3GPP TSG-RAN WG1#49*bis* **R1-073028**
Orlando, USA, June 25—29, 2007

Source: **Ericsson**
Title: **Catastrophic Puncturing Avoidance**
Agenda Item: **5.5 Channel Coding**
Document for: **Discussion and Decision**

1. Introduction

A circular buffer structure was agreed in RAN1#49 Kobe for LTE turbo coding rate matching [1—5]. In view of the performance results reported in [5—8], it was also agreed further improvement in the circular buffer rate matching (CBRM) algorithms might be beneficial. Toward this end, we present an analytical framework based on avoiding catastrophic puncturing patterns [9] and propose a simple modification to the algorithm that yields marked performance improvements (see Section 5 and [10]) and simplifies HARQ operations [11].

The overall CBRM structure employs three 32-column rectangular interleavers to process the three streams multiplexed from the four outputs of a turbo encoder [12]. The three streams are conveniently referred to as the Systematic, Parity1 and Parity2 streams (though each contains tail bits of different origins). The interleaved streams are conceptually organized into a buffer illustrated in Figure 1. That is, the first 32 columns of the buffer consist of the interleaved Systematic stream while the next 64 columns are multiplexed from the interleaved Parity1 and Parity2 streams. A number of redundancy versions (RVs) can then be defined as consisting of coded bits starting at certain columns in this buffer [4]. The pseudo codes for CBRM generation shown in Figure 2 will be referred to make the discussion in this paper more precise. The broad findings in this paper are summarized in the following:

- CBRM(σ=2,δ=0)
  In the basic CBRM algorithm, Systematic bit of the first σ=2 columns are skipped (and hence punctured from the transmitted codewords). The interleavers for the two parity streams are identical without applying any address offsets (i.e., δ=0). As shown in Section 2 and [5—8], this basic algorithm suffers from catastrophic puncturing patterns in both constituent codes for high coding rates.

- CBRM(σ=2,δ=1)
  An improvement to the basic algorithm is to offset interleaving addresses for the Parity2 stream relative to those for the Parity1 stream by δ=1. As shown in Section 3, this algorithm avoids catastrophic puncturing patterns for the 2nd constituent code while leaving those in the 1st constituent code unfixed. While general performance at high coding rates is significantly improved, instability remains and leads to occasional performance flooring [5—8].

- CBRM(σ=4,δ=4)
  This new CBRM design (of skipping the first σ=4 Systematic columns and offsetting the Parity2 interleaving addresses by δ=4) is proposed based on the analysis and the constraint of applying only one address offset to saving computational complexity. As shown in Section 4, catastrophic puncturing is avoided in both constituent codes at high coding rates. Furthermore, this set of parameters move all critical parity bit positions to the front (of the parity bit region) of the circular buffer. As a result, this algorithm picks up more important parity bits more quickly than the other two proposals and thus is potentially capable of producing codewords of higher Hamming weights. An added benefit of this feature is that the system could be operated more simply because it would be unnecessary to switch between RVs with or without systematic bit puncturing for higher or lower coding rates. These performance improvements and operational benefits are verified in Section 5 and [10, 11].

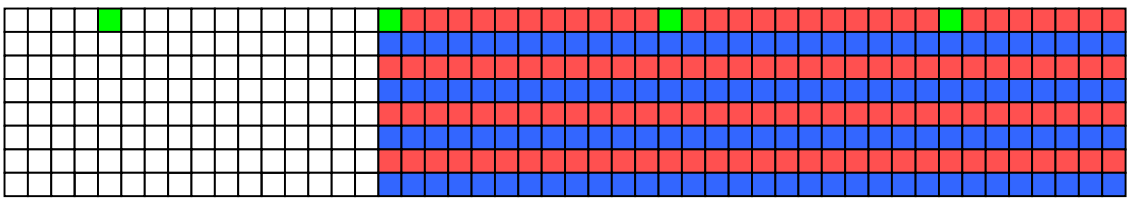

Figure 1 Conceptual composition of circular buffer for LTE rate matching. While, red and blue cells contains bits from the Systematic, Parity1 and Parity2 streams. Green cells mark starting points of redundancy versions.

```
Input  RV                                % redundancy version
Input  Ndata                             % number of desired coded bits
Input  Sys, Par1, Par2                   % buffers for sys, Parity1, Parity2
                                         %   length = K+4
Output Out                               % buffer for retained bits
```

```
1  n = 0;                                % index of the output buffer
2  c = σ+RV*(96/nRV);                    % get the starting column
3  while n<Ndata
4    c = mod(c, 96);                     % go through the columns circularly
5    if c<32                             % the first 32 cols are Systematic
6      A = ColPerm(c);                   % get starting address for Sys
7      while A<K+4                       % before the end of column
8        Out(n++) = Sys(A);              % take the systematic bit at A
9        if n==Ndata, return, end;
10       A += 32;                        % next address
11     end
12     c += 1;                           % next column
13   else                                % the next 2*32 cols are parities
14     A = ColPerm(c/2-16);              % get starting address for Parity1
15     while A<K+4                       % before the end of column
16       Out(n++) = Par1(A);             % take the Parity1 bit at A
17       if n==Ndata, return, end;
18       Out(n++) = Par2(mod(A+δ,K+4)); % take the Parity2 bit at A+δ
19       if n==Ndata, return, end;
20       A += 32;                        % next address
21     end
22     c += 2;                           % next two columns
23   end
24 end
```

Figure 2 Pseudo codes for circular buffer rate matching (assuming no pre-padding before turbo encoding). Three different CBRM algorithms discussed in this paper are based on different settings for the (σ, δ) pair. The number of redundancy versions is denoted by nRV.[1]

2. Catastrophic Puncturing in CBRM(σ=2,δ=0)

The inter-column permutation adopted in the CBRM structure is given by

$$\text{ColPerm} = [0, 16, 8, 24, 4, 20, 12, 28, 2, 18, 10, 26, 6, 22, 14, 30, 1, 17, 9, 25, 5, 21, 13, 29, 3, 19, 11, 27, 7, 23, 15, 31]. \quad (1)$$

Hence, operations of the CBRM algorithms have a period of 32 (ignoring the tails).

By skipping the first σ=2 columns (0th and 16th), a periodic systematic bit puncturing is introduced in the 1st constituent code. This can be represented by the following periodic puncturing mask:

$$\text{Systematic Puncturing Mask} = [\mathtt{01111111111111110111111111111111}], \quad (2)$$

where 1 and 0 indicate retaining and discarding of systematic bits at the corresponding positions, respectively. Note that, if the size of the QPP interleaver is divisible by 16, then we have, for any integer k,

$$\Pi_{QPP}(16k) \bmod 16 = 0 \quad (3)$$

[1] By calling this algorithm with RV=0 and Ndata=3K+12, a complete physical circular buffer can be constructed. The algorithm computes the exact set of addresses for Ndata memory accesses and requires no multiplications for address computation. Once the (σ,δ) parameters are finalized, line 2 can be implemented with a look-up table of nRV entries. Note also that ine 18 requires only a range check instead of a full-blown modulo operation in implementation

by the maximally contention free property of the QPP interleavers. That is, for all LTE QPP interleavers of size $K \geq 512$ and half of those with $K<512$ [12], skipping the first two columns in the circular buffer induces the same periodic-16 systematic bit puncturing in both constituent codes.

Systematic bit puncturing has been shown to be effective improving the performance of turbo codes at higher coding rates [9, 13, 14, 15]. However, to reap the full benefits of this technique, it is important to avoid certain puncturing patterns in the parity bits streams. In particular, the overall encoding process (including turbo encoder and rate matcher) might become catastrophic if all (or almost all) parity weights corresponding to an input bit sequences of potentially infinite weight are punctured away [16]. Catastrophic puncturing patterns for the 8-state constituent codes have been studied in [9]. For the Systematic puncturing pattern mask in (2), the corresponding Parity bit puncturing patterns to avoid are classified in the following critical bit mask:

Critical Parity Bit Mask = [**xCCxxCxCCCxxCxCC**xCCxxCxCCCxxCxCC], (4)

where C and x indicate critical and uncritical parity bit positions, respectively. That is, if all parity bits at the critical positions are punctured, the overall encoding processing becomes catastrophic by mapping an input block of very large (or infinite) weight into a codeword of little or no weight. With low Hamming weights, small amount of channel noises can induce the decoder to make completely different (and wrong) decisions, which results in very high block error rates (BLER). To avoid such catastrophic puncturing, at least one of the critical parity bit in each period of 16 should be retained by the rate matcher.

To examine the parity bit puncturing pattern of the, it is necessary to interleave the criticality mask of (4) by the inter-column permutation pattern of (1), which gives

Critical Bit Mask for Parity1 and Parity2 CB = [**x**xCC**x**xCCCC**x**x**x**xCCCCCCCC**x**x**x**x**x**CCCC]. (5)

Note that the mask applies to both constituent codes because the interleavers for the two parity streams are identical (δ=0). This allows us to make the following analysis concerning this CBRM(σ=2,δ=0) algorithm:

- For coding rates $r \geq 32/34$, transmitted bits for RV=0 consist of 30 columns of the Systematic stream and no more than 2 columns for each of the Parity streams. It is then clearly from (5) that the overall encoding process is catastrophic because none of the critical parity bits are retained.
- For coding rates $r \leq 32/38$, transmitted bits for RV=0 consist of 30 columns of the Systematic stream and more than 4 columns for each of the Parity streams. Catastrophic puncturing in both constituent codes is avoided since at least one of the critical parity bit positions is retained periodically.
- For coding rates $32/34 \leq r \leq 32/38$, transmitted bits for RV=0 consist of 30 columns of the Systematic stream and no more than 4 columns for each of the Parity streams. With less than half of the parity bits retained from the critical positions in both constituent codes, catastrophic puncturing cannot be completely avoided. However, as more and more critical parity bits are retained in the two constituent codes, a kind of statistical avoidance can take place through iterative decoding. That is, as the code rate lowers toward 32/38, small amount of punctured critical positions in one constituent code could be saved by interleaved/deinterleaved extrinsic information from unpunctured positions of the other constituent code. Since such process depends on the specific QPP interleavers, catastrophic puncturing could be avoided for some block lengths but not for others. This instability can cause BLER performance flooring for certain block sizes.

From this analysis, it can be expected the performance of the CBRM(σ=2,δ=0) algorithm to be inferior for code rate $r \geq 0.94$. The performance should gradually improve as the coding rate decreases toward 0.84. For code rate $r \leq 0.84$, its performance should become competitive to other RM algorithms. These predictions are verified in Section 5. The findings are consistent with the performance results reported in [5—8].

3. Partial Catastrophic Puncturing Avoidance in CBRM(σ=2,δ=1)

Two equivalent partial solutions to the catastrophic puncturing problems at high coding rate are proposed in [15] and [17], both of which are based on simple modifications to the interleaver for the Parity2 stream. As indicated in (1), adding an offset of δ=1 to the interleaving address is equivalent to cyclically shifting the interleaver to its middle. With this slight modification, the critical bit mask for the Parity2 stream becomes

Critical Bit Mask for Parity2 CB = [CCCCCC**x**x**x**x**x**CCCCCC**x**x**x**CC**x**xCCCCx**x**]. (6)

The critical parity bit mask for the Parity1 stream remains the same as (5).

We can make the following analysis:

- For coding rates $r \leq 32/38$, transmitted bits for RV=0 consist of 30 columns of the Systematic stream and more than 4 columns for each of the Parity streams. Catastrophic puncturing is avoided in both constituent codes.
- For coding rates $32/38 \leq r \leq 32/34$, transmitted bits for RV=0 consist of 30 columns of the Systematic stream and no more than 4 columns for each of the Parity streams. That is, catastrophic puncturing is avoided in the 2nd constituent code but such avoidance is not guaranteed in the 1st constituent code. However, as more and more critical parity bits are retained, statistical avoidance through iterative decoding can take place in a process similar to that described in Section 2 for CBRM(σ=2,δ=0). It is similarly noted that this statistical avoidance of catastrophic puncturing could result in BLER performance instabilities manifesting as SNR spikes for certain block sizes.
- For coding rates $r \geq 32/34$, transmitted bits for RV=0 consist of 30 columns of the Systematic stream and no more than 2 columns for each of the Parity streams. In these cases, while certain critical parity bits are retained in the 2nd constituent code, the puncturing in the 1st constituent code is catastrophic. This could break the effectiveness of the turbo decoding algorithm since the 1st constituent code is essentially irrelevant for certain error events.

From this analysis, it can be expected the performance of the CBRM(σ=2,δ=1) algorithm to be somewhat inferior for code rate $r \geq 0.94$. The performance would then improve as the coding rate decreases though BLER flooring could be expected for certain block sizes because statistical catastrophic puncturing avoidance is not always sufficient. For code rate $r \leq 0.84$, its performance should be competitive to other RM algorithms. These predictions are verified in Section 5 and [10]. The findings are consistent with the performance results reported in [5—8].

4. Improved Rate Matching with CBRM(σ=4,δ=4)

We investigate alternative CBRM algorithm improvements with two performance objectives and one complexity constraint. First, the improved algorithm should avoid catastrophic puncturing to the most extent possible. Secondly, performance improvement should be appreciable not only for high coding rates but also over ranges of more frequent usage. Thirdly, it is constrained to maintain the same computational complexity as the current working assumption (i.e., CBRM(σ=2,δ=1)) by using only one address offset (instead of offsetting both constituent codes).

Under these considerations, it is clearly that the periodic-16 systematic bit puncturing employed in the previous two algorithms needs be replaced. Because the corresponding critical parity bit mask starts with uncritical bits, catastrophic puncturing in the 1st constituent code cannot be avoided for high coding rates. The search of an alternative period for systematic bit puncturing is further constrained by the properties of LTE QPP interleavers. More specifically, if the period is not a divisor of the interleaver length K, then periodic puncturing of the 1st constituent code becomes aperiodic for the 2nd constituent code after QPP interleaving (which makes optimization and performance guarantee difficult, if not impossible). This leaves only one choice: periodic-8 systematic bit puncturing.

It is proposed for this new algorithm to skip the first four columns by setting σ=4. This induces the following periodic puncturing mask for the Systematic stream:

Systematic Puncturing Mask = [**01111111**01111111**01111111**01111111]. (7)

Since all QPP interleaver lengths in LTE are divisible by 8 [12], we have

$$\Pi_{QPP}(8k) \bmod 8 = 0 \tag{8}$$

for any integer k. That is, for all LTE turbo codes, this setting introduces identical periodic-8 systematic bit puncturing to both constituent codes.

The corresponding critical parity bit mask is given by the following [9]

Critical Parity Bit Mask = [**CxCxCCCx**CxCxCCCxCxCxCCCxCxCxCCCx]. (9)

For the Parity1 stream, the critical bit mask after interleaving is given

Critical Bit Mask for Parity1 CB = [**CCCCCCCCCCCCCCCCx**xxx**CCCCx**xxx**x**xxx]. (10)

By offsetting δ=4 to the interleaver, the critical bit mask for the Parity2 stream becomes

Critical Bit Mask for Parity2 CB = [**CCCCCCCCCCCCCCCCCCCC**xxx**xx**xxxxxx**x**]. (11)

We can make the following analysis:

- For coding rates $r \leq 32/36$, transmitted bits for RV=0 consist of 28 columns of the Systematic stream and more than 4 columns for each of the Parity streams. Catastrophic puncturing is avoided in both constituent codes.
- For coding rates $r \geq 32/36$, transmitted bits for RV=0 consist of 28 columns of the Systematic stream and no more than 4 columns for each of the Parity streams. Catastrophic puncturing avoidance cannot be guaranteed. However, as with previous two algorithms, statistical avoidance through iterative decoding can take place unless the coding rate is extremely high. Some occasional BLER performance instability could be expected in this rate region.
- The new CBRM(σ=4,δ=4) algorithm exhibits a very attractive feature not observed in either of the alternative algorithms. It can be seen in (10) and (11) that the new CBRM algorithm moves all the critical parity bit positions to the front (of the parity bit region) of the circular buffer. More specifically, no uncritical parity bits are transmitted in RV=0 for all coding rates higher than 0.53. This feature is not only beneficial in avoiding catastrophic puncturing at high coding rates. It also allows the algorithm to retain more critical parity bits than the alternatives and thus could produce codewords with higher Hamming weights. This could lead to improved performance over a wide range of coding rates that are normally not affected by catastrophic puncturing.

  This feature could also simplify HARQ system operations by removing the need to switch between redundancy versions with or without systematic bit puncturing for high or low coding rate cases, respectively. The switching operations are motivated by the concerns that systematic bit puncturing might be detrimental to performance of initial transmissions at lower coding rates. However, since all critical parity bits are at the front (of the parity bit region) of the circular buffer, RV=0 might still be optimal for low rate cases even though some systematic bits are punctured. Performance results supporting this conjecture are discussed in [11].

From this analysis, it can be expected the performance of the CBRM(σ=4,δ=4) algorithm to be very competitive for all rates $r \leq 0.89$. Performance for rates higher than 0.89 is expected to degrade gradually as some critical parity bits will be left out and statistical avoidance of catastrophic puncturing starts to play a role in the BLER performance. These performance predictions are verified in Section 5 and [10].

5. Preliminary Performance Analysis

We illustrate the performance of the three CBRM algorithms discussed in the previous sections using a fixed code block size K=2048 over a wide range of higher coding rates. The methodology is summarized in Table 1 and is similar to that set in [18]. More comprehensive performance evaluation is presented in [10, 11].

The required E_b/N_0 for achieving 1% BLER is plotted in Figure 3. We make the following observations:

- For this particular block size, performance of the CBRM(σ=2,δ=0) algorithm does not deteriorate from that of the alternative algorithms until the code rate increases over around 0.86. Above this rate, performance of this algorithm suffers significantly from catastrophic puncturing. Statistical avoidance of catastrophic puncturing has been effective in saving performance in the region of $0.84 \leq r \leq 0.86$.
- For this particular block size, statistical avoidance of catastrophic puncturing has been very effective in keeping the performance of the CBRM(σ=2,δ=1) algorithm competitive in the code rate region of $0.84 \leq r \leq 0.93$. (Note that the SNR spikes for many block sizes shown in [5—8] are cases when such statistical avoidance is insufficient.) For code rates higher than 0.94, the negative effects of leaving the 1st constituent code catastrophic start to affect the BLER performance.
- Performance of the proposed CBRM(σ=4,δ=4) algorithm remains competitive across the entire range of considered code rates. Statistical avoidance of catastrophic puncturing plays a role for code rates higher than 0.89.

6. Conclusion

Catastrophic puncturing pattern avoidance has been used in this paper to analyze performance of alternative CBRM algorithms. The methodology explains when and how the CBRM(σ=2,δ=0) algorithm breaks down as well as why the CBRM(σ=2,δ=1) algorithm performs better in comparison. With this tool, we further propose an improved CBRM(σ=4,δ=4) algorithm. While maintain identical computational complexity, the new proposal has been verified to achieve better performance over a wide range of coding rates (in Section 5 and [10]) as well as to simplify HARQ operations [11]. It is proposed to adopt this improved CBRM design for LTE.

Table 1 Simulation Parameters

Common Code Structure	LTE Turbo Coding [12]
Rate Matching Algorithms	1. CBRM(σ=2,δ=0) as described in Section 2 and [3, 5] 2. CBRM(σ=2,δ=1) as described in Section 3 and [3, 5] 3. CBRM(σ=4,δ=4) as described in Section 4
Test Block Lengths	K = 2048
Codeword Lengths	N = 2144, 2176, 2208, …, 4096 (in increments of 32)
Coding Rates	$r = K/N$ = 0.5—0.96
Redundancy Version	RV=0
Decoding Algorithm	Improved Max-Log-MAP (i.e., 0.75 scaling on extrinsic information)
Iterations	8
Modulation	QPSK
Channel	Static AWGN

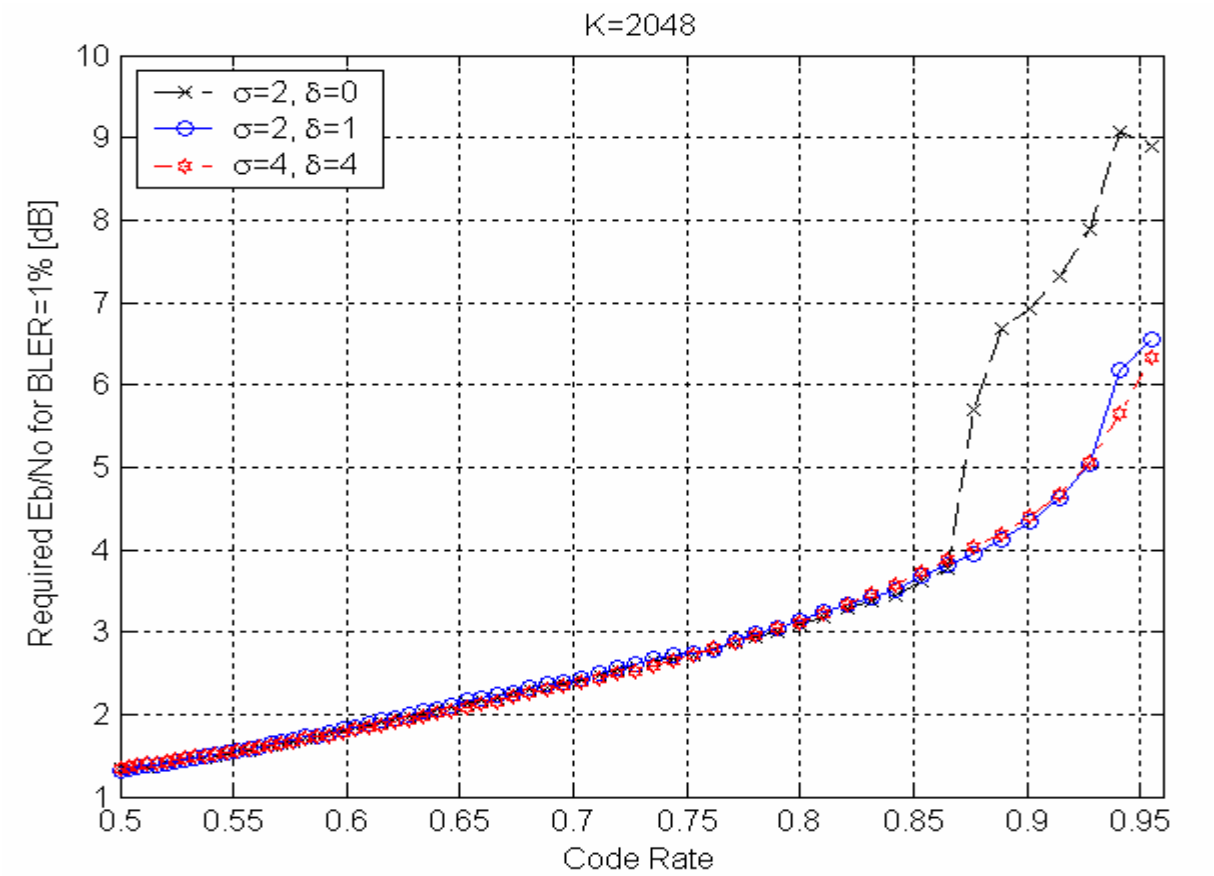

Figure 3 Performance comparison of rate matching algorithms at BLER target=1%. The information block size is fixed at K=2048. The code rate is varied by setting different codeword lengths.

7. References

[1] R1-072604, Ericsson, ETRI, Freescale, ITRI, LGE, Motorola, Qualcomm, Samsung, ZTE, "Way forward on HARQ rate matching for LTE," 3GPP TSG RAN WG1 #49, Kobe, Japan, May 7– 11, 2007.

[2] R1-072621, Nokia Siemens Networks, Nokia, Motorola, Qualcomm, ZTE, Ericsson, LGE, Samsung, TI, ETRI, InterDigital, "LTE rate matching conclusion," 3GPP TSG RAN WG1 #49, Kobe, Japan, May 7– 11, 2007.

[3] R1-072245, Samsung, Qualcomm, LGE, ITRI, "Circular buffer rate matching for LTE," 3GPP TSG RAN WG1 #49, Kobe, Japan, May 7– 11, 2007.

[4] R1-072138, Motorola, "Redundancy version definition for circular buffer rate matching," 3GPP TSG RAN WG1 #49, Kobe, Japan, May 7– 11, 2007.

[5] R1-072137, Motorola, "Turbo rate matching in LTE," 3GPP TSG RAN WG1 #49, Kobe, Japan, May 7– 11, 2007.

[6] R1-072272, Nokia-Siemens Networks and Nokia, "LTE rate matching simulation results," 3GPP TSG RAN WG1 #49, Kobe, Japan, May 7– 11, 2007.

[7] R1-072452, Ericsson, "Performance evaluation of rate matching algorithms," 3GPP TSG RAN WG1 #49, Kobe, Japan, May 7– 11, 2007.

[8] R1-072531, Samsung, "Performance comparison of circular buffer RM and Rel'6 RM," 3GPP TSG RAN WG1 #49, Kobe, Japan, May 7– 11, 2007.

[9] S. Crozier, P. Guinand, and A. Hunt, "On Designing Turbo-Codes with Data Puncturing," Proceedings of the 2005 Canadian Workshop on Information Theory (CWIT 2005), Montréal, Quebec, Canada, June 5-8, 2005.

[10] R1-073029, Ericsson, "Performance of improved circular buffer rate matching design," 3GPP TSG RAN WG1#49bis, Orland, USA, June 25– 29, 2007.

[11] R1-073030, Ericsson, "Considerations of CBRM and HARQ operations," 3GPP TSG RAN WG1#49bis, Orland, USA, June 25– 29, 2007.

[12] 3GPP TS 36.212 v1, "Multiplexing and Channel Coding (Release 8)," 2007.

[13] R1-061737, High Dimension, ITRI, "Modifying puncturing pattern to improve performance and iterations of release 6 turbo coding," 3GPP TSG RAN WG1 LTE *ad hoc*, Cannes, France, June 27 – 30, 2006.

[14] R1-071322, Motorola, "Turbo rate-matching in LTE," 3GPP RAN1#48bis, St. Julians, Malta, March 26 - 30 2007.

[15] R1-071795, Motorola, "Parameters for turbo rate-matching in LTE," 3GPP RAN1#48bis, St. Julians, Malta, March 26 -30 2007.

[16] S. Lin and D. J. Costello, Jr., *Error Control Coding: Fundamentals and Applications*, Prentice-Hall, Inc., 1983.

[17] R1-071589, Ericsson, "Performance study of rate matching algorithms," 3GPP TSG RAN WG1#48bis, St. Julians, Malta, Mar. 26– 30, 2007.

[18] R1-071817, NTT DoCoMo, "Way forward for rate matching study," 3GPP TSG RAN WG1#48bis, St. Julians, Malta, Mar. 26– 30, 2007.