TSG-RAN Working Group 1(Radio) meeting #4 **_TSGR1#4(99)388_**
Shin-Yokohama, Japan (18)19-20, April 1999

Agenda Item: Ad Hoc 4

Source: **Fujitsu**

Title: **Optimised puncturing scheme for Turbo coding**

Document for: Discussion

---

1 Abstract

In RAN WG1 #3 meeting, in order to achieve the best possible parallel concatenated convolutional code (PCCC) Turbo code performance, Ad Hoc 5 asks Ad Hoc 4 whether there are appropriate rate matching solutions for which the following puncturing guidelines can be incorporated:

- Minimise puncturing of systematic coded bits
- Provide approximately equal puncturing of parity bits from the two constituent encoders

This contribution provides solutions for both uplink and downlink. For downlink, a modified rate matching algorithm to resolve the problem is proposed in rather straight forward way, in which puncturing is applied only to non-systematic bit of Turbo code. For uplink, we recommend to use the current rate matching scheme using modified order input bit sequence into the rate matching block.

2 Rate matching for downlink

2.1 Algorithm

In downlink, rate matching is located after the channel coding. In the case, it would be rather easy to optimise the puncturing scheme in the rate matching. The current puncturing scheme [1] is a periodic puncturing of the input sequence. It is obvious that some specific puncturing patterns show inferior performance compared to more optimised puncturing patterns. The current puncturing algorithm does neither prevent the puncturing of the systematic coded bits nor does it balance the puncturing of the parity bits of the two constituent encoders. We could think of extreme cases, e.g. puncturing every 3*n bits, where only systematic coded bits or only bits of one encoder are punctured. This could result in performance losses of the Turbo code.

The puncturing conditions, which will improve the performance, are as follows,

(1) Puncture only the parity bits of the Turbo encoder and skip the systematic coded bits.
(2) Balance the puncturing of the parity bits of the two constituent encoder.
(3) Uniformly puncture the parity bits of each constituent encoder.

As for tail bits, due to the existence of a frame segmentation for Turbo coding before rate matching, the 12 bits (K=4) can be punctured to avoid the extra increase in complexity. Simulation results show no noticeable performance degradation.

To satisfy the condition (2) and (3), a periodic pairwise puncturing is proposed for the two parity bit sequences as illustrated in Figure 1. The algorithm can be easily obtained by modifying the current rate matching algorithm. Because the puncturing operations are performed by pairs, the calculation complexity in rate matching is about half of the original algorithm. The detailed algorithm is described in Section 4.

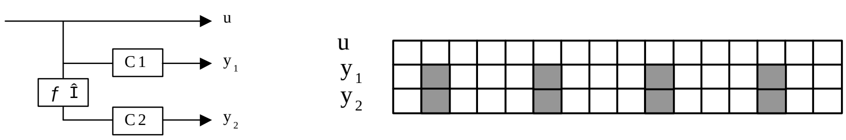


Figure 1. Proposed puncturing pattern (example)

[Rate matching algorithm]

$S_N = \{N_1, N_2, \dots, N_L\}$ = ordered set (in ascending order from left to right) of allowed number of bits per block

N_C = number of bits per matching block

$S_0 = \{d_1, d_2, \dots, d_{N_C}\}$ = set of N_C data bits

P = maximum amount of puncturing allowed

$$P = \begin{cases} 0.2\text{: for downlink} \\ 0.2\text{: for uplink} \end{cases}$$

The rate matching rule is as follows:

    find N_i and N_{i+1} so that $N_i \le N_C < N_{i+1}$

    $if\ (\frac{N_i}{N_C} > 1 - P)$

        $y = \left\lceil \frac{N_C - N_i}{2} \right\rceil$

        $e = N_C / 3$ *-- initial error between current and desired puncturing ratio*

        $m = 1$ *-- index of current bit*

        $pb = 0$ *-- number of punctured bits*

        do while $m <= N_C / 3$

            $e = e – 2 * y$ *-- update error*

            if $e <= 0$ then *-- check if bit number m should be punctured*

                $pb = pb + 2$

                if $mod\ (\ (\ N_C - N_i\ ) / 2\ )\ )\ \ 0$ *and* $pb = (\ N_C - N_i\ ) + 1$

                    puncture bit $3\ m$ from set S_0

                else

                    puncture bit $3\ m – 1$ and $3\ m$ from set S_0

                    $e = e + 2*N_C / 3$ *-- update error*

                end if

            end if

            $m = m + 1$ *-- next bit*

        end do

    else

        $y = N_{i+1} - N_C$

        $e = N_C$ *-- initial error between current and desired puncturing ratio*

-- this offset is flexible, e.g. e = 2Nc

        $m = 1$ *-- index of current bit*

        do while $m <= N_C$

            $e = e – 2 * y$ *-- update error*

            do while $e <= 0$ *-- check if bit number m should be repeated*

                repeat bit m from set S_0

                $e = e + 2*N_C$ *-- update error*

            enddo

            $m = m + 1$ *-- next bit*

        end do

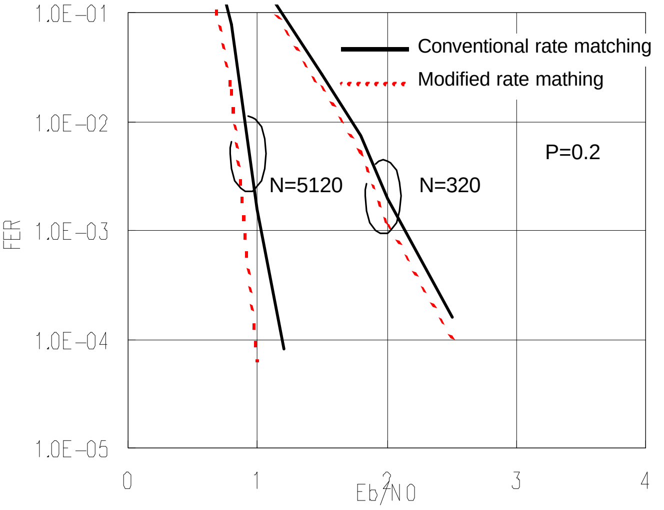
    end if

Legend :

    $y = \lceil x \rceil$: The nearest integer that is larger than or eqyal to *x*.

2.2 Performance evaluation in AWGN channel

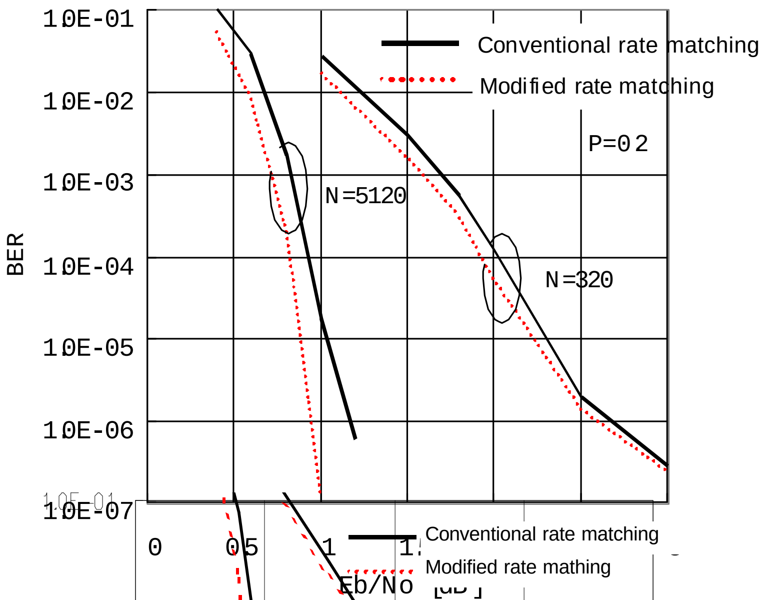
BER and FER performance using the modified rate matching scheme and the conventional rate matching are



shown in

Figure **2** and Figure 3 respectively. Coding block size of 320bit and 5120bit are used as for the extreme cases. Simulation conditions are listed bellow.

- Puncturing ratio: 0.2 (20% puncturing)
- Coding block size: 320 and 5120
- Constraint length: 4
- Log-MAP SISO decoding
- Turbo internal interleaver: MIL
- Trellis termination: conventional method
- Decoder iteration: 8
- Channel: AWGN



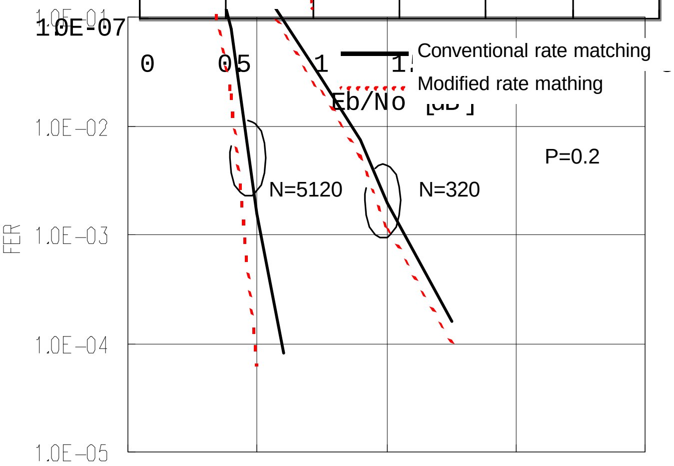


Figure 2. BER performance

Figure 3. FER performance

We have evaluated the effect when the proposed rate matching is used for convulutional coding. Figure 4 shows BER performance of convolutional coding using two kind of rate matching schemes. At the BER of 10^{-3}, no performance degradation can be found.

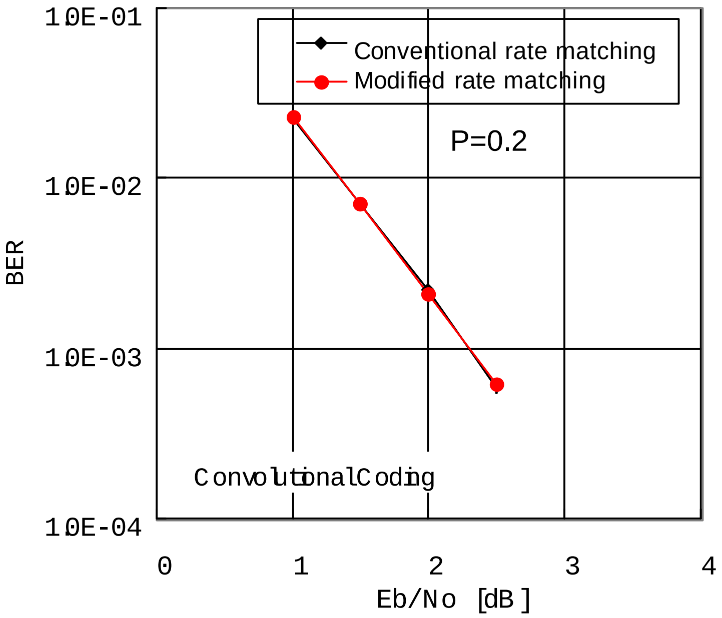


Figure 4. BER performance of convolutional coding

3 Uplink

Similar optimisation as for the downlink could be done in uplink to satisfy the conditions described above. However, such an optimasation of the rate matching algorithm may introduce considerable complexity increase. Therefore, it is proposed to use the current rate matching algorithm using an input sequence with a modified bit order into the rate matching block, which is illustrated in Figure 5. By using this order of input sequence for rate matching, the condition (2) and (3) can be satisfied. Though the condition (1) can not be satisfied and there is a possibility that all 3bits at the same time instance could be punctured, it may not introduce much performance degradation.

u	y_1	y_2

Figure 5. Input sequence for uplink rate matching block

4 Conclusion

The proposed rate matching algorithm is superior to the original algorithm in terms of error rate performance and calculation complexity.

5 References

[1] 3GPP TSG RAN WG1 Multiplexing and channel coding (FDD) S1.12 v1.1.0 (1999-03)