TSG-RAN Working Group 1(Radio) meeting #5
Cheju, Korea, 1-4 June 1999

TSGR1#5(99)665

Agenda Item:

Source: **Fujitsu**

Title: **Tail bits and puncturing of rate 1/2 turbo coding**

Document for: **Discussion & Decision**

1 Introduction

In the current 3GPP specification [1][2], code rate 1/2 turbo code is obtained by alternately puncturing the two parity bit streams produced by the two constituent encoders, see Figure 1. The output sequence is: X(0), Y(0), X(1), Y'(1), X(2), Y(2), X(3), Y'(3), etc. However, the following issues have not been settled yet:

- Should the tail bits be punctured or not?
- If the tail bits should be punctured, what is best puncturing pattern?

In this contribution, the performance between tail bits punctured and tail bits unpunctured 1/2 turbo codes is compared, and a simple puncturing method is proposed.

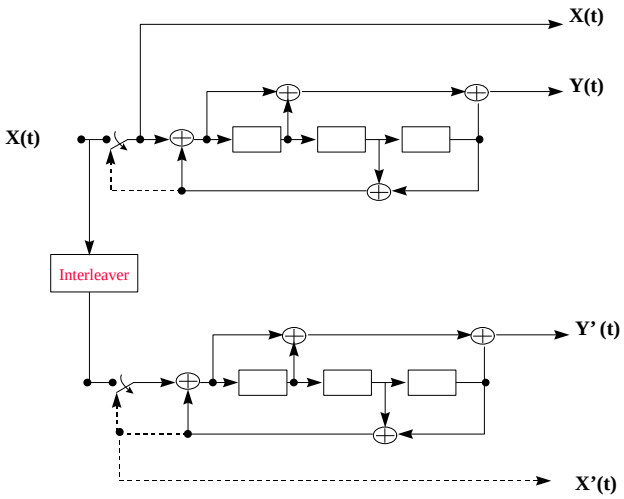

Figure 1. Structure of the 8-state PCCC encoder

2 Should we puncture tail bits or not?

The advantages of puncturing tail bits are as follows:

- Simplify the puncturing operation, because tail bits are punctured in the same way as parity bits. If tail bits are not allowed to be punctured, we have to know the position of tail bits.
- Better BER & FER performances.

The simulation results of tail bits punctured and tail bits unpunctured 1/2 turbo codes are shown in Figure 2. Simulation conditions are listed bellow.

- Coding block size: 320
- Constraint length: 4
- Log-MAP SISO decoding
- Turbo internal interleaver: MIL
- Trellis termination: conventional method

- Decoder iteration: 12
- Channel: AWGN
- At Eb/N0=3dB, at least 200 frame errors were counted.

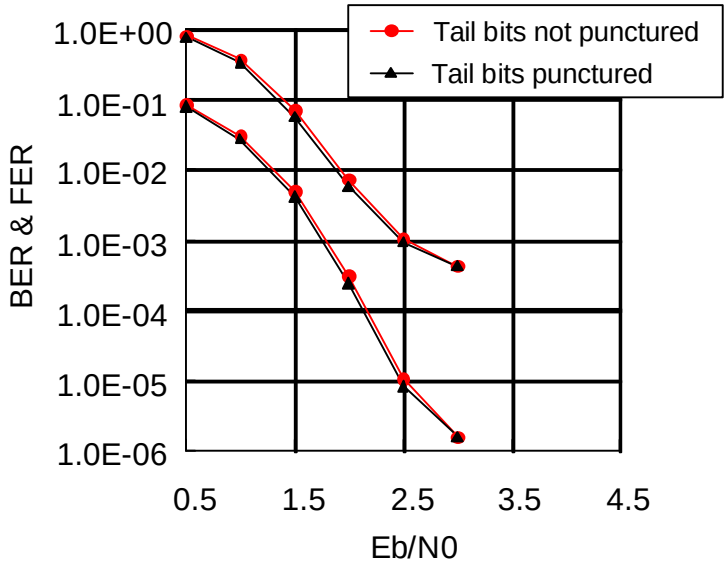

Figure 2. BER & FER performance of rate 1/2 turbo codes

From the Figure 2, we can see that both the BER and FER performance of tail bits punctured turbo code is slightly better than that of tail bits unpunctured turbo code.

We can conclude that the tail bits of turbo codes should be allowed to be punctured in the same way as parity bits in order to simplify the puncturing operation and to improve the performance.

3 Proposed puncturing method (symbol-based puncturing)

The following figure shows the propsed puncturing method[3]. Parity bits are punctured by pairs, tail bits are treated in the same way.

X(0)	X(1)	X(2)	X(3)	X(4)	X(5)	…	X(N-1)	X(N)	Y(N+1)	X'(N)	Y'(N+1)
Y(0)	Y(1)	Y(2)	Y(3)	Y(4)	Y(5)	…	Y(N-1)	Y(N)	X(N+2)	Y'(N)	X'(N+2)
Y'(0)	Y'(1)	Y'(2)	Y'(3)	Y'(4)	Y'(5)	…	Y'(N-1)	X(N+1)	Y(N+2)	X'(N+1)	Y'(N+2)

Tail bits

Figure 3. Propsed puncturing method

Therefore, the output sequence is

X(0), Y(0), Y’(0), X(1), X(2), Y(2), Y’(2), X(3), X(4), Y(4), Y’(4), …, etc.

Figure 4 shows the simulation results. The simulation conditions are the same as in Section 2.

From Figure 4, we can see that our proposed puncturing method shows slightly better performance than the conventional method.

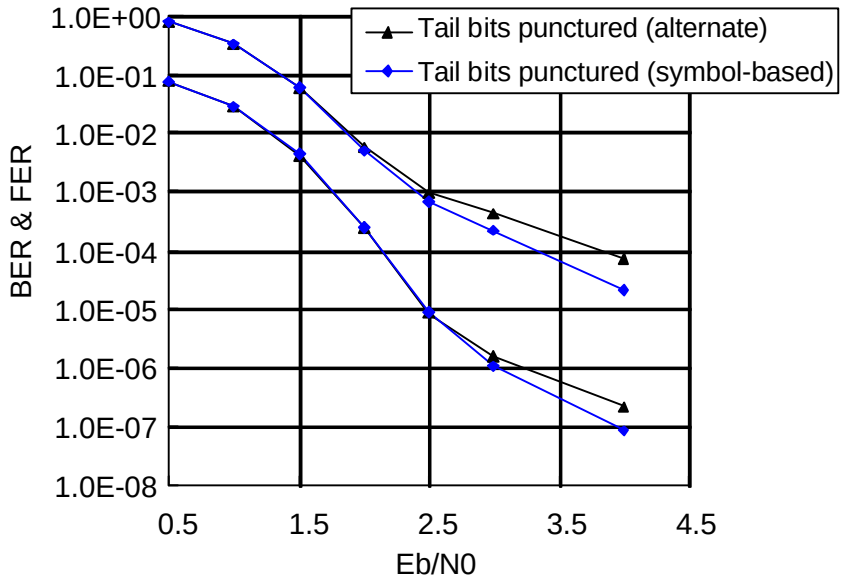

Figure 4. BER & FER performance of rate 1/2 turbo codes with the proposed puncturing scheme

4 Conclusion

The proposed puncturing method for both parity bits and tail bits is very easy to implement, and shows good BER and FER performance. We propose to adopt it in the text.

Text proposal for 25.212 and 25.222

4.2.2.2 Turbo coding (25.212)

6.2.2.2.1 Turbo coder (25.222)

The initial value of the shift registers of the PCCC encoder shall be all zeros.

The output of the PCCC encoder is punctured to produce coded bits corresponding to the desired code rate 1/3 or 1/2. For rate 1/3, none of the systematic or parity bits are punctured, and the output sequence is X(0), Y(0), Y'(0), X(1), ~~For rate 1/2, the parity bits produced by the constituent encoders are alternately punctured to produce the output sequence X(0), Y(0), X(1), Y'(1), X(2), Y(2), X(3), Y'(3), etc.~~ For rate 1/2, the parity bits and the tail bits produced by the constituent encoders are punctured by pairs to produce the output sequence X(0), Y(0), Y'(0), X(1), X(2), Y(2), Y'(2), X(3), X(4), Y(4), Y'(4), …, etc.

References

[1] 3GPP TSG RAN WG1 Multiplexing and channel coding (FDD) 25.212 v1.0.0 (1999-04)
[2] 3GPP TSG RAN WG1 Multiplexing and channel coding (TDD) 25.222 v1.0.0 (1999-04)
[3] Fujitsu, "Optimised puncturing scheme for turbo coding", TSGR#4(99)330 (1999-04).