3GPP TSG RAN WG1#12 *TSGR1#12 R1-00-0486*
Seoul, Korea, April 10 – 13, 2000

Agenda item:

Source: **Siemens AG, LGIC**

Title: **Simplification of Rate Matching Description and Optional Correction of Rate Matching Pattern Offset for Repetition**

Document for: **Decision**

1 Introduction

As the flexible multiplexing concept of UMTS is one of the crucial improvements compared to 2G systems, it is not surprising, that much work went into the drafting of the specification and several incremental changes have been done until the present state was finally reached. Of course, during the entire process the functionality defined by the specification was of prime importance, legibility and coherent presentation were only second priorities. As we now have finalised the concepts and have a better understanding of the complete picture than we had when writing individual pieces, it is a good opportunity now to have again a look over some of the chapters and rewrite them.

Several sections have already undergone such a redrafting and we have now done some work to simplify the description of rate matching, in particular the generation of the rate matching pattern itself. We found that both the algorithm for repetition and puncturing, which are described independently, share so much commonality, that a common formulation of both algorithms would surely simplify the specification. At the same time, we found that an optimisation which has recently been done for the puncturing algorithm unintentionally has the reverse effect for the repetition algorithm, it turned out that with the new formulation this can be fixed in a very straight forward way.

We are also working on a reformulation of the bit separation and collection description, integrating this operation into the rate matching step, again maintaining the present functionality. This can be done if the modified formulation presented in this contribution is accepted, bit separation and collection will be simplified and integrated into the pattern generation without making use of the *d*-bits, but a contribution is not ready now.

2 Reformulation of puncturing algorithm

In the present description, the puncturing algorithm selects all bits to be punctured, based on instances, when the error control parameter e becomes negative, thus spreading out punctured bits as evenly as possibly. In the same way, the bits to be transmitted could be selected, again based on instances, when the error control parameter e becomes negative. This will spread out the bits to be transmitted as evenly as possible which is equivalent to spreading out the bits to be punctured. It can be shown indeed that both formulations generate identical patterns, if the parameters (e_{init}, e_{minus}) are adapted accordingly. This new formulation is very similar to the formulation of the repetition algorithm, so both can easily be combined, which will be shown in the next chapter.

We selected the parameter e_{minus} for the new simplified rate matching algorithm so that the pattern is compatible with the current scheme for puncturing, i.e. exactly the same pattern will be generated. We have changed the definition of e_{minus} in all the places where it appears, so be aware that e_{minus} now has a different value than before but the same pattern is generated.

3 Combined puncturing and repetition algorithm

With the modifications described above, the puncturing algorithm is also applicable to repetition, if a bit is output several times, as often as e remains negative even after adding e_{plus}. This is a very straight forward extension.

Again the same pattern will be generated, a possible cyclic shift depends on the selection of e_{ini}. The initial offset value for convolutional coding (especially for rate 1/3 coding) was selected with care to make sure to avoid puncturing exclusively the bitsream from the generator polynomial '711'. Because this polynomial is the strongest of the three constituent polynomials, a better performance can be achieved if only the weaker ones are punctured. This was discussed in WG1 meeting #7 in Hanover in contribution [1], extensive simulation results were provided in WG1 meeting #8 in New York [2] and finally the corresponding CRs for release 99 were agreed in WG1 meeting #9 in Dresden [3], [4].

In order to avoid puncturing the ‘711’ polynomial bit stream exclusively for specific puncturing rates (e.g. 1:3 or 1:6), the offset value e_{ini} was changed accordingly in Dresden. Unfortunately it went undetected, that at the same time the repetition pattern was also shifted: While repetition was performed on the strong polynomial before for the above mentioned rates, now only bits from the weaker polynomial ‘557’ are doubled. This is clearly undesirable as it is contradictory to the objective of the applied change.

It turns out that with the new formulation of the rate matching algorithm, the selection of e_{ini} that creates the optimum pattern for puncturing (identical to the presently defined puncturing pattern) will also create the optimum pattern for repetition (the repetition pattern as defined before of the change agreed in Dresden). While it would be possible to preload e with e_{ini} in the case of repetition, it doesn't seem that this artificial distinction would be reasonable, just for the sake of keeping compatibility with a particular non optimised state.

Obviously the proposed changes do not intend to mandate any specific implementation of the puncturing or repetition algorithm. As usual, only the observable generated pattern is mandatory, not the way how it is generated. If desired, any other procedure can be implemented, including the old formulation of the algorithm. In the latter case the optimum repetition offset can be achieved by substituting $e_{ini}=e_{plus}[i]+1-e_{ini.}$

4 Purely editorial version generating identical repetition patterns

While there are no technical arguments against changing the pattern for repetition, there is of course a procedural argument against such a change. If some companies have already designed an implementation where they have hard coded the offset values, it would be unfair to now request such a change. As the initial offset has very recently been changed (in Dresden) and we propose to use the same pattern as defined prior to Dresden, we would assume that most implementations have implemented the initial offset a parameter that can be changed easily rather than carving it hard in silicon. But of course, is such implementations exist, we would have to accept such an argument, noting that the deadline has already passed by.

We have therefor prepared the attached CRs in a purely editorial way, by preloading e differently for repetition and puncturing. If desired, the CRs can easily be converted to a version that selects the optimum repetition pattern as described in the previous chapter. All that's necessary is to drop the else part of the relevant if statement and to simplify the formula at the end of chapter 4.2.7.2.1.3 "Determination of rate matching parameters for uncoded and convolutionally encoded TrCHs". This formula counts the number of repetitions for fixed position transport channels and therefore depends on the particular selection of the offset. It can be simplified from

$$\Delta N_{i,l}^{TTI} = \left\lceil \frac{|\Delta N_{max}| \times X_i}{N_{max}} \right\rceil \times \operatorname{sgn}(\Delta N_{max})$$

to

$$\Delta N_{i,l}^{TTI} = \left\lfloor \frac{\Delta N_{\max} \times X_i}{N_{\max}} \right\rfloor$$

If agreed by the WG1 delegates these changes could be incorporated into the CR, changing the category to F.

5 Hooks for Release 2000

For end puncturing schemes integrated into rate matching as described in [5], which are in discussion for release 2000, the weights w_m ($m=1..X_i$) give a hook for fine tuning the rate matching patterns. For release 99 $w_m=1$ and the sum of weights $W=X_i$. In this way the weights do not influence the pattern generation. But for release 2000, if w_m is decided to take different values for the bits at the beginning and the end of a block compared to the middle of the block, different local repetition and puncturing rates can be achieved and this has appealing properties regarding performance. It gives a higher flexibility and allows puncturing and repetition within one frame.

6 Summary

The complexity of the current rate matching pattern determination is reduced by deleting the first conditional branch and one of the two inner loops. This allows to merge the puncturing and repetition part and in this way avoids having two different algorithms, one for repetition and one for puncturing.

In summary the change request consists of:

- Simplification of the algorithm for rate matching pattern determination

- Merging of the repetition part and the puncturing part
- Optionally correction of the initial offset for repetition
- No change for puncturing
- Hook for release 2000 ‘End puncturing integrated into Rate Matching’

If desired, the optional optimisation of the repetition pattern can be implemented which provides a further simplification of the algorithm.

7 References

[1] LGIC, SAMSUNG, “Simulation Results of Puncturing Algorithms for Turbo Code”, TSG-RAN WG1 Meeting #7, Tdoc R1-99B89

[2] LGIC, “Simulation Results of Convolutional Code Puncturing with Initial Offset of '1'”, TSG-RAN WG1 Meeting #8, Tdoc R1-99F48

[3] LGIC,“ Revised CR to 25.212 for initial offset value change for convolutional code rate matching”, TSG-RAN WG1 Meeting #9, Tdoc R1-99J11

[4] LGIC,“ Revised CR to 25.222 for initial offset value change for convolutional code rate matching”, TSG-RAN WG1 Meeting #9, Tdoc R1-99J12

[5] Siemens, “End puncturing for short convolutional codes”, TSG-RAN WG1 Meeting #9, Tdoc R1-99J08

3GPP TSG RAN WG1#12
Seoul, Korea, 10-13 April 2000

***Document* R1-00-0486**
e.g. for 3GPP use the format TP-99xxx
or for SMG, use the format P-99-xxx

CHANGE REQUEST

Please see embedded help file at the bottom of this page for instructions on how to fill in this form correctly.

25.212	**CR**	**64**	Current Version:	3.2.0

GSM (AA.BB) or 3G (AA.BBB) specification number ↑ *↑ CR number as allocated by MCC support team*

For submission to:	RAN#8	for approval	X	strategic		*(for SMG use only)*
list expected approval meeting # here ↑		for information		non-strategic		

Form: CR cover sheet, version 2 for 3GPP and SMG   The latest version of this form is available from: ftp://ftp.3gpp.org/Information/CR-Form-v2.doc

Proposed change affects: *(at least one should be marked with an X)*	(U)SIM		ME	X	UTRAN / Radio	X	Core Network	

Source: Siemens AG, LGIC   **Date:** 7.4.2000

Subject: Simplification of Rate Matching Description and Optional Correction of Rate Matching Pattern Offset for Repetition

Work item:

Category:	F	Correction		**Release:**	Phase 2	
	A	Corresponds to a correction in an earlier release			Release 96	
(only one category	B	Addition of feature			Release 97	
shall be marked	C	Functional modification of feature			Release 98	
with an X)	D	Editorial modification	X		Release 99	X
					Release 00	

Reason for change: Simplification of Rate matching generation, can be extended optionally to change of initial offset value for puncturing of convolutional codes to ensure repetition starting at the bits from polynomial '711' output of rate 1/3 convolutional coder
Hook for release 00 'end puncturing integrated into rate matching'

Clauses affected: 4.2.7.1.2.1; 4.2.7.1.2.2; 4.2.7.2.1.3; 4.2.7.2.1.4; 4.2.7.2.2.2; 4.2.7.2.2.3; 4.2.7.5

Other specs affected:	Other 3G core specifications		→ List of CRs:	
	Other GSM core specifications		→ List of CRs:	
	MS test specifications		→ List of CRs:	
	BSS test specifications		→ List of CRs:	
	O&M specifications		→ List of CRs:	

Other comments:

4.2.7.1.2.1 Uncoded and convolutionally encoded TrCHs

```
$R = DN_{ij} \bmod N_{ij}$ -- note: in this context $DN_{ij} \bmod N_{ij}$ is in the range of 0 to $N_{ij}$-1 i.e. -1 mod 10 = 9.
if $R \neq 0$ and $2R \leq N_{ij}$
    then $q = \lceil N_{ij} / R \rceil$
else
    $q = \lceil N_{ij} / (R - N_{ij}) \rceil$
endif
-- note: $q$ is a signed quantity.
if q is even
        then $q' = q + \gcd(|q|, F_i)/F_i$ -- where $\gcd(|q|, F_i)$ means greatest common divisor of $|q|$ and $F_i$
    -- note that $q'$ is not an integer, but a multiple of 1/8
else
        $q' = q$
endif
for x = 0 to $F_i$-1
        $S(I_F(|\lfloor x*q' \rfloor| \bmod F_i)) = (|\lfloor x*q' \rfloor| \operatorname{div} F_i)$
end for
$\Delta N_i = \Delta N_{i,j}$
a = 2
```

For each radio frame, the rate-matching pattern is calculated with the algorithm in section 4.2.7.5, where :

$X_i = N_{i,j}$., and

$e_{ini} = (a \cdot S(n_i) \cdot |\Delta N_i| + 1) \bmod (a \cdot N_{ij})$.

$e_{plus} = a \cdot N_{ij}$

$e_{minus} = a \cdot$ ~~$|\Delta N_i|$~~$(N_{ij}+\Delta N_i)$

puncturing for DN<0, repetition otherwise.

4.2.7.1.2.2 Turbo encoded TrCHs

If repetition is to be performed on turbo encoded TrCHs, i.e. $DN_{i,j} > 0$, the parameters in section 4.2.7.1.2.1 are used.

If puncturing is to be performed, the parameters below shall be used. Index b is used to indicate systematic (b=1), 1st parity (b=2), and 2nd parity bit (b=3).

$a=2$ when b=2

$a=1$ when b=3

$$\Delta N_i = \begin{cases} \lfloor \Delta N_{i,j}/2 \rfloor, & b=2 \\ \lceil \Delta N_{i,j}/2 \rceil, & b=3 \end{cases}$$

If ΔN_i is calculated as 0 for b=2 or b=3, then the following procedure and the rate matching algorithm of section 4.2.7.5 don't need to be performed for the corresponding parity bit stream.

```
    $X_i = \lfloor N_{i,j}/3 \rfloor$,
    $q = \lfloor X_i / |\Delta N_i| \rfloor$
if($q \leq 2$)
        for $x$=0 to $F_i$-1
            $S[I_F[(3x+b-1) \bmod F_i]] = x \bmod 2$;
        end for
else
        if $q$ is even
        then  $q' = q - gcd(q, F_i)/ F_i$   -- where $gcd(q, F_i)$ means greatest common divisor of $q$ and $F_i$
            -- note that $q'$ is not an integer, but a multiple of 1/8
        else     $q' = q$
        endif
        for $x$=0 to $F_i$ -1
            $r = \lceil x*q' \rceil \bmod F_i$;
            $S[I_F[(3r+b-1) \bmod F_i]] = \lceil x*q' \rceil \operatorname{div} F_i$;
        endfor
endif
```

For each radio frame, the rate-matching pattern is calculated with the algorithm in section 4.2.7.5, where:

X_i is as above,

$e_{ini} = (a \cdot S(n_i) \cdot |\Delta N_i| + X_i) \bmod (a \cdot X_i)$, if $e_{ini} = 0$ then $e_{ini} = a \cdot X_i$.

$e_{plus} = a \cdot X_i$

$e_{minus} = a \cdot$ ~~$|\Delta N_i|$~~$(X_i + \Delta N_i)$

4.2.7.2.1.3 Determination of rate matching parameters for uncoded and convolutionally encoded TrCHs

$$\Delta N_i = \Delta N_{max}$$

For compressed mode by puncturing, *DN*DN_i is defined as: DN_i = $\Delta N_{i,\max}^{TTI,cm,m}$, instead of the previous relation.

a=2

$$N_{max} = \max_{l \in TFS(i)} N_{il}^{TTI}$$

For each transmission time interval of TrCH *i* with TF *l*, the rate-matching pattern is calculated with the algorithm in section 4.2.7.5. The following parameters are used as input:

$$X_i = N_{il}^{TTI}$$

$$e_{ini} = 1$$

$$e_{plus} = a \cdot N_{max}$$

~~$e_{minus} = a \cdot |\Delta N_i|$~~ $e_{minus} = a \cdot (N_{\max} + \Delta N_i)$

Puncturing if $\Delta N_i < 0$, repetition otherwise. The values of $\Delta N_{i,l}^{TTI}$ may be computed by counting repetitions or puncturing when the algorithm of section 4.2.7.5 is run. The resulting values of $\Delta N_{i,l}^{TTI}$ can be represented with following expression.

$$\Delta N_{i,l}^{TTI} = \left\lceil \frac{|\Delta N_{max}| \times X_i}{N_{max}} \right\rceil \times \operatorname{sgn}(\Delta N_{max})$$

4.2.7.2.1.4 Determination of rate matching parameters for Turbo encoded TrCHs

If repetition is to be performed on turbo encoded TrCHs, i.e. $\Delta N_{max} > 0$, the parameters in section 4.2.7.2.1.3 are used.

If puncturing is to be performed, the parameters below shall be used. Index *b* is used to indicate systematic (*b*=1), 1st parity (*b*=2), and 2nd parity bit (*b*=3).

a=2 when *b*=2

a=1 when *b*=3

The bits indicated by *b*=1 shall not be punctured.

$$\Delta N_i = \begin{cases} \lfloor \Delta N_{max}/2 \rfloor, & b = 2 \\ \lceil \Delta N_{max}/2 \rceil, & b = 3 \end{cases}$$

In Compressed Mode by puncturing, the following relations are used instead of the previous ones:

DN_i = ë $\Delta N_{i,\max}^{TTI,cm,m}$ /2 û , *b=2*

DN_i = é $\Delta N_{i,\max}^{TTI,cm,m}$ /2 ù , *b=3*

$$N_{max} = \max_{l \in TFS(i)} (N_{il}^{TTI} / 3)$$

For each transmission time interval of TrCH *i* with TF *l*, the rate-matching pattern is calculated with the algorithm in section 4.2.7.5. The following parameters are used as input:

$$X_i = N_{il}^{TTI} / 3$$

$$e_{ini} = N_{max}$$

$$e_{plus} = a \cdot N_{max}$$

~~$e_{\min us} = a \cdot |\Delta N_i|$~~ $e_{\min us} = a \cdot (N_{\max} + \Delta N_i)$

The values of $\Delta N_{i,l}^{TTI}$ may be computed by counting puncturing when the algorithm of section 4.2.7.5 is run. The resulting values of $\Delta N_{i,l}^{TTI}$ can be represented with following expression.

$$\Delta N_{i,l}^{TTI} = -\left\lfloor \frac{|\lfloor \Delta N_{max} / 2 \rfloor| \times X_i}{N_{max}} + 0.5 \right\rfloor - \left\lfloor \frac{|\lceil \Delta N_{max} / 2 \rceil| \times X_i}{N_{max}} \right\rfloor$$

In the above equation, the first term of the right hand side represents the amount of puncturing for *b*=2 and the second term represents the amount of puncturing for *b*=3

.

4.2.7.2.2.2 Determination of rate matching parameters for uncoded and convolutionally encoded TrCHs

$$\Delta N_i = \Delta N_{il}^{TTI}$$

a=2

For each transmission time interval of TrCH *i* with TF *l*, the rate-matching pattern is calculated with the algorithm in section 4.2.7.5. The following parameters are used as input:

$$X_i = N_{il}^{TTI}$$

$$e_{ini} = 1$$

$$e_{plus} = a \cdot N_{il}^{TTI}$$

~~$e_{\min us} = a \cdot |\Delta N_i|$~~ $e_{\min us} = a \cdot (N_{il}^{TTI} + \Delta N_i)$

puncturing for $\Delta N_i < 0$, repetition otherwise.

4.2.7.2.2.3 Determination of rate matching parameters for Turbo encoded TrCHs

If repetition is to be performed on turbo encoded TrCHs, i.e. $\Delta N_{il}^{TTI} > 0$, the parameters in section 4.2.7.2.2.2 are used.

If puncturing is to be performed, the parameters below shall be used. Index *b* is used to indicate systematic (*b*=1), 1st parity (*b*=2), and 2nd parity bit (*b*=3).

a=2 when *b*=2

a=1 when *b*=3

The bits indicated by *b*=1 shall not be punctured.

$$\Delta N_i = \begin{cases} \lfloor \Delta N_{il}^{TTI} / 2 \rfloor, & b = 2 \\ \lceil \Delta N_{il}^{TTI} / 2 \rceil, & b = 3 \end{cases}$$

For each transmission time interval of TrCH *i* with TF *l*, the rate-matching pattern is calculated with the algorithm in section 4.2.7.5. The following parameters are used as input:

$$X_i = N_{il}^{TTI} / 3,$$

$$e_{ini} = X_i,$$

$$e_{plus} = a \cdot X_i$$

~~$e_{\min us} = a \cdot |\Delta N_i|$~~ $e_{\min us} = a \cdot (X_i + \Delta N_i)$

4.2.7.5 Rate matching pattern determination

Denote the bits before rate matching by:

$x_{i1}, x_{i2}, x_{i3}, \ldots, x_{iX_i}$, where *i* is the TrCH number and the sequence is defined in 4.2.7.3 for uplink or in 4.2.7.4 for downlink. Parameters X_i, e_{ini}, e_{plus}, and e_{minus} are given in 4.2.7.1 for uplink or in 4.2.7.2 for downlink. The weights are defined by w_m=1 and W= X_i is the sum of the weights w_m (m=1..X_i). Note that the weights are introduced for future compatibility.

The rate matching rule is as follows:

```
if puncturing is to be performed
    $e = e_{plus}+1-e_{ini}$          -- initial error between current and desired puncturing ratio
else
    $e = e_{ini}$                     -- initial error between current and desired puncturing ratio
endif
    $m = 1$             -- index of current bit
    $e_{plus}=e_{plus}-a*(X_i-W)$
    do while $m <= X_i$
        $e = e – w_m*e_{minus}$          -- update error
        if e ~~<=~~> 0 then             -- check if bit number m should be punctured
            set bit $x_{i,m}$ to $d$ where $d$ Ï {0, 1}
        else
            do
                select bit $x_{i,m}$
                $e = e + e_{plus}$    -- update error
            while $e <= 0$
        end if
        $m = m + 1$                 --  next bit
    end do
~~else~~
    ~~$e = e_{ini}$              -- initial error between current and desired puncturing ratio~~
    ~~$m = 1$                 -- index of current bit~~
    ~~do while $m <= X_i$~~
        ~~$e = e – e_{minus}$           -- update error~~
        ~~do while $e <= 0$         -- check if bit number m should be repeated~~
            ~~repeat bit $x_{i,m}$~~
            ~~$e = e + e_{plus}$ -- update error~~
        ~~end do~~
        ~~$m = m + 1$ -- *next bit*~~
    ~~end do~~
~~end if~~
```

Bits are output in the order in which they are selected by the algorithm, thus ~~A~~a repeated bit is placed directly after the original one.

3GPP TSG RAN WG1#12
Seoul, Korea, 10-13 April 2000

***Document* R1-00-0486**
e.g. for 3GPP use the format TP-99xxx
or for SMG, use the format P-99-xxx

CHANGE REQUEST

Please see embedded help file at the bottom of this page for instructions on how to fill in this form correctly.

25.222	**CR**	**32**	Current Version:	3.2.0

GSM (AA.BB) or 3G (AA.BBB) specification number ↑ *↑ CR number as allocated by MCC support team*

For submission to:	RAN#8	for approval	X	strategic		*(for SMG use only)*
list expected approval meeting # here ↑		for information		non-strategic		

Form: CR cover sheet, version 2 for 3GPP and SMG The latest version of this form is available from: ftp://ftp.3gpp.org/Information/CR-Form-v2.doc

Proposed change affects: (U)SIM [ ] ME [X] UTRAN / Radio [X] Core Network [ ]
(at least one should be marked with an X)

Source:	Siemens AG, LGIC	**Date:**	6.4.2000
Subject:	Simplification of Rate Matching Description and Optional Correction of Rate Matching Pattern Offset for Repetition		
Work item:			

Category:	F	Correction	X	**Release:**	Phase 2	
	A	Corresponds to a correction in an earlier release			Release 96	
(only one category	B	Addition of feature			Release 97	
shall be marked	C	Functional modification of feature			Release 98	
with an X)	D	Editorial modification			Release 99	X
					Release 00	

Reason for change:	Simplification of Rate matching generation, can be extended optionally to change of initial offset value for puncturing of convolutional codes to ensure repetition starting at the bits from polynomial '711' output of rate 1/3 convolutional coder Hook for release 00 'end puncturing integrated into rate matching'
Clauses affected:	4.2.7.1.1; 4.2.7.1.2; 4.2.7.3

Other specs affected:	Other 3G core specifications		→ List of CRs:	
	Other GSM core specifications		→ List of CRs:	
	MS test specifications		→ List of CRs:	
	BSS test specifications		→ List of CRs:	
	O&M specifications		→ List of CRs:	

Other comments:	

4.2.7.1.1 Uncoded and convolutionally encoded TrCHs

```
$a = 2$

$\Delta N_i = \Delta N_{i,j}$

$X_i = N_{i,j}$

$R = \Delta N_{ij} \bmod N_{ij}$ -- note: in this context $\Delta N_{ij} \bmod N_{ij}$ is in the range of 0 to $N_{ij}-1$ i.e. $-1 \bmod 10 = 9$.

    if $R \neq 0$ and $2R \leq N_{ij}$

        then $q = \lceil N_{ij} / R \rceil$

    else

        $q = \lceil N_{ij} / (R - N_{ij}) \rceil$

    endif

-- note: q is a signed quantity.


    If q is even

        then $q' = q + \gcd(|q|, F_i)/F_i$ -- where $\gcd(|q|, F_i)$ means greatest common divisor of $|q|$ and $F_i$

-- note that q' is not an integer, but a multiple of 1/8

    else

        $q' = q$

    endif

for $x = 0$ to $F_i-1$

    $S(I_F(|\lfloor x*q' \rfloor| \bmod F_i)) = (|\lfloor x*q' \rfloor| \operatorname{div} F_i)$

end for

$e_{ini} = (a \cdot S(n_i) \cdot |\Delta N_i| + 1) \bmod (a \cdot X_i)$

$e_{plus} = a \cdot X_i$

$e_{minus} = a^{\chi}$ ~~$|DN_i|$~~ $(X_i+DN_i)$
```

puncturing for $DN_i<0$, repetition otherwise.

4.2.7.1.2 Turbo encoded TrCHs

If repetition is to be performed on turbo encoded TrCHs, i.e. $DN_{i,j} >0$, the parameters in section 4.2.7.1.1 are used.

If puncturing is to be performed, the parameters below shall be used. Index b is used to indicate systematic (b=1), 1st parity (b=2), and 2nd parity bit (b=3).

$a = 2$ when b=2

$a = 1$ when b=3

$$\Delta N_i = \begin{cases} \lfloor \Delta N_{i,j}/2 \rfloor, & b=2 \\ \lceil \Delta N_{i,j}/2 \rceil, & b=3 \end{cases}$$

If ΔN_i is calculated as 0 for *b*=2 or *b*=3, then the following procedure and the rate matching algorithm of section 4.2.7.3 don't need to be performed for the corresponding parity bit stream.

```
    $X_i = \lfloor N_{i,j} / 3 \rfloor$,
    $q = \lfloor X_i / |\Delta N_i| \rfloor$
    if($q \leq 2$)
        for $x$=0 to $F_i$-1
        $S[I_F[(3x+b-1) \bmod F_i]] = x \bmod 2$; end for
    else
        if $q$ is even
            then  $q' = q - gcd(q, F_i)/ F_i$   -- where $gcd(q, F_i)$ means greatest common divisor of $q$ and $F_i$
-- note that $q'$ is not an integer, but a multiple of 1/8
        else    $q' = q$
    endif
    for $x$=0 to $F_i - 1$
        $r = \lceil x*q' \rceil \bmod F_i$;
    $S[I_F[(3r+b-1) \bmod F_i]] = \lceil x*q' \rceil \operatorname{div} F_i$;
    endfor
    endif
```

For each radio frame, the rate-matching pattern is calculated with the algorithm in section 4.2.7.3, where:

X_i is as above,

$e_{ini} = (a \cdot S(n_i) \cdot |\Delta N_i| + X_i) \bmod (a \cdot X_i)$, if $e_{ini} = 0$ then $e_{ini} = a \cdot X_i$.

$e_{plus} = a \cdot X_i$

$e_{minus} = a \cdot$ - ~~$|\Delta N_i|$~~ $(X_i + \Delta N_i)$

4.2.7.3 Rate matching pattern determination

The bits input to the rate matching are denoted by $x_{i1}, x_{i2}, x_{i3}, \ldots, x_{iX_i}$, where i is the TrCH and X_i is the parameter given in section 4.2.7.1.1 and 4.2.7.1.2. The weights are defined by w_m=1 and W= X_i is the sum of the weights w_m (m=1..X_i). Note that the weights are introduced for future compatibility.

Note that the transport format combination number j for simplicity has been left out in the bit numbering.

The rate matching rule is as follows:

```
if puncturing is to be performed
    e = e_plus+1-e_ini    -- initial error between current and desired puncturing ratio
else
    e = e_ini             -- initial error between current and desired puncturing ratio
endif
    m = 1                 -- index of current bit
    e_plus=e_plus-a*(X_i-W)
    do while m <= X_i
        e = e – w_m*e_minus        -- update error
        if e ~~<=~~ > 0 then       -- check if bit number m should be punctured
            set bit x_{i,m} to d where d ∈ {0, 1}
        else
            do
                select bit x_{i,m}
                e = e + e_plus     -- update error
            while e <= 0
        end if
        m = m + 1                  -- next bit
    end do
~~else~~
    ~~e = e_ini      -- initial error between current and desired puncturing ratio~~
    ~~m = 1          -- index of current bit~~
    ~~do while m <= X_i~~
        ~~e = e – e_minus          -- update error~~
        ~~do while e <= 0          -- check if bit number m should be repeated~~
            ~~repeat bit x_{i,m}~~
            ~~e = e + e_plus       -- update error~~
        ~~end do~~
        ~~m = m + 1                -- next bit~~
    ~~end do~~
```

~~end if~~

Bits are output in the order in which they are selected by the algorithm, thus a~~A~~ repeated bit is placed directly after the original one.