行政院國家科學委員會專題研究計畫成果報告
具延遲處理器及訊號點對應器之迴旋碼的進一步研究
Further Study on Convolutional Codes with a Delay Processor and a Signal Mapper

計畫編號：NSC 90-2213-E-002-064
執行期間：90 年 8 月 1 日至 91 年 7 月 31 日
主持人：林茂昭 國立臺灣大學電信所

一、中文摘要

在[1,2]中提及一種迴旋碼 T，其架構為在低編碼速率迴旋碼 C 後接上一個延遲處理器(delay processor)及訊號點對應器(signal mapper)，因為延遲處理器的加入，使得該迴旋碼 T 之限制長度(constraint length)比低編碼速率迴旋碼 C 大很多(即使迴旋碼 C 的限制長度很小)，在此情形下，迴旋碼 T 可具有更大的自由距離，其解碼可使用 C 的籬柵(trellis)及解碼過的訊息來解碼，然而，此次佳化解碼(suboptimally decoding) [1,2]會導致大錯誤係數，使得除錯性能降低。再者，在短封包傳輸通訊中，過多的補零動作也會使得編碼速率因而降低。

在此計畫中，我們使用重覆性解碼及 Tail Biting 來克服消除上述次佳化解碼所造成的錯誤係數及編碼速率降低問題，並專注於改良次佳化解碼[1,2]。研究成果顯示我們所設計的籬柵碼在短碼長時比二位元渦輪碼具有更佳除錯性能表現。除此之外，我們還設計此籬柵碼之解碼電路來驗證 Tail Biting T 之優越性。

英文摘要

In [1,2], a convolutional coding scheme which is implemented by serially concatenating a delay processor and a signal mapper to the encoder of a conventional convolutional code C was proposed. Since the introduction of the delay processor, the constraint length of resultant code T is large even if the constraint length of C is short. In this manner, T can achieve large free distance and can be suboptimally decoded by using the trellis of C with the help of some previously decoded information. However, the suboptimum decoding used in [1] and [2] results in a large error coefficient and hence degrades the error performance. Moreover, for short packet transmission, the rate loss resulted from zero-tail method is large.

In this project, we use iterative decoding and the tail-biting technique to solve these problems. We will focus on improving the suboptimal decoding used in [1] and [2]. Binary codes constructed from our design has better error performance and lower decoding complexity as compared to turbo codes [4] in case of short codeword length. We also provide circuit design of the suboptimal decoder of the proposed trellis code to verify its advantage.

二、計畫的緣由與目的(Goals)

The encoding of trellis codes given in [1,2] is shown in Fig.1. It is already known that by a trellis code T which is constructed by using the encoder of a convolutional code C with short constraint length followed by a delay processor and a signal mapper is equivalent to a trellis code with large constraint length. In [1] and [2], a bound on free distance of T is derived and a suboptimum decoding is used to achieve error performance guaranteed by the free distance bound. However, the suboptimum decoding used in [1] and [2] will results in a large error coefficient and dense distance spectrum and hence the associated error performance is very poor at low to moderate signal-to-noise ratio conditions. To overcome these problems, in this project we propose an improved suboptimal decoding based on the concept of turbo decoding (iterative decoding). When using the delay processor given in [1,2] and not using the interleaver as used in [1], we encounter severe error propagation. To mitigate this error propagation, we employ various weight factors to reflect the various influences of the feedback extrinsic information among various coding level. Moreover, for short packet transmission, the rate loss resulted from terminating the trellis of T in the all zero state by appending some zeros to a message frame, i.e., the zero-tail method, is large. The rate loss problem resulted from zero-tail method can be avoided by using the tail-biting design [3]. If we directly apply the conventional tail-biting design as shown in [3] to T, then the decoding is extremely difficult. We propose an alternative method to design the tail-biting trellis code of T. With this, both the encoding and decoding can be easily implemented. Simulation results show that the trellis code with a delay processor and a signal mapper can achieve error performance better than that of binary turbo code at moderate to high signal-to-noise ratios (or moderate to low bit error rate) in case of short decoding delay.

三、研究方法與成果 (Methods and Results)

Tail-Biting Trellis Codes

For trellis codes with a delay processor, i.e., T, conventional zero-tail encoded trellis code will result in a lot of fractional rate loss for short packet transmission since the number of memory bits in the encoder of T, $v' = v + \sum_{j=1}^{m} \tau_j$, is very large for typical values used in [2]. Hence tail-biting trellis code is highly desired for short packet transmission. The trellis of tail-biting trellis code is different from that of zero-tail trellis code. For zero-tail code, the trellis starts from the zero state and ends in the zero state while for tail-biting code, the trellis can start from any possible state and end in the starting state. To obtain the associated tail-biting code of T, we can use the linear feed forward shift register of the encoder of C' for which the generator matrix is $G_c(D)Q(D)$, where $G_c(D)$ and $Q(D)$ are transfer function matrices of the convolutional code C and the delay processor, respectively. With the zero-tail encoding, the shift register is initialized with zeros. To achieve a tail-biting code T' in a conventional way[3], the shift register is initialized with the last $v' = v + \sum_{j=1}^{m} \tau_j$ message bits. This ensures that the encoder of C' will start from and end in the same state [3]. However, with this tail-biting method, it is difficult to decode T' efficiently.

1. Encoding

We now propose to design the tail-biting trellis code T'' in a way which is suitable for the suboptimal decoding that will be described later. Let the packet size of message bits be L× λ × r, i.e., L× λ message symbols, where L and λ are positive integers. To obtain T'', we can initialize the linear feed forward shift register of the encoder of C with last v message bits. We then feed the whole message frame to the encoder of C as usual and arrange the resultant code bits $v_j(t), j = 1,\ldots,m$ *and* $t = 0,1,\ldots,L\lambda - 1$, in an m×L λ matrix. Let the entry at the jth row and tth column of the m×L λ matrix be $s_j(t-1)$. The relation between $v_j(t)$ and $s_j(t)$ is as follows

$$v_j(t) = s_j((t + \sum_{l=j}^{m} \lambda_j) \bmod L\lambda)$$

$$= s_j((t + \tau_j) \bmod L\lambda) \qquad \textit{for} \quad 1 \le j \le m$$

Then $\hat{s}(0),\ldots,\hat{s}(L\lambda - 1)$ are fed to the signal mapper to yield the desired codeword $z(0),\ldots,z(L\lambda - 1)$ of T''. If we define the state of T'' as the contents of the memory elements of the convolutional encoder of C and the delay processor, then the encoder of T'' will start from and end in the same state and hence the code T'' is indeed a tail-biting trellis code. We prove the following theorem.

Theorem 1. If L and λ are large enough and T is noncatastrophic, then the minimum distance $d_{\min}$ of the associated tail-biting code T'' is the same as the free distance of T and is upper bounded by $d^{u}_{\min} = \min_{\bar{v} \in C,\ \hat{v}(0) \ne \overset{\wedge}{0}} \sum_{j=1}^{m} \sum_{t=0}^{\infty} v_j(t)\delta_j$.

2. Decoding

The proposed decoder consists of two modules. One is the decoder of C and the other is a demapper. The decoder of C employs the tail-biting trellis of C where the "time axis" has the topology of a circle rather than that of an interval. With this circular trellis, we can run the sliding window Log-BCJR algorithm [5] continuously without any boundary effect as the trellis level or decoding time increases. The decoder of C will yield extrinsic value $L_e^{(I)}(v_i(t))$ of code bits of C after a window size of λ time units (symbols) rather than a block size of $L\lambda$ time units (symbols). Note that these code bits are also used as labeling bits $s_i(t)$ of the signal mapper and hence these extrinsic values can be used as a priori values $L_e^{(I)}(s_i(t))$ of these labeling bits. The demapper uses these a priori values and the received symbol y(t) which is the error-corrupted form of z(t) as input and produces the a posteriori value and extrinsic value of labeling bits $s_j(t)$ as described in the following. Let $\| y-z \|^2$ represent the squared Euclidean distance between symbol y and symbol z. If y or z is a binary m-tuple, then a bit "0" must be replaced by "-1" and a bit "1" remains to be "1". For j = 1, 2, … , m and i = 0, 1, define set $B_j^i = \{w(s_1(t),\ldots,s_j(t) = i, s_{j+1}(t),\ldots,s_m(t)) \mid s_q(t) \in \{0,1\} \mathit{for}\ q \neq j\}$ and function

$$\overset{*}{\max_{b\in B}}(f(b)) \equiv \ln\{\sum_{j=1}^{J}\exp(f(b_j))\} \approx \max_{b\in B}(f(b)) + \delta(f(b_1), f(b_2),\ldots f(b_J)),$$

where f is a function of b and $B = \{b_j\ \mathit{for}\ 1 \leq j \leq J\}$ and $\delta(f(b_1), f(b_2),\ldots f(b_J))$ is a correction term that can be computed recursively by using a simple lookup table [6]. If $L_e^{(I)}(s_i(t))$, j=1,2,..,m are independent, then the a posteriori value $L^{(o)}(s_j(t))$ of labeling bit $s_j(t)$ at the demapper output can be calculated according to

$$L^{(o)}(s_j(t)) \approx$$
$$\overset{*}{\max_{x\in B_j^1}}\{-\frac{\|y(t)-w(x)\|^2}{No}\} + \frac{1}{2}\sum_{i=1}^{m}\zeta_{i,j}[2s_i(t)-1]L_e^{(I)}(s_i(t))\}$$
$$-\overset{*}{\max_{x\in B_j^0}}\{-\frac{\|y(t)-w(x)\|^2}{No}\} + \frac{1}{2}\sum_{i=1}^{m}\zeta_{i,j}[2s_i(t)-1]L_e^{(I)}(s_i(t))\}$$

where $\zeta_{i,j} = 1$ for all i and j. If the a priori information of $s_i(t)$ is not available, then $L_e^{(I)}(s_i(t))$ equals zero. The independence of $L_e^{(I)}(s_i(t))$, j=1,2,…,m may be achieved by using a large interleaver between the decoder of C and the demapper. However, we do not use such an interleaver since doing so will increase the decoding delay. Without the interleaver, severe error propagation may occur. We can alleviate this problem by using suitable weighting factor $\zeta_{i,j}$ which reflects the amount of influence of a priori value $L_e^{(I)}(s_i(t))$ on the a posteriori value $L^{(o)}(s_j(t))$. These weight factors can be represented by an m × m matrix Ξ for which the entry at the *i*th row and *j*th column is $\zeta_{i,j}$. The matrix of weighting factors, Ξ, is dependent on the choices of signal mapper and in this chapter is designed by intuition in a trial-and-error way. The extrinsic values of these labeling bits are obtained from $L_e^{(O)}(s_j(t)) = L^{(o)}(s_j(t)) - L_e^{(I)}(s_j(t))$. Then these values $L_e^{(O)}(s_j(t))$ are also used as intrinsic values for code bits of C and are fed to the sliding-window Log-BCJR decoder of C for further processing.

3. Performance Evaluation

In this section, we use specific examples to verify the error performance by simulation. In our simulation, six iterations are used unless the number of iterations is mentioned.

Example 1: Let $r = 2, m = 4$ and $\Omega = \{1,0\}^4$. Let the 4-state ($v = 2$) generator matrix G_C^1 of C, the labeling K_1 and Ξ_1 be respectively given by

$$G_c^1 = \begin{pmatrix} 3 & 0 & 1 & 3 \\ 0 & 3 & 3 & 2 \end{pmatrix}, K_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\Xi_1 = \begin{pmatrix} 1.00 & 0.83 & 0.83 & 0.83 \\ 0.77 & 1.00 & 0.83 & 0.77 \\ 0.77 & 0.50 & 1.0 & 0.71 \\ 0.77 & 0.50 & 0.33 & 1.00 \end{pmatrix}$$

It can be checked that $\{\delta_1, \delta_2, \delta_3, \delta_4\}$ $=\{1,2,3,4\}$ for K_1. Consider cases (a) and (b) for which the parameters are given in Table 1. For both cases (a) and (b), the minimum distance $d_{\min}^{l}$ (for sufficiently large λ and L) is 13. □

Simulation results of binary turbo code are shown in Fig.2 and Fig.3. We consider a random interleaver, 4-state component codes and Log-BCJR (Log-MAP) algorithm [5]. Fig.2 and Fig.3 show that the error performances of Examples 1(a) and 1(b) are better than those of binary turbo codes at moderate to high SNR in the AWGN or Rayleigh fading channels based on the same short codeword length. The inferior error performance of binary turbo codes considered here may be due to that both the "interleaver gain" and minimum distance are small. Note that for binary turbo codes, the Log-MAP algorithm must run twice for each iteration in the iterative decoding. In contrast, for the proposed code, the Log-MAP algorithm runs only once for each iteration.

4. Circuit Design

We complete a circuit design of the suboptimal decoder for a tail-biting code T which is composed of concatenating a rate 2/4 convolutional code followed by a 4-level delay processor and a signal mapper. We use L = 16 and λ = 8 respectively. The code length is 512. The design is suitable for implementation on the Altera FLEX10K200 FPGA chips. Performance measurement indicates the tail-biting code T is indeed superior to the binary turbo codes of comparable lengths.

四、結論與討論(Concluding Remarks)

We study trellis codes with large constraint length which have been constructed from the scheme proposed in [1, 2]. We find better bounds on free distances than those derived in [1, 2]. We propose a decoding method based on the concept of iterative decoding to take advantage of the larger free distance bounds. We consider the tail-biting structure of this trellis code for short packet transmission. Simulations show that in case of short codeword length, the proposed tail-biting trellis coding can perform better than either the turbo coding or the conventional trellis coding. We also provide circuit design to verify the advantage of the proposed code.

五、參考資料(References)

[1] G. Hellstern, "Coded Modulation with Feedback Decoding Trellis Codes," in *Proc. IEEE Int. Conf. on Communications* (Geneva, Switzerland, May 1993), pp.1071-1075.

[2] J.Y. Wang and M.C. Lin, "On constructing trellis codes with large free distances and low decoding complexities," *IEEE Trans. Commun.*, vol.45, no.9, pp.1017-1020, Sept. 1997.

[3] H. H. Ma and J. K.Wolf, "On tail-biting convolutional codes," *IEEE Trans. Commun.*, vol. COM-34, no. 2,pp. 104-111.Feb. 1986.

[4] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Trans. Commun.*, vol. 44, no. 10, pp. 1261-1271, Oct. 1996.

[5] P. Robertson, E. Villebrun and P. Hoeher, "A comparison of optimal and suboptimal MAP decoding algorithms operating in the log domain," in *Proc. ICC'95*,pp. 1009-1013.

[6] S. Le Goff, A. Glavieux and C. Berrou, "Turbo-codesand high spectral efficient modulation," in *Proc. ICC'94*, pp.1064-1070, 1994.

六、圖表 (Figures and Tables)

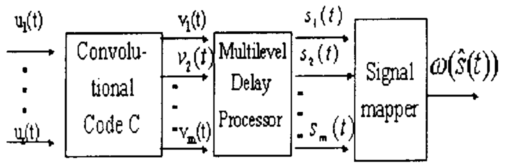

Figure 1: The encoder of T which uses a delay processor and a signal mapper.

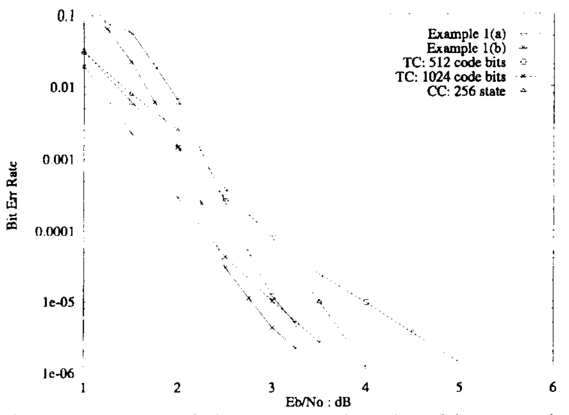

Figure 2: BER of the proposed codes, binary turbo codes (TC), and the 256-state conventional convolutional code (CC) in the AWGN channel.

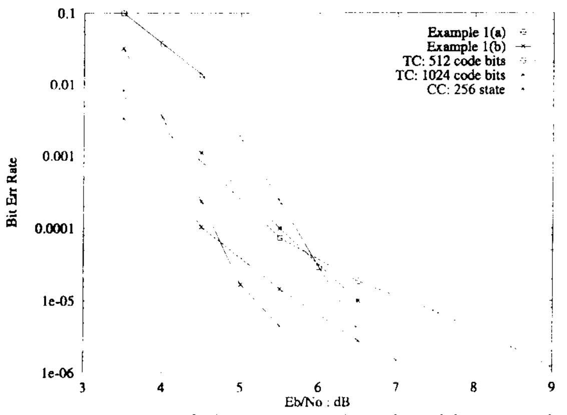

Figure 3: BER of the proposed codes, binary turbo codes (TC), and the 256-state conventional convolutional code (CC) in the Rayleigh fading channel.

Ex:	$(\tau_1,\ldots,\tau_m)$	λ,L	Codeword length
1a	$(9\lambda,5\lambda,2\lambda,0)$	8,16	512 code bits
1b	$(9\lambda,5\lambda,2\lambda,0)$	16,16	1024 code bits

Table 1: parameters of the proposed tail-biting trellis codes