so-called **Serial Concatenated** or **Turbo code,** with an increased error-correcting capacity, results (Figure 1).

D11	H 11	D12	H 12	D13	H 13
V11	A11	V12	A12	V13	A13
D21	H21	D22	H22	D23	H23
V21	A21	V21	A21	V23	A23
D31	H31	D32	H32	D33	H33
V31	A31	V32	A32	V33	A33

Figure 1. SCCC encoding Principle.

Two systematic codes are applied on each row of the data matrix (D) to deduce the horizontally parity matrices (H_i), then column-by-column to determine the vertically parity matrices (V_i) and the optional auxiliary parity matrices (A_i).
If the same constituent code is used for row- and column-encoding, a symmetric Turbo code results.
The input data segment length is established based on the decision length (L) of the convolutional code, defined as the necessary number of levels in which Viterbi algorithm could make a decision and the trellis state could be reinitialized.
In Figure 1, a symmetric SCCC based on a binary convolutional code with a decision length equal to 3 is considered.
The systematic convolutional code with a coding rate k:n is first applied on each row of the data matrix. The horizontal parity matrices result.
The second encoding process applies the code on each column of the row-encoded matrix. The vertically and auxiliary parity matrices are obtained. In this way, a serial concatenation is made using this data structure and the resulting code is a SCCC.
Each $\boldsymbol{k^2}$ –symbols input-word is converted into a D matrix and the parity symbols are deduced. Then the coded matrix is converted into an $\boldsymbol{n^2}$ – symbols codeword.
The coding rate of the symmetric SCCC based on a systematic ***k:n*** code is given by:

$$R_T = \frac{k^2}{n^2} \tag{1}$$

(***k*** is the input-word length; ***n*** represents the codeword length.)

It is easier to use **the equivalent coding rate** of a SCC code [3]:

$$R_{Te} \stackrel{\Delta}{=} \frac{w}{w+1} \quad (2)$$

This definition is based on the integer *w,* computed as:

$$w = \left\lfloor \frac{R_T}{1-R_T} \right\rfloor \quad (3)$$

(We denote by [x], the floor of the number x.)

The 'tree' codes (trellis or convolutional) are obtained first with non-systematic and non-recursive coding (NNC) structures. In order to apply a trellis code for a turbo encoding technique, a recursive systematic coding (RSC) structure must be used. This is easily done for the half-rate convolutional codes but these one could not be used in a turbo code because the redundancy increased by 300%. Therefore for SCC codes we have to choose a convolutional code with a coding rate of 2:3 or 3:4.

Optimum Convolutional Codes

The systematization of a convolutional code with higher dimensions (coding rate or constraint length) becomes difficult and the Viterbi decoding algorithm is too complicated for a real-time application.

So we are interested in 2:3 or 3:4 systematic convolutional codes with only one memory cell for each input, which generate half-rate TURBO codes (1:2).

The initial trellis code with a coding rate k:n is generated by a non-recursive and non-systematic structure described by the generating matrix G:

$$G = [g_{ij}(x)]_{i=1:k,\, j=1:n} \quad (4)$$

The elements of G are constant or higher order polynomials.

The polynomial variable is the delay operator D (for binary case) or x (in the general case of the GF symbols).

The code will be denoted based on the vector of coefficients of the generating polynomials, decreasing the variable power.

There are a lot of 2:3 binary symmetric convolutional codes with two memory cells. We have to find out the non-catastrophic codes which admit unique decoded output and to select the optimum one, deduced based on the minimum Hamming distance criteria.

These optimal codes are non-recursive and non-systematic except the first one (Table 1). The half-rate codes are not interesting for SCC codes. The 2:3 codes must be systematized if they are used in SCC coding algorithm.

In Table 1, we denoted by m the encoder memory cell number, by R_c the convolutional coding rate and by d_{Hfree} the free Hamming distance of the convolutional code. For optimal properties, the encoder structure is symmetric regarding to the memory cells. It means that the same memory cell number is used for each input variable.

Table 1. Optimum Binary Convolutional Code with Maximum 4 State Variables

Convolutional Code	**m**	**R_c**	**d_{Hfree}**
C(3,1)	1	1:2	3
C(31;13;12)	2	2:3	4
C(7;11)	3	1:2	6
C(71;53;35)	4	2:3	5
C(11;18;6;52)	6	3:4	7

The generating matrices of some non-catastrophic codes are given below:

1. C (31; 13; 12)

$$G1_{NNC}(x) = \begin{bmatrix} 1+x & 1 & 1 \\ 1 & 1+x & 1+x \end{bmatrix} \quad (5)$$

$$G1_{RSC}(x) = \begin{bmatrix} 1 & 0 & \frac{1+x^2}{1+x+x^2} \\ 0 & 1 & \frac{1+x}{1+x+x^2} \end{bmatrix} \quad (6)$$

The RSC (31; 13; 12) has the modified generating matrix, delayed by two bits in order to be causal and non-recursive:

2. C (71; 53; 35)

$$G2_{NNC}(x) = \begin{bmatrix} 1+x+x^2 & 1+x^2 & 1+x \\ 1 & 1+x & 1+x^2 \end{bmatrix} \quad (7)$$

$$G2_{RSC}(x) = \begin{bmatrix} 1 & 0 & \frac{x+x^2}{1+x^2} \\ 0 & 1 & \frac{1+x}{1+x^2} \end{bmatrix} \quad (8)$$

Based on relations (6) and (8) the systematic coding structures result with minimum number of state variables (Figure 2 and 3).

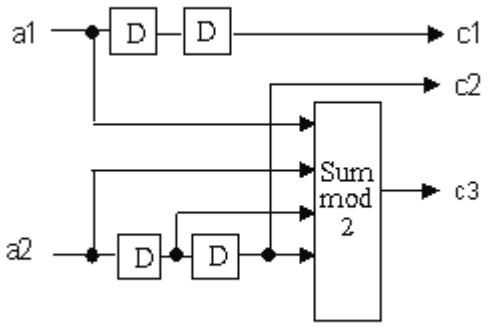

Fig.2 NSC(31;13;12) Encoder

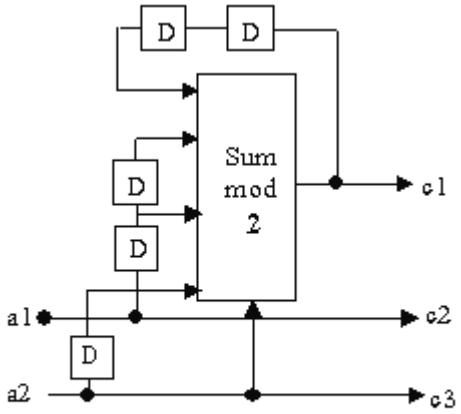

Fig.3 RSC(71;53;35) Encoder

SCCC Parameters

Let us denote:
r - the redundant increase of word length by the encoding process.
t - the maximum number of errors which can be removed from a codeword by
the block-code.
b - the maximum length of an error-burst which is recovered from fading by the error-correcting code.
γ - the code error-correcting capacity as percentage
L - the Viterbi decoder decision length
g - the coefficients vector of the convolutional code
SCCC (*g*) - a symmetric SCCC based on a binary convolutional code C (*g*).

If a systematic k:n constituent code could correct ***t*** errors in one ***n***-bits word, then the associate SCCC code is able to correct error-burst of ***nt*** consecutive bits in a n^2 –bits received word [3]:

$$b_{SCCC} = nt \ \ (bits) \qquad (9)$$

For example, a H (7, 4) code corrects one bit-error from a 7-bit received word. The associated TH (7, 4) code could correct 7 consecutive bits in a 49-bit received word. But it has a low equivalent coding rate of about 1:3. Hamming and Turbo-Hamming codes work on the binary Galois field and high word lengths result.
The convolutional code C (7, 11) code corrects two bit-error from a 14-bit received frame. The associated SCCC (7, 11) code could correct 28 consecutive bits in a 196-bit received word. It also has a low equivalent coding rate of about 1:3.
Some SCCC parameters are presented below, in Table 2.

Table 2. SCCC Parameters

SCC Code Name	**SCCC (31;13;12)**	**SCCC (71;53;35)**	**SCCC (11;18;6;52)**
n^2	9	9	16
k^2	4	4	9
R_{Te}	1:2	1:2	1:2
r (%)	125	125	77.7
t	1	2	3
b_{sccc}	9	18	48
γ (%)	11.1	11.1	6.45

Higher dimensions SCCC have a reduced redundant increase of length and an improved error-burst correction capacity which ensures a good quality of digital communications even for high channel error-rate of about 10^{-2}, at low SNR.

Fast Decoding Algorithm

The Turbo decoder works iterative and uses the increased redundancy of the code words to correct random errors or packets of errors so the Turbo code has a high error-correcting capacity.

Unfortunately the coding and decoding software algorithms are more complex for higher code dimensions. The decoding complexity is critical for real-time application. Some optimal detection and estimation algorithms must be applied to reduce the decoders' decision time.

The turbo-processing time is higher than for a simple decoder but a compromise time-quality could be made restricting the iterations number of the turbo-decoder.

The operations number made on a trellis level in the decoding process is:

$$N = S \cdot I = 2^{mM} \cdot 2^{mk} \qquad (9)$$

Note:

S - is the states number of the coder;

I - is the input edges number into a specified node of the trellis-diagram;

m - is the length of the binary combination associated with a GF element.

(m=2 in the binary case);

M - is the memory cells number;

k:n - represents the coding rate.

There are made *N/k* operations for every decoded symbol and *N/n* operations for every corrected symbol using the Viterbi algorithm.

The classic Viterbi algorithm [5] is applied with *N* operations on every level of the coding trellis-diagram except the first level, at which is applied with only one state. Based on this fact, the decoding process may be partitioned. So, after every decision an initial state with null weight will be defined and the next level will be treated as the first one, with a reduced number of decoding operations. It is necessary to make a decision only after minimum *L* levels starting from the initial state. This *modified Viterbi algorithm* [6] can be used for binary or multi-bit trellis codes (Figure 4).

A Turbo code could be described by some weight-enumerating functions [1], [2].

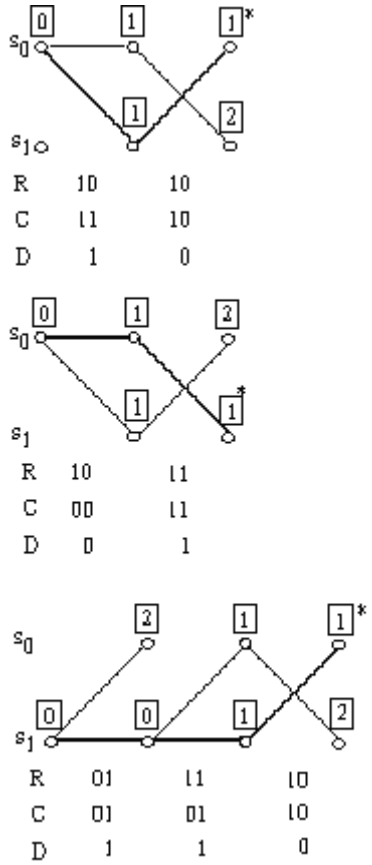

Figure 4. The decoding process of C (3; 1) code using the modified Viterbi algorithm (R – received sequence; C – corrected sequence; D – decoded sequence).

SCCC Performances Estimation

A SCCC code could be described by the enumerating functions using the equivalent block-code (EBC) of the convolutional one.

The **Input Output Weight Enumerating Function** (IOWEF) is defined as it follows, based on the number of code words ($A_{w,h}$) with weight *h* associated with an input word of weight *w*:

$$\mathrm{IOWEF(W,H)} = \sum_{w=0}^{k}\sum_{h=0}^{n} A_{w,h} W^{w} H^{h} \qquad (10)$$

The **Conditional Weight Enumerating Function** (CWEF) is defined as:

$$\mathrm{CWEF(H)} = \sum_{w=0}^{k} \frac{w}{k} \sum_{h=0}^{n} A_{w,h} H^{h} =$$

$$= \frac{1}{k}\sum_{h=0}^{n} H^h \sum_{w=0}^{k} wA_{w,h} = \sum_{h=0}^{n} D_h H^h \tag{11}$$

The coefficients of the IOWEF could be grouped into a matrix ***A*** which describes very well the code.
We denote by D_h a multiplicity of code words with weight h:

$$D_h = \frac{1}{k}\sum_{w=0}^{k} wA_{w,h} \tag{12}$$

These coefficients form a vector ***D*** of values which also describes the code.
The **Weight Enumerating Function** (WEF) is defined as:

$$\mathrm{WEF(H)} = \sum_{h=0}^{n} B_h H^h = 1 + \sum_{h=h_m}^{n} B_h H^h \tag{13}$$

where h_m is the minimum weight of the code words excepting the all-zero word and B_h is the number of code words with weight h.
In fact, the weight of the code words could have only some values included into the **weight profile** of the code:

$$\Omega_h = \{h_1, h_2, \ldots h_M\} \tag{14}$$

(M is the number of distinct codeword weight values).
So the WEF could be expressed as:

$$\mathrm{WEF(H)} = \sum_{h\in\Omega_h} B_h H^h \tag{15}$$

An upper bound of the Bit-Error-Rate (BER) of a block Turbo Code could be expressed using the same weight enumerating functions [2] but the computation complexity could exceed the available memory for high code dimensions.
The IOWEF coefficients of the SCC code are computed based on the A – matrix of the convolutional constituent code as it follows:

$$A_{w,h}^{SCC} = \sum_{q=t+1}^{N} \frac{A_{w,q} A_{q,h}}{\binom{N}{q}} \tag{16}$$

The upper bound of BER could be expressed using the CWEF of the SCC code [1], [2]:

$$BER \le \sum_{h\in\Omega_h} D_h H^h \Big|_{H=\exp(-RE_b/N_0)} \tag{17}$$

R is the coding rate and E_b / N_0 is the signal-to-noise ratio (SNR).
When high dimensions codes are analyzed, the upper bound of BER has a great number of terms and only some of them are significant. Therefore a good approximation is obtained if only the first terms of the sum, starting with the minimum codeword weight, are computed.
The coding gain is deduced as:

$$G = 10Log_{10}(h_m R)\,[dB] \tag{18}$$

The algorithm for the BER upper bound estimation should compute the coefficients of the IOWEF of the Turbo code for at least [h_m/2] values of the input word weight (w) in order to deduce the correct value of the multiplicity D_{hm} corresponding to the most important term of the sum (see (17)). For input weight greater than [h_m/2], the minimum codeword weight is not obtained any more, so the value of D_{hm} is completely deduced.
The algorithm for BER upper bound estimation will estimate the IOWEF coefficients with the input word weight varying from 1 to [h_m/2].
SCC codes have good performances on error-correcting.

Example 1:
Let us consider the 1:2 C (3, 1) code with a decision length of 2 frames and its associate 1:3 SCCC (3, 1) with an interleaver length of N = 8.
The A matrix of the C (3, 1) code is written using the 2:4 EBC with 4 bits-code words:

$$A_{3,1} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

The D-vectors of these codes are:
$D_{C(3,1)} = [0, 0, 2, 2]$
$D_{SCCC(3,1)} = [0, 0, 0.1429, 0.1429]$
Example 2:
Let us consider another convolutional code 1:2 C (7, 11), with a decision length of 7 frames and a free Hamming distance of 6, and the associated 1:3 SCCC (7, 11) code with an interleaver length of N = 98 bits.
The A matrix of this code is written using the 7:14 EBC with 14 bits-code words.

$A_{C(7,11)}$= [1,0,0,0,0,0,0,0,0,0,0,0;
0,0,1,0,1,1,4,0,0,0,0,0;
0,0,1,1,0,2,4,1,6,1,3,1;
0,0,0,3,1,2,4,3,6,5,6,1;
0,0,0,1,2,4,2,7,3,4,2,0;
0,0,0,0,1,2,0,3,1,4,0,0;
0,0,0,0,0,0,0,0,0,1,0,1]

The D-vectors of both C (7, 11) and SCCC (7, 11) codes are computed based on (12):

$D_{C(7,11)}$ = [0, 0, 0.4286, 2.1429, 2.4286, 5.2857, 4.5714, 7.7143, 6.7143, 8.4286, 4.5714, 1.5714]

$D_{SCCC(7,11)}$ = [0, 0, 0, 0.3006, 0.1086, 0.2172, 0.4039, 0.3304, 0.6064, 0.5141, 0.6011, 0.0987]

Based on the deduced D-vectors, the decoders BER is estimated (Table 3).

Table 3. BER upper bound of some convolutional and SCCC decoders

Code / SNR (dB)	**C (3, 1)**	**SCCC (3, 1)**	**C (7,11)**	**SCCC (7,11)**
3	0.1362	0.0017	0.2665	7.63E-6
6	0.0056	4E-4	0.0143	1.07E-7
9	1.25E-5	8.9E-7	1.5E-4	3.8E-18
12	7.5E-11	5.3E-12	4.8E-8	4.8E-32

The SCCC (7, 11) decoder could work for lower SNR, for example its BER is about 10^{-5} for a SNR of about 1 dB.

Conclusions

A simple design mode of SCC codes is presented. A modified Viterbi decoder is used to reduce the decoding time. The upper bound of BER is deduced based on the conditional weight enumerating function (CWEF) of the code. Numerical results are presented for some binary SCCC and a comparison of their performances is made.

References

[1] Divsalar D. (1996) *Turbo Codes*, MILCOM 96 Tutorial

[2] Benedetto S. (1996) *Turbo Codes*, ICC'96 Tutorials, Dallas USA, June 23 – 25.

[3] Scripcariu L., Cotae P. *About Turbo-Codes for Error-Correction*, Proceedings of the International Symposium on Signals, Circuits and Systems 'SCS'2001', Iasi, Romania, 10-11 July, 2001, pp.521 – 524

[4] Sklar B. (1997) *A Primer on Turbo Code Concepts*, IEEE Comm. Magazine, Vol.35, No.12, pp.94-102.

[5] Viterbi A.J., Omura J.K. (1979) *Principles of Digital Communication and Coding*, McGraw-Hill, New-York.

[6] Scripcariu L., Duma P. *Modified Viterbi Algorithm for Convolutional Codecs Defined on Galois Field*, Buletinul Institutului Politehnic Iasi, Sect. III 'Electrotehnica, Energetica, Electronica', Fasc.3-4, 1999.