

Agenda Item: 7.3

Source: Broadcom

Title: On Rate Matching Parameters for TS36.212

Document for: Decision

## 1 Summary

In 5.1.4.1.2 and 5.1.4.1.3 of TS36.212 [1], a pair of parameters for circular buffer rate matching, namely, the skip index  $\sigma$  for starting point of systematic bits and offset index,  $\delta$ , for the sub-block interleave of the second parity bits are given to be (2, 1). With these parameters and 6 iterations of turbo decoding, there are many poorly performing block sizes, some of them can cause as large as a 0.4 dB degradation in performance (which are generally referred to as spikes), see Fig. 1 and [2].

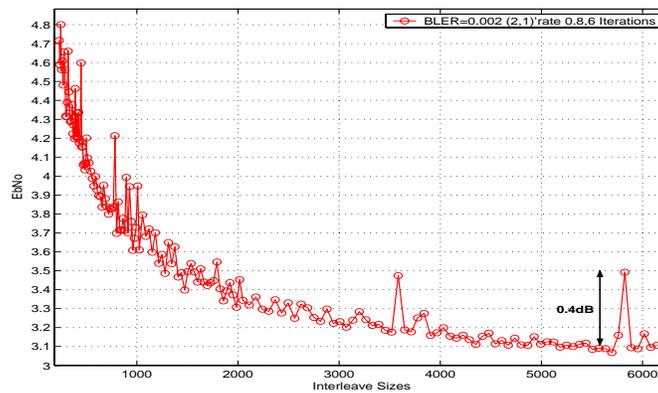


Figure 1 Rate =0.8, TS36.212

In [2] replacing these parameters by  $(\sigma, \delta) = (4, 4)$  was suggested. Indeed this reduces the spikes tremendously under various conditions. However, as also indicated in [2], with  $(\sigma, \delta) = (4, 4)$  the overall performance with 6 iterations deteriorates.

With puncturing pattern analysis and thorough simulations, we found that a better choice should be  $(\sigma, \delta) = (3, 3)$ . For example, with these parameters and 6 iterations of turbo decoding all the spikes in Fig. 1 disappear (see Fig.2). Moreover the overall performance improves as shown in Fig. 2.

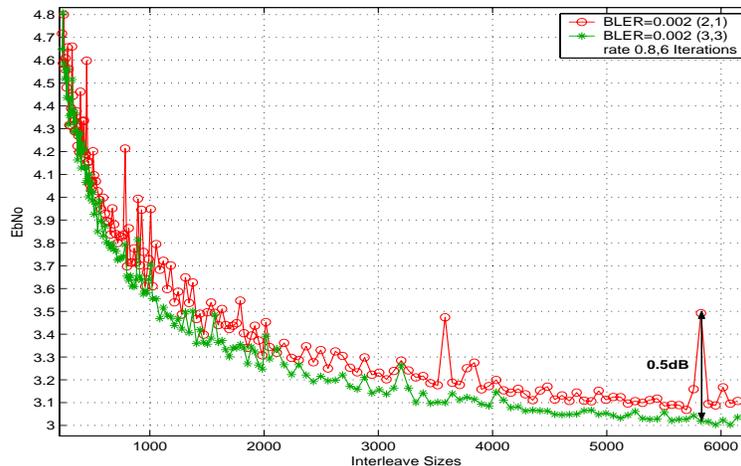


Figure 2 Rate 0.8. Compare TS36.212 to the proposal

Simulation results for different BLER (0.1, 0.01, 0.002) as well as different code rates (0.4, 0.5, 0.6, 0.7, 0.8) given in this document show the reduction of spikes and improvement in overall performance. Therefore, we propose:

**Modification 1)** On page 14 of TS36.212, replace  $\pi(i) = \left( P \left( \left\lfloor \frac{i}{R} \right\rfloor \right) + C \times (i \% R) + 1 \right) \% K$  with

$$\pi(i) = \left( P \left( \left\lfloor \frac{i}{R} \right\rfloor \right) + C \times (i \% R) + 3 \right) \% K .$$

**Modification 2)** On page 15 of TS36.212, replace  $R \times (24 \times r_{vidx} + 2)$  with  $R \times (24 \times r_{vidx} + 3)$ , where R is the number of rows for the sub-block interleave.

## 2 Analysis and simulations on skip and offset indices for CBRM

It was pointed out in [3] that CBRM offers periodic puncturing patterns. Furthermore, [3] suggests using sub-optimal periodic puncturing patterns that are related to the mother encoder. In [2], based on avoiding catastrophic puncturing patterns, a different offset index and skip index were suggested. Unfortunately, the modified parameters only work for turbo decoding with 8 iterations.

Consider puncturing patterns for rate 0.8 codes.

1.1) When the skip index is 2 (defined in TS 36.212), the puncturing pattern of systematic bits is

1111011111111111

which has a period of 16. Only one bit is punctured in each period.

1.2) When the skip index is 4 [2], the corresponding puncturing pattern is

11110111

which has a period of 8. Only one bit is punctured in each period. Although the period is enough as noticed in [3], the puncture pattern given by CBRM is not optimally selected. Therefore, this period is too short for 6 iterations decoding.

1.3) When skip index is 3 as we propose here, the corresponding puncturing pattern is

11110111111101111111011111111111

which has a period of 32. Different to the previous patterns 3 bits are punctured in each period, i.e. there is a systematic bit being punctured in approximately every 10 bits. The amount of puncturing lies between those of puncturing patterns 1) and 2). This turns out to remove almost all of the previously existing spikes.

Now consider the puncturing pattern for parity-1 bits of the turbo encoder.

2.1) With the parameter given in TS 36.212, the puncturing pattern is

00001000100010000000100000001000

which has a period of 32. 5 bits are not punctured in each period; i.e. there is one un-punctured bit in approximately every 6 bits.

2.2) If the parameter is taken to be (4, 4) as in [2] or to be (3, 3) as we propose, the puncturing pattern is

0000100010001000

which has a period of 16. 3 bits are not punctured in each period; i.e. there is one un-punctured bit in approximately every 5 bits. Comparing to pattern 2.1), fewer parity-1 bits are punctured.

Consider the puncturing patterns for parity-2 bits.

3.1) With the parameter given in TS 36.212, the puncturing pattern is

00000100010001000000010000000100

which has a period of 32. 5 bits are not punctured in each period; i.e. there is one un-punctured bit in approximately every 6 bits.

3.2) If the parameter is taken to be (4, 4) in [2], the puncturing pattern is

1000000010001000 (\*)

which has a period of 16. 3 bits are not punctured in each period, i.e., there is one un-punctured bit in approximately every 5 bits.

3.3) If the parameter is taken to be (3, 3) as we propose, the puncturing pattern is

0000000100010001 (\*\*)

The difference between (\*) and (\*\*) is the starting position of unpunctured bits and therefore the total number of punctured bits in the output stream will be different.

In the rest of this section, the BLER performance comparison for code rate 0.8, 0.7, 0.6, 0.5 and 0.4 will be given.

#### Simulation parameters

Common code structure	Turbo code in TS 36.212
Circular buffer rate matching parameters	1) $(\sigma, \delta)=(2, 1)$ (TS 36.212) 2) $(\sigma, \delta)=(3, 3)$
First transmission code rate	$r = 0.4, 0.5, 0.6, 0.7, 0.8$
Tested block lengths	For all QPP interleave size $K \geq 288 \times r$
Redundancy version (RV)	0
Number of maximal iterations	6
Modulation	QPSK
Channel	AWGN

## 2.1 Rate 0.8

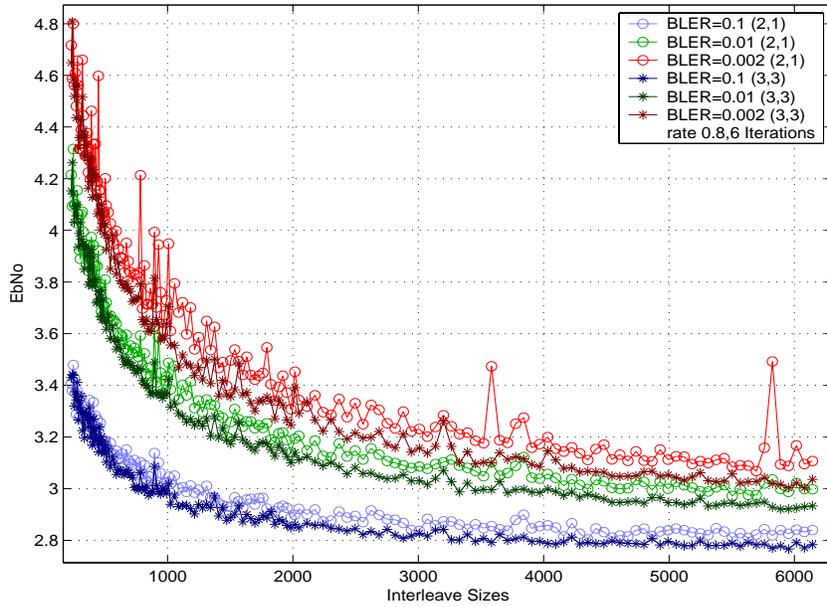


Figure 4 Rate 0.8 (3, 3) vs. (2, 1)

All the spikes caused by using parameters (2, 1) disappear. Moreover, overall performance improves. Specifically, among the 164 interleave sizes, when BLER = 0.1, there are 150 interleave sizes where parameters (3,3) outperform (2,1), i.e. 91.5% are better; when BLER = 0.01, there are 155 interleave sizes where parameters (3,3) outperform (2,1), i.e. 94.5% are better; when BLER = 0.002, there are 154 interleave sizes where parameters (3,3) outperform (2,1), i.e. 93.9% are better.

## 2.2 Rate 0.7

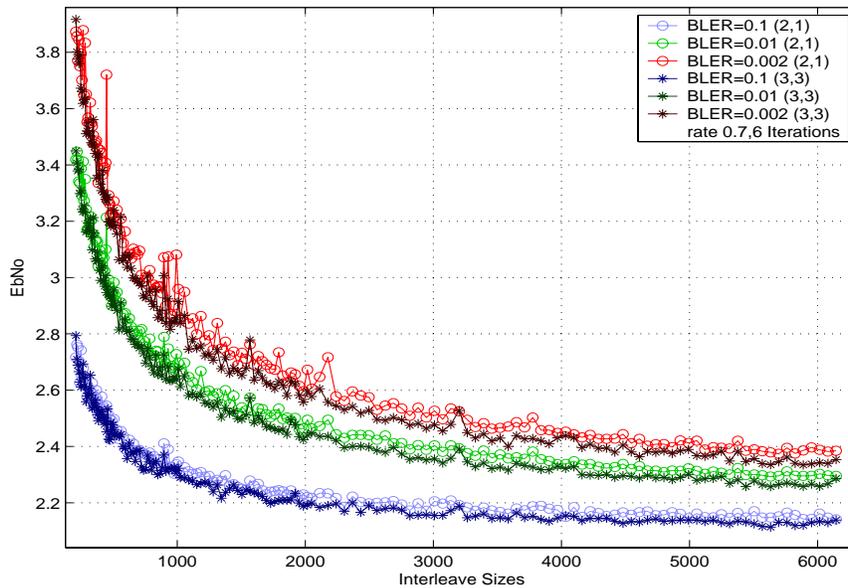


Figure 5 Rate= 0.7 (3, 3) vs. (2, 1)

Once again, all the spikes caused by using parameters (2,1) disappear and overall performance improves. Specifically, among 167 interleave sizes, when BLER = 0.1, there are 149 interleave sizes where parameters (3,3) outperform (2,1), i.e. 89.2% are better; when BLER = 0.01, there are 157 interleave sizes where parameters (3,3) outperform (2,1), i.e. 94% are better; when BLER = 0.002, there are 156 interleave sizes where parameter (3,3) outperform (2,1), i.e. 93.4% are better.

### 2.3 Rate 0.6

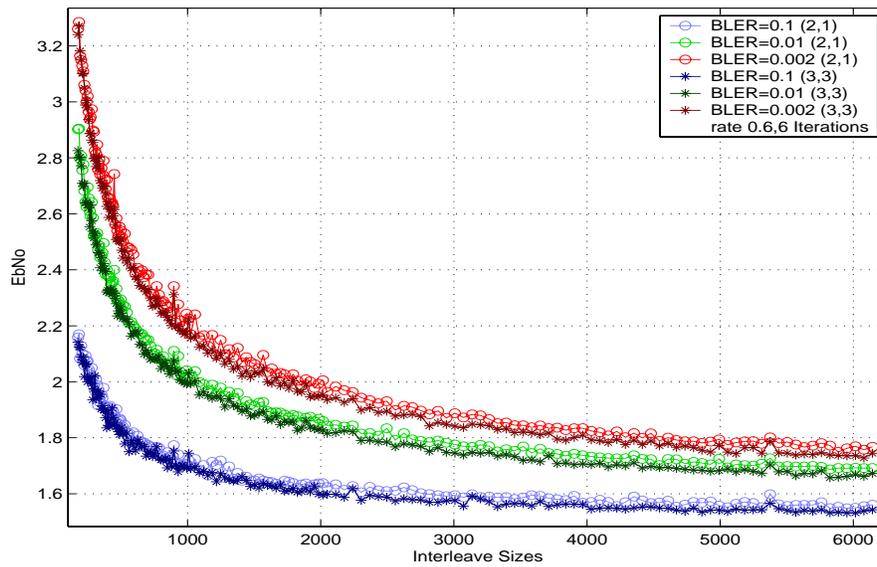


Figure 6 Rate =0.6 (3, 3) vs. (2, 1)

Clearly, overall performance improves with the new parameters. Specifically, among 171 interleave sizes, when BLER = 0.1, there are 156 interleave sizes where parameters (3,3) outperform (2,1), i.e. 91.2% are better; when BLER = 0.01, there are 161 interleave sizes where (3,3) outperform (2,1), i.e. 94.2% are better; when BLER = 0.002, there are 165 interleave sizes where (3,3) outperform (2,1), i.e. 96.4% are better.

### 2.4 Rate 0.5

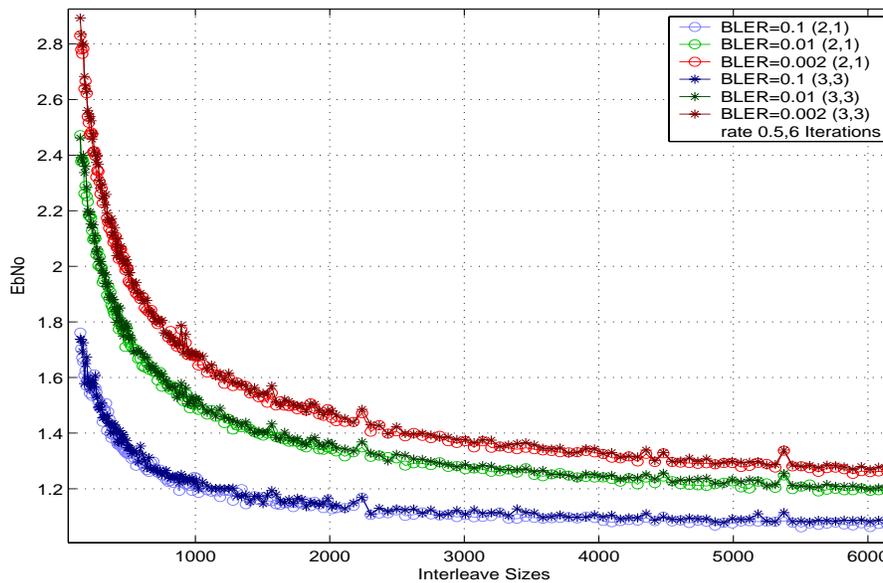


Figure 7 Rate 0.5 (3, 3) vs. (2, 1)

With a code rate of 0.5 and using parameters (3, 3) gives almost the same performance as using parameters (2, 1).

## 2.5 Rate 0.4

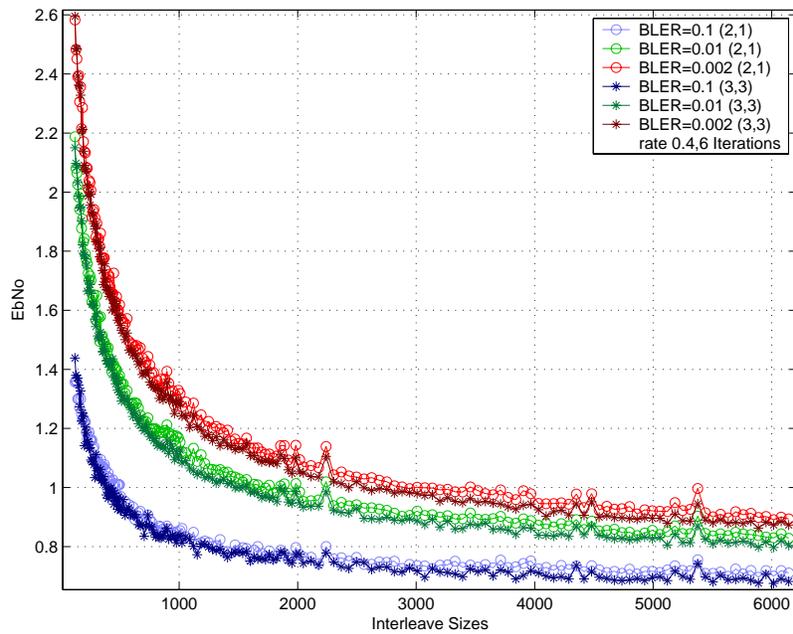


Figure 8 Rate 0.4 (3, 3) vs. (2, 1)

Overall performance improves with the new parameters. Specifically, among 178 interleave sizes, when BLER = 0.1, there are 156 interleave sizes where parameters (3,3) outperform (2,1), i.e. 87.6% are better; when BLER = 0.01, there are 167 interleave sizes where parameters (3,3) outperform (2,1), i.e. 93.8% are better; when BLER = 0.002, there are 172 interleave sizes where parameters (3,3) outperform (2,1), i.e. 96.6% are better.

## References

- [1] Technical Specification Group Radio Access Network: Multiplexing and channel coding (release 8), 3GPP TS 36.212 V1.3.0 (2007-07)
- [2] Ericsson, "Decoder performance of CBRM algorithms," 3GPP TSG RAN WG1 #49bis R1- 073163.
- [3] Broadcom, "Comparing rate matching puncturing to optimal periodic puncturing," 3GPP TSG RAN WG1 #49 R1- 072406.