**Military Technical College**
**Kobry Elkobbah,**
**Cairo, Egypt**

**5th International Conference**
**on Electrical Engineering**
**ICEENG 2006**

# AN ANALYTICAL PERFORMANCE BOUNDS OF NON-SYSTEMATIC PUNCTURED PARALLEL CONCATENATED CODES

Moataz Mohamed Salah*

## ABSTRACT

A class of powerful error-correcting codes called parallel concatenated codes, or turbo codes, have performance superior than all other coding techniques. Turbo codes have been shown to achieve bit error rate performance close to Shannon's limit.

In this paper, based on random puncturing of non-systematic bits of low rate turbo codes, we derive an analytical performance bound for high rate parallel concatenated turbo codes. The new performance bound calculations and evaluations have been investigated and compared with the simulation results.

**KEYWORDS:** Turbo Codes, Space-Time Codes, Diversity

## 1. INTRODUCTION

Parallel concatenated codes, turbo codes [1], have been shown to achieve near-Shannon-limit error correction performance with relatively simple component codes and large interleavers. For a bit error probability of $10^{-5}$ and code rate = 1/2, it has been shown that an *Eb/N0* of 0.7 dB is required for block lengths of 65,536 bits. A typical turbo code encoder is shown in Fig. 1. This encoder consists of two binary rate 1/2 convolutional encoders, an interleaver of length *N*, along with puncturing and multiplexing devices. Without the puncturing device, the encoding is rate 1/3. The decoding process relies on iterative processing in which each component decoder takes advantage of the work performed by the other in the previous step.

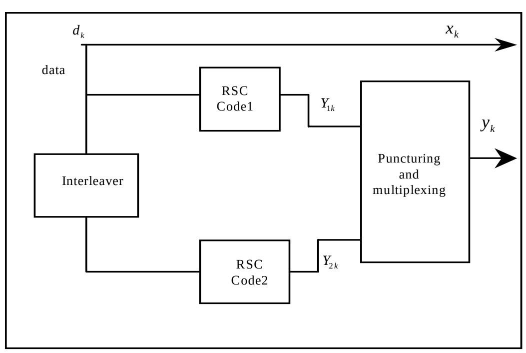


Fig. 1. Turbo-Code Encoder.

* Egyptian Armed Forces

A punctured turbo code is a high-rate code obtained by the periodic deleting of specific code symbols from the output of a low-rate code. The resulting high-rate code depends on both the low-rate code (original code) and on the number and specific positions of the punctured symbols. Since the first appearance of turbo codes, puncturing has been used to increase the code rate. Studies [2, 3, 4] that focused on punctured turbo codes have relied on simulation due to complexity associated in the analytical modeling.

Many of the theoretical and structural properties of turbo codes are discussed in [4, 5, 6, 7]. The most complete works in the analytic bounds are presented in [6, 7]. In [6], the authors derive an analytical upper bound for the average performance of the coding scheme. The average upper bound is constructed by averaging over all possible interleaver configurations. This upper bound is shown to be independent of the interleaver used and reveals the influence of the interleaver length on the code performance. In [7], the authors apply the transfer function bound techniques to obtain an upper bound on the probability of bit error. In their study, the authors developed a method for a recursive computation of the Weight Enumerating Function (WEF) of the convolutional code that is used as a constituent encoder with random interleaving for the calculation of the bound of the turbo code.

This paper extends the results presented in [6] and [7] by deriving an upper bound of the punctured turbo codes. This is accomplished by, for each codefragment, averaging over all punctured positions, which in turn yields all possible punctured weights. The derivation of the punctured bound is presented in the next Section. Section 3 presents the calculation and evaluation of the punctured bound in case of AWGN channel.

## 2. DERIVATION OF THE PUNCTURED BOUND

The error performance of punctured convolutional codes may be evaluated by computing the upper bounds on the bit error probability. Using the transfer function bounding technique, the upper bound of the bit error probability can be evaluated. The transfer function of a convolutional code is evaluated by solving equations describing the transitions between the states of the finite-state encoder. For punctured convolutional codes, the first step in the evaluation is the drawing of a proper state diagram for the encoder under consideration. Knowledge of the perforation matrix is necessary to get the transfer function of the punctured code.

For punctured convolutional codes, extensive computer searches have been performed to get an optimal (maximum free distance) puncturing pattern to obtain higher rates from the original low-rate code. Once the puncturing pattern is obtained, it is applied to the original low-rate trellis to get the WEF of the higher rate code. The complexity here is how to get the weight enumerating function of the parallel concatenated punctured code. For turbo codes, the computer search for optimum puncturing patterns is very complex due to the existence of the interleaver. One of the goals of this research is to get the WEF of the punctured turbo code that is independent of specific puncturing pattern.

Using the conventional union bound on the probability of codeword error, $P_w$, as follows:

$$P_w \leq \sum_d t(d)\ P_2(d) \qquad (1)$$

Where *t(d)* represents the number of codewords of length *N* and weight d and $P_2(d)$ is the pair-wise error probability for codewords with weight *d*. In case of puncturing the output codewords, the output punctured weight will be denoted by *p*. Then for punctured turbo codes, the probability of codeword error can be written as follows:

$$P_w \le \sum_p t(p)\ P_2(p) \tag{2}$$

Where *t(p)* represents the number of codewords of length N and weight p and $P_2(p)$ is the pair-wise error probability for the punctured codeword with weight *p*.

The problem now is how to get *t(p)* from knowledge of the original (before puncturing) code, *t(d)*. From equation (2), the probability of bit error, $P_{bit}$, can be defined as follows:

$$P_{bit} \le \sum_{i=1}^{N} \frac{i}{N}\ t(p)\ P_2(p) \tag{3}$$

Let *t(p,d)* represents number of codewords with punctured weight *p* produced from puncturing a codeword with weight d. Then, we can write *t(p,d)* in terms of *t(d)* as follows:

$$t(p,d) = t(d)p(p/d) \tag{4}$$

Taking summation over d, then

$$t(p) = \sum_d\ t(d)\ p(p/d) \tag{5}$$

Let *t(i,d)* represents the number of codewords of length *N* and weight d which generated from input sequence with weight *i*. Using the uniform interleaver introduced in [6] to calculate *t(d)* from the knowledge of *t(i,d)* by averaging over all possible interleavers of length *N* as follows:

$$t(i,d) = \binom{N}{i} p(d/i) \tag{6}$$

Where $\binom{N}{i}$ is the number of input sequences of length *N* with weight *i*, then,

$$t(d) = \sum_{i=1}^{N} \binom{N}{i} p(d/i) \tag{7}$$

Substituting in equation (5), we get:

$$t(p) = \sum_d \sum_i \binom{N}{i} p(d/i)p(p/d) \tag{8}$$

The probability of output codeword with weight d of turbo code can be written in terms of the individual output weights $d_0, d_1, d_2$, where $d_0$ represents the output weight of the systematic part and $d_1$ and $d_2$ represent the output weight of each constituent code fragment, then $d = d_0 + d_1 + d_2$. Since each codefragment weight is independently generated, then:

$$p(d/i) = \sum_{\substack{d_0, d_1, d_2: \\ d = d_0 + d_1 + d_2}} p(d_0, d_1, d_2/i) \tag{9}$$

Where $p(d_0, d_1, d_2/i) = p(d_0/i)p(d_1/i)p(d_2/i)$, substitute in equation (9), then

$$p(d/i) = \sum_{d_0} \sum_{\substack{d_1 \\ :d = d_0 + d_1 + d_2}} \sum_{d_2} p(d_0/i)p(d_1/i)p(d_2/i) \tag{10}$$

Assuming no puncturing of systematic bits, then

$$p(d_o/i)=\begin{cases}1 & i=d_0\\ 0 & i\neq d_0\end{cases} \tag{11}$$

From equation (6), we can get $p(d_1/i)$ and $p(d_2/i)$ as follows:

$$p(d_j/i)=\frac{t(i,d_j)}{\binom{N}{i}} \qquad j=1,2 \tag{12}$$

Now, calculate the probability of getting a punctured codeword of weight *p* from puncturing a codeword with weight d, $p(p/d)$, from the individual components of the code. Since each codefragment puncturing is independently processed, then:

$$p(p/d)=\sum_{p_0}\sum_{\substack{p_1\\ :p=p_0+p_1+p_2}}\sum_{p_2} p(p_0/d_0)p(p_1/d_1)p(p_2/d_2) \tag{13}$$

Assuming no puncturing of systematic bits, then

$$p(p_o/d_0)=\begin{cases}1 & i=d_0\\ 0 & i\neq d_0\end{cases} \tag{14}$$

To calculate $p(p_j/d_j)$ using random puncturing. Since there is no specific optimal pattern for puncturing, by averaging over all possible puncturing patterns that yield different punctured weights, so use the hypergeometric probability distribution function to get the probabilities required. The hypergeometric distribution [8] is a well known distribution in probability theory.

For a given codeword of length *N* and weight *d* generated from a codeword of information weight i, the process is to puncture (delete) *M* bits from the *N* bits (i.e., randomly choose *N-M* bits from *N* bits to survive). The hypergeometric probability distibution gives the probability that a codeword with weight *p* is generated from puncturing a codeword with weight d as follows:

$$P\left(p_j \mid d_j\right)=\frac{\binom{d_j}{p_j}\binom{N-d_j}{N-M-p_j}}{\binom{N}{N-M}} \qquad j=1,2 \quad 0\le p_j\le d_j \tag{15}$$

## 3. APPLICATION OF THE BOUND AND PERFORMANCE EVALUATION

This section shows the results of the derived punctured bound for different turbo code rates derived from the original code rate 1/3 turbo code. Using the simulation model, obtained results are compared with derived analytic bounds. The simulation model uses a turbo code with two identical parallel concatenated recursive systematic convolutional constitute encoders, separated by a random interleaver. Each encoder uses octal generators 5 (feedforward) and 7 (feedback) and is denoted as (5/7)8. The encoded bits are punctured and then modulated using Binary Phase Shift Key (BPSK) modulation, and then transmitted over an Additive White Gaussian Noise (AWGN) channel. The decoding is performed using a Soft Output Viterbi Alogorithm (SOVA) as a constitute decoder [9] with 8 iterations.

The derived punctured bound of the turbo code with two identical constituent encoders is applied with generator functions (5/7)8. The algorithm in [7] is used to calculate the weight enumerating function, *t(i,d)*, of the constituent codes.

It is assumed that the channel has AWGN with two-sided noise power spectral density of $N_0/2$. Using BPSK modulation, the pair-wise probability is given by [10]:

$$P_2(p) = Q\left(\sqrt{\frac{2prE_b}{N_0}}\right) \tag{16}$$

where r is the code rate of the code, $E_b/N_0$ is the signal-to-noise ratio per information bit, $v$ is the codeword weight, and $Q(x)$ is the tail integral of a standard Gaussian density with zero mean and unit variance.

Fig. 2 shows the bound for frame length of 100 bits for different code rates of 1/3, 2/5, 1/2, and 2/3 respectively. As expected, the punctured bounds diverge at signal-to-noise ratios larger than those that occur at rate 1/3.

The abrupt transition of the bound occurs when the signal-to-noise ratio, $E_b/N_0$, drops below the threshold determined by the computation cutoff rate $R_0$, i.e., when $E_b/N_0 < -1/r \ln (2^{1-r} - 1)$ for a code rate r [7]. Fig. 2 shows the abrupt change occurring for rates 1/3, 2/5, 1/2, and 2/3 at 2.03 db, 2.2 db, 2.5 db, and 3.1 db, respectively. The error floor (the low slope region of the performance curve where the error rate decreases very slowly with increasing the signal-to-noise ratio) still exists with the punctured bound.

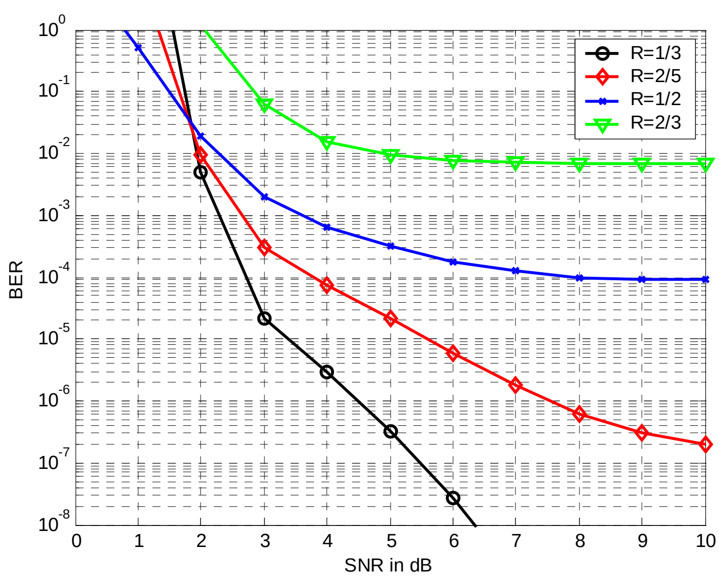


Fig. 2. Analytical performance bound for various code rates (*N*=100).

In computing these bounds for low signal-to-noise ratio, there is a false convergence. For frame length of 100 bits, Fig. 3 illustrates this false convergence behavior for code rate 2/3 at signal-to-noise ratio $E_b/N_0$ =3 db (just below the threshold). On the other hand, when the signal-to-noise ratio, $E_b/N_0$, is above the threshold, false convergence is not a problem. Fig. 4 show the probability of error as a function of information weight i and output weight of each codefragment, $p_j$. Fig. 4 shows how quickly the probability of error convergences when $E_b/N_0$ =4 db. It is only some terms from i and $p_j$ are needed for convergence (around 20 for both and independent of the frame length).

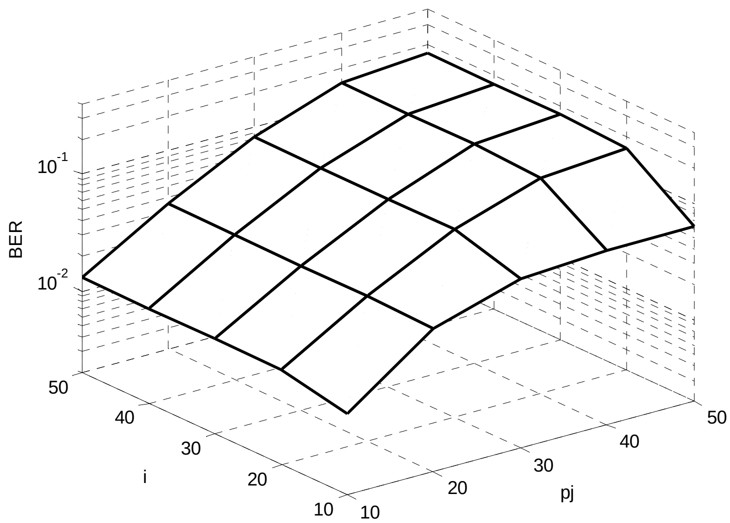


Fig. 3. False convergence behavior at $E_b/N_0$ =3 db (*N*=100, rate=2/3).

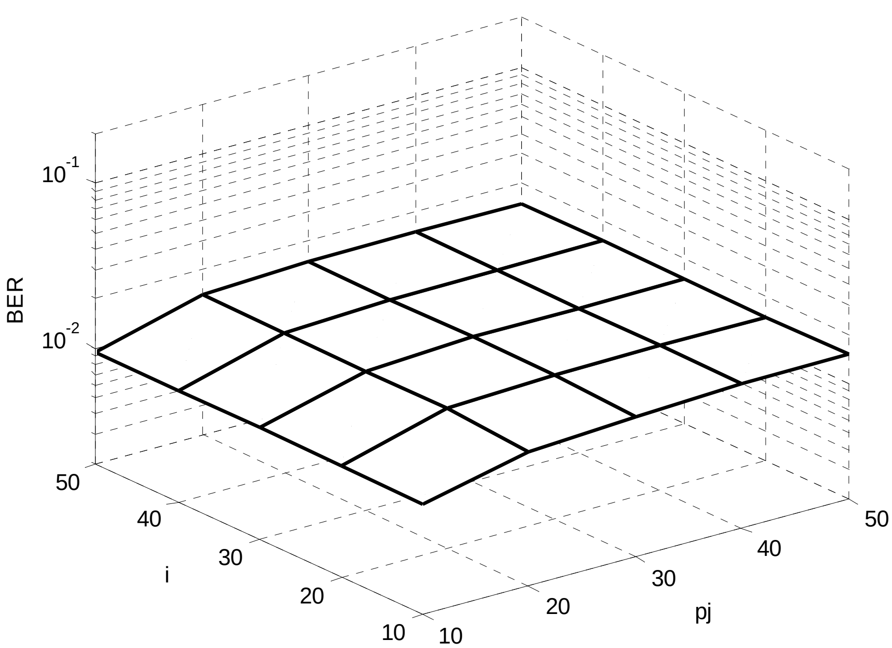


Fig. 4. Convergence behavior at $E_b/N_0$ =4 db (*N*=100, rate=2/3).

Fig. 5 shows a simulated punctured turbo code with a 200-bit frame length compared with the analytical punctured bound at rates 2/5 and 1/2. At higher signal-to-noise ratios (greater than approximately 2 dB), the bound accurately predicts the turbo decoder performance. At signal-to-noise ratios less than $R_0$, simulation is the only way to predict the performance of turbo codes due to the divergence in the performance of the analytical bound in this region.

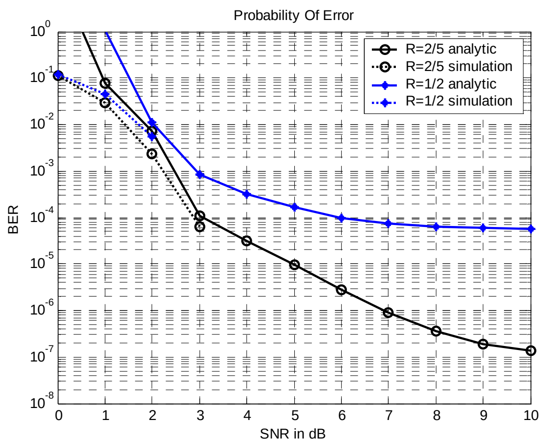


Fig. 5. Analytical performance bound versus simulated bit error rate ($N$=200).

## 4. CONCLUSIONS

In this paper, the hypergeometric puncturing was introduced. The introduction of the hypergeometric puncturing device makes the derivation of the analytical punctured bound of turbo codes tractable. The hypergeometric puncturing device allows for averaging over all possible punctured positions. Simulation results in AWGN were also presented along with the analytical bound. The analytic performance bound was compared with the simulation results obtained for various code rates. The comparison shows that the two bounds, analytical and simulation are identical at higher signal-to-noise ratios but diverge at lower signal-to-noise ratios. This bound can also be extended to be used with different channels and for assistance in designing punctured turbo codes.

## 5. REFERENCES:

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near Shannon limit error-correcting coding and decoding: TURBO-CODES,” *in Proceeding of 1993 IEEE International Conference On Communication* (Geneva, Switzerland, 1993), pp. 1064-1070.

[2] M. Oberg and P.H. Siegel, “The effect of puncturing in turbo encoders,” *International Symposium On Turbo codes & related topics*, Brest, France, pp.184-187, Sep. 1997.

[3] F. Vatta, G.Montorsi, and F. Bahich, “Achievable performance of turbo codes over the correlated Rician channel,” *IEEE Transactions on Communications*, Vol.51, No. 1,Jan..2003.

[4] S. Benedetto and G. Montorsi, “Design guidelines of parallel concatenated convolutional codes,” *IEEE Global Communications GLOBECOM*, pp. 2273-2277, November 1995.

[5] D. Divsalar, S. Dolinar, R. J. McEliece, and F. Pollara, “Performance analysis of turbo codes,” *IEEE Military Communications MILCOM*, pp. 91-96, November 1995.

[6] S. Benedetto and G. Montorsi, “Unveiling turbo codes: some results on parallel concatenated coding schemes,” *IEEE Transactions on Information Theory*, Vol. 42, No. 2, pp. 409-428, Mar. 1996.

[7] D. Divsalar, S. Dolinar, F. Pollara, and R. J. McEliece, “Transfer function bounds on the performance of turbo codes,” *TDA Progr. Rep. 42-122, Jet Propulsion Lab., Pasadena, CA*, pp. 44-55, Aug. 15, 1995.

[8] R. V. Hogg and A. T. Craig, *Introduction to Mathematical Statistics,* Englewood Cliffs, NJ, Prentice Hall, 1995, pp. 517.

[9] J. Hagenauer and P. Hoeher, “A Vitrbi Algorithm with Soft-Decision Outputs and its Applications,” *Proc. of Global Communications Globecom’89*, Dallas, Vol. 3, pp. 47.1.1-47.1.7, Nov. 1989.

[10] J. G. Proakis, *Digital Communications*, New York: McGraw-Hill, 3rd ed., 2000.