# Turbo Compression

*Andrej Rikovsky, Advisor: Pavol Hanus*

***Abstract – Turbo codes which performs very close to channel capacity in channel coding can be also used to obtain very efficient source coding schemes. In this paper lossless turbo compression algorithm has been presented. Turbo source coding can be very efficiently used in combination with channel coding applying the concept of incremental and decremental redundancy algorithm. Performance of turbo compression can be analyzed by modified ten Brick's EXIT charts.***

## I. INTRODUCTION AND BACKGROUND

The theory of data compression was first formulated by Claude E. Shannon in his monumental work "A Mathematical Theory of Communication", in 1948. He, as a first, stated the basics of source coding and established a fundamental limit to lossless data compression. This limit is called entropy rate and it is defined by the statistical distribution of information source. Since the time of invention of this limit, engineers are trying to achieve it by applying different compression schemes.

After the invention of "Turbo Codes" in 1993 by C. Berrou and A. Glavieux, a new channel coding technique was introduced. Since channel coding and source coding are dual problems, the same principle was successfully applied to data compression of binary memoryless sources [1], [2], [3].

The secret of turbo principle lies in the use of a feedback loop which allows iterative decoding. This is used in channel coding to correct the errors after transmission over a noisy channel. In source coding scenario we don't have any channel but in order to decrease the redundancy of encoded data the bits are heavily punctured depending on the desired compression rate. In order to guarantee lossless compression, we use algorithm of decremental redundancy. The redundancy of data is removed step-by-step as long as the decoder can guarantee perfect reconstruction. This approach is dual to incremental redundancy presented in hybrid automatic repeat request (ARQ Type II) scheme with forward error correction (FEC), where additional parity bits are gradually transmitted through a noisy channel until the decoder can correct all errors**.**

In order to achieve a very efficient scheme of communication, both techniques turbo source coding and turbo channel coding can be applied together using combined decremental and incremental redundancy [4], [5].

The issue of making the best compression scheme can be investigated by applying ten Brink's EXIT chart analysis to select the most efficient component codes and puncturing matrices to compress discrete memoryless source.

## II. TURBO COMPRESSION PRINCIPLE, IT'S ANALYSIS AND APPLICATIONS

### *A. Problem statement*

The goal of data compression is to represent an information source as accurately as possible using the fewest number of bits. It means that by applying any source coding schemes we are trying to decrease the redundancy as much as possible. In this paper we are assuming lossless compression, thus perfect reconstruction of the compressed data is guaranteed.

Let *U* be a discrete memoryless source emitting independent and identically distributed symbols from alphabet $U = \{1,2,...,L\}$ characterized by probability mass function ~ *p(u)*. According to Shannon, the optimal compression rate for the source *U* is given by the entropy

$$H(U) = E\left\{\log\left(\frac{1}{p(u)}\right)\right\} = -\sum_{i=1}^{L} p(u_i)\log(p(u_i)) \,. \quad (1)$$

In order to achieve this rate the turbo compression encoding is done as follows. Source encoder for source *U* takes a block of N symbols $U^N = U_1, U_2, ..., U_N$ and encodes it to a binary codeword

$X^K = X_1, X_2, ..., X_K$ with $X_i \in \{0,1\}$, $X_i \in \{-1,+1\}$ respectively. Let us assume binary source which can be easily constructed by simple bit mapping of every symbol to a binary codeword of length $l = \lceil \log_2 L \rceil$ as shown in Fig.1. This codeword is passed through a turbo encoder and finally the encoded sequence is punctured to achieve the desired compression rate.

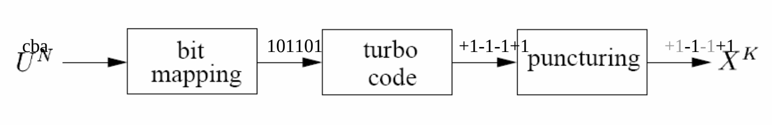


**Fig. 1.** Encoder for turbo coding

To fulfill the requirements of good lossless data compression schemes we have to guarantee perfect reconstruction of compressed data and design turbo code and puncturing matrix in order to be able to reach the entropy of the source. The design process can be very well analyzed by using ten Brick's EXIT charts described below.

*B. Source encoding*

As proposed in [1] the source encoder uses turbo code to compress the source data. The turbo codes are well known to be good channel codes as they perform very close to channel capacity. Applied to source coding scenario it is assumed to perform very close to source entropy rate, which is fundamental limit stated by Shannon in his source coding theorem. It tells us that the source block $U^N$ can be perfectly reconstructed from the sequence $X^K$ with the length $K \cong NH(U)$ for N sufficiently large.

Let us assume a binary source used to encode the data in parallel concatenation scheme as shown in Fig. 2, and pass the sequence $U^N$ and interleaved sequence $\prod(U^N)$ through rate 1 convolutional codes (or scrambles) to generate the parity sequences $P_1^N$ and $P_2^N$. The parity bits are than heavily punctured to obtain the desired compression rate $R = \frac{K}{N}$. Since we are assuming a binary source (L=2), by knowing the source bits statistics a priori (probabilities *p(u=0)* and *p(u=1)*), we can discard the information bits and take into consideration only parity sequences.

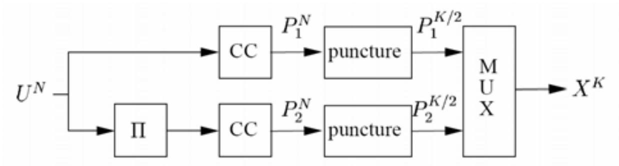


**Fig. 2.** Parallel concatenation of convolution codes CC (R=1) with interleaver

If the chosen puncturing scheme is taken in random manner, we can compare it to "Binary Erasure Channel" (BEC). The channel is specified by erasure probability which can be adjusted to achieve required compression rate. We can then consider turbo compression as transmitting encoded bits over BEC. If the proportion of erased bits holds the equation (2) then the codeword length is $K \cong NH(U)$, and we compressed our sequence close to entropy

$$\varepsilon \approx 1 - \frac{H(U)}{2}. \tag{2}$$

*C. Source decoding*

The compressed source is decoded by using turbo decoder, depicted in fig. 3. The decoder uses soft decision decoding that means it describes the input by using L-values (log-likelihood ratios). For source U with $u = \{-1,+1\}$ the L – value is defined as follows

$$L = \log_e \left( \frac{P(u=+1)}{P(u=-1)} \right). \tag{3}$$

Turbo decoder uses two independent component decoders with feedback loop as shown in Fig. 3.

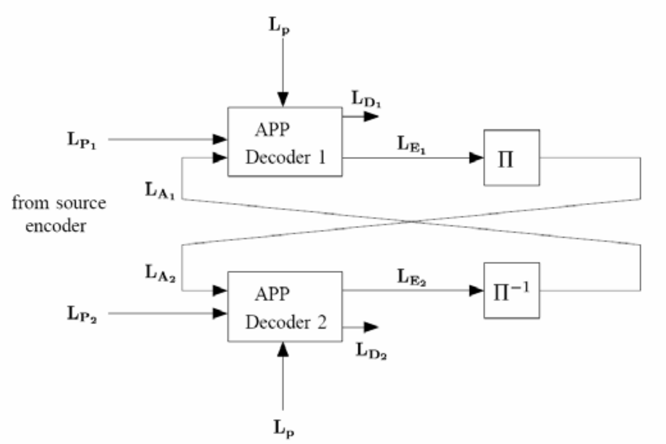


**Fig. 3.** Parallel turbo source decoder

Since the parity sequence $P_1^{K/2}, P_2^{K/2}$ are assumed to be transmitted over BEC, the input values $L_{P1}$ and $L_{P2}$ take on values $\pm\infty$ (if $u = +1$ or $u = -1$) or 0 (if u bit is erased). In the case of non-uniform binary source ($P(u = +1) = p$ and $P(u = -1) = 1 - p$) with entropy $H(U) = H_b(p)$, each decoder has additional input vector $L_P$ called "source a priori knowledge" where each element of the vector is equal to $L_P = \log_e\left(\frac{p}{1-p}\right)$.

The input values $Lp_1$ and $Lp_2$ are passed through a posteriori probability decoder (APP Decoder 1) with soft input and soft output (SISO). From decoded sequence $L_{D1}$ we calculate extrinsic information $L_{E_1} = L_{D_1} - L_{A_1} - L_P$. This information called "learnt a priori knowledge" is then interleaved and used as a priori information $L_{A1}$ for the other decoder. The same process is performed with APP Decoder 2. The algorithm is in iterative way repeated until convergence is achieved. Interleaver and deinterleaver ($\Pi, \Pi^{-1}$) introduced in the scheme remove the dependencies between bits in decoded sequences.

### *D. Decremental redundancy*

For lossless data compression the convergence of the above algorithm has to be guaranteed. It means that we can puncture only so many bits that can be corrected. This is done by testing the decodability of the compressed source. As shown in fig. 4 the decoder is puncturing the sequence step by step and after each step it verifies the integrity of reconstructed source sequence. The disadvantage of this scheme is that the decoder has to be present also during encoding process.

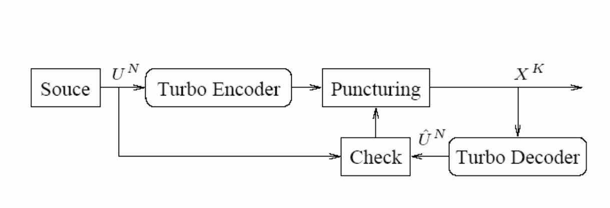


**Fig. 4.** Lossless compression of a source block

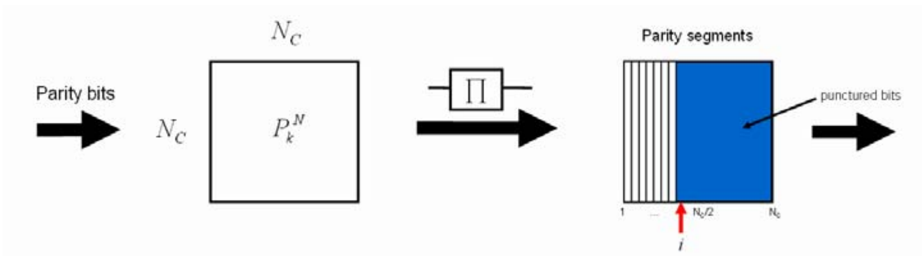


**Fig. 5.** Puncturing scheme for decremental redundancy

The principle of decremental redundancy is shown in Fig.5. As proposed in [1] each of the parity sequences $P_1^N$ and $P_2^N$ of length $N$ are first line by line stored in a matrix $N_C \times N_C = N$. Every column of the matrix is indexed by $i \in \{1,..,N_C/2,...,N_C\}$ and called parity segment. In order to spread out the erased bits in a block we interleave the parity bits before puncturing them. The redundancy of the source is removed by eliminating parity segments sequentially (parallel for $P_1^N$ and $P_2^N$ matrices) as long as the sequence is perfectly decodable. Together we have 2x$N$ parity bits, since the parity sequences $P_1^N$ and $P_2^N$ are length of $N$. To be able to achieve any compression we have to discard at least half of the parity segments.

To be able to reconstruct the data, at the decoding side the decoder has to know the block length, the random puncturing matrix, interleaver, index $i$ of the last punctured segment and a priori knowledge $L_P$, if the source is not uniformly distributed. All these information are provided except the index $i$, which has to be encoded in the compressed sequence. Assuming that half of the parity segments have to be eliminated, the maximum number of bits needed to encode the index is equal to $\lceil \log_2(N_C/2) \rceil$. The algorithm of decremental redundancy can be summarized into following steps:

1) Let $i = Nc/2$
2) Encode the source block with a turbo encoder and store the output block.
3) Puncture the encoded block using $i$ parity segments.
4) Decode the compressed block.
5) Check for errors. If the encoded block is error free, let $i = i-1$ and go back to step 3.
6) Let $i = i+1$. Repeat 3, include a binary codeword corresponding to index $i$ and stop.

A simple example of performance of decremental redundancy algorithm comparing to Lempel-Ziv coding and entropy is shown in Fig. 6. It can be seen that for the same block length (10000 bits) the turbo source

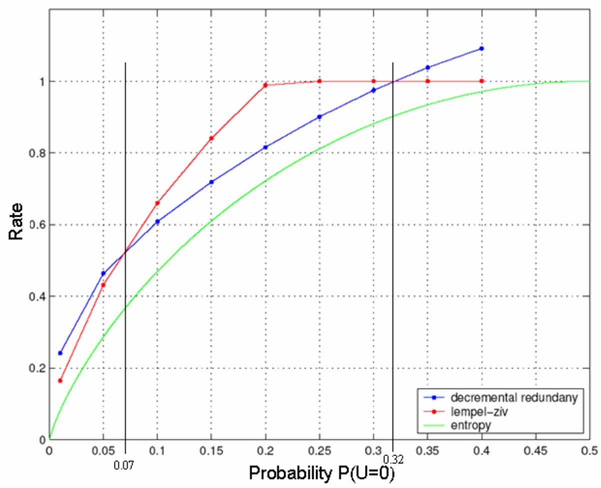


**Fig. 6.** Average compression rate for decremental redundancy algorithm compared to Lempel-Ziv and entropy.

coding algorithm achieves lower compression rates between source probabilities 0.07 and 0.32 which corresponds to source entropy ranges 0.38 and 0.9. At source probability of 0.2 the distance to entropy is halved comparing to Lempel-Ziv coding.

*E. Decremental and incremental redundancy in joint source-channel coding*

Shannon's separation theorem states that source coding (compression) and channel coding (error protection) can be performed separately and sequentially, while maintaining optimality. However, this is true only in the case of asymptotically long block lengths of data. In many practical applications, the conditions of the Shannon's separation theorem neither hold, nor can be used as a good approximation. Thus, considerable interest has developed in various schemes of joint source-channel coding. One of these schemes uses decremental and incremental redundancy algorithm mentioned above. As in [4] the Fig. 6 shows us how this algorithm is applied.

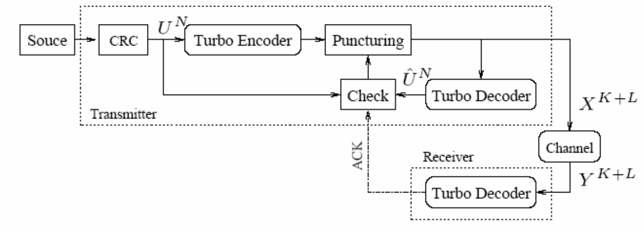


**Fig. 7.** Transmission of compressed data

The transmitter is our turbo source encoder, which compresses the source data. As integrity test a cyclic redundancy check word (CRC) is added to every transmitted block. Afterwards sequence $U^N$ (containing CRC) is encoded as described in section B.

If the channel is noisy, additional parity bits have to be added to sequence $X^K$ in order to decode the data error free (incremental redundancy). Let $L$ be the number of additional parity bits added to encoded sequence to compensate channel errors. Than the compressed data $X^{K+L}$ including the punctured source sequence, CRC and index of puncturing matrix are transmitted over a noisy channel. The receiver receives sequence $Y^{K+L}$ and checks the integrity of received data with CRC. If test is not successful, encoder increases the index of puncturing matrix and sends the new sequence again. This process is repeated until positive acknowledgement (ACK) is received.

Note, that for this kind of joint source-channel coding, we need the state channel information to simulate the channel in encoder.

*F. Algorithm analysis using EXIT charts*

One of the ways how to analyze the convergence and thus obtain the desired compression rate is to use slightly modified version of EXIT charts [1]. Exit charts use mutual information to parameterize the L-values being exchanged within the source decoder. They were first applied to analyze the turbo decoder for channel coding, observing the mutual information between channel input U and channel output Y. Let's assume binary source U with $u = \{-1,+1\}$ and $P(u=+1) = p$. If Y represents the channel output than the mutual information between random variable U and Y is

$$I(U;Y) = \sum_{u=\pm 1} p(U=u) \int_{-\infty}^{\infty} f(y\,|\,U=u) \cdot \log_2 \frac{f(y\,|\,U=u)}{p \cdot f(y\,|\,U=+1) + (p-1) \cdot f(y\,|\,U=-1)}. \tag{4}$$

For ergodic sources this can be simplified to

$$I(U;Y) = H_b(p) - E\left\{\log_2\left(1+e^{-u \cdot L(U|Y)}\right)\right\} \cong H_b(p) - \frac{1}{N}\sum_{n=1}^{N}\left(1+e^{-u_n \cdot L(u_n|y_n)}\right), \tag{5}$$

where the expectation is taken over all possible observations. This mutual information measure is now used to analyze turbo decoder by observing $I(U;L_{E_1}) = I_{E1}$, $I(U;L_{A_2}) = I_{A2}$, $I(U;L_{E_1}) = I_{E1}$, $I(U;L_{A_1}) = I_{A1}$. To construct the EXIT charts we have to characterize the component decoders of turbo decoder and determine functions $I_{E1} = f_1(I_{A1})$ , $I_{E2} = f_2(I_{A2})$. The procedure to obtain characteristic curves is described in Fig.7. From the figure we can see that $\boldsymbol{L_A}$ is generated by passing the source $U$ through an "a priori channel" assumed to be BEC. The output of APP decoder is generated based on the L-values $\boldsymbol{L_P}$ of the punctured parity bits and a priori sequence $\boldsymbol{L_A}$. From the output sequence we can determine the extrinsic information $\boldsymbol{L_E}$ and by applying equation (5) calculate the mutual information $I_E$ as well as the mutual information $I_A$. We start plotting the EXIT chart by noting that $I_{E1}$=$I_{A2}$, $I_{E2}$=$I_{A1}$ and then continue to plot $I_{E1} = f_1(I_{A1})$ and its mirrored version $I_{E2} = f_2(I_{A2})$. If there exists a tunnel between these two curves than the sequence can be successfully decompressed. As shown in Fig. 8 we can adjust the erasure probability of the puncturing scheme and see if the specified compression rate is acceptable and thus the decompression is successful.

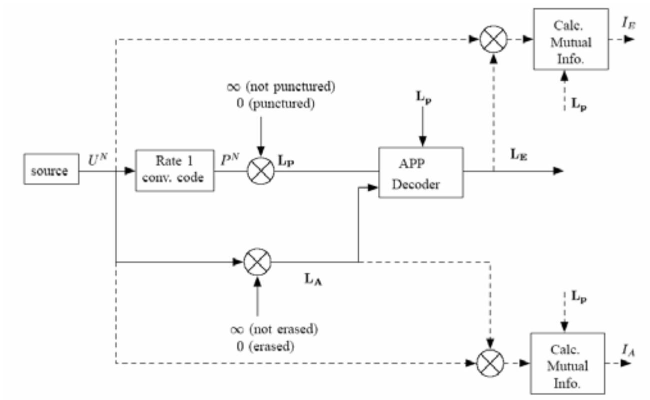


**Fig. 8.** Determining characteristic curves of component decoders

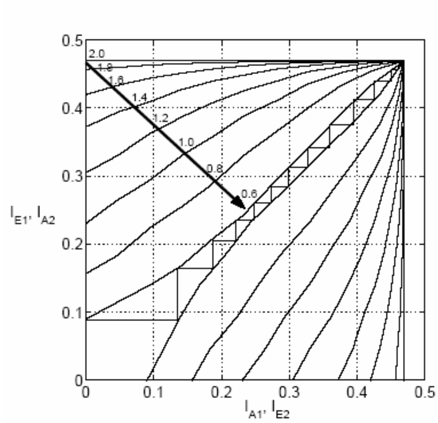


**Fig. 9.** EXIT chart for binary source H(U)=0.469, $N = 9.10^4$; parallel concatenation of two convolution codes with polynomials [7,5]

## III. CONCLUSIONS

The compression algorithm presented in this paper uses turbo coding principle. Based on the fact that source coding and channel coding are dual schemes, turbo codes can be applied in source coding scenario as shown in [1]. Lossless compression is guaranteed by using decremental redundancy algorithm which performs very close to entropy rate. This algorithm can be very efficiently used together with incremental redundancy in joint-source channel coding [4]. The convergence of turbo source decoder can be analyzed by using a modified EXIT charts. This analysis can help to optimize component codes and puncturing rates to be able to compress data close to entropy of the source.

## REFERENCES


[1] J. Hagenauer, J. Barros, and A. Schaefer, “Lossless turbo source coding with decremental redundancy,” in Proc. *International ITG Conference on Source and Channel Coding*, Erlangen, Germany, January 2004, pp. 333–339.

[2] J. Garcia-Frias and Y. Zhao, “Compression of binary memoryless sources using punctured turbo codes,” *IEEE Communications Letters*, pp. 394– 396, September 2002.

[3] J. Garcia-Frias and Y. Zhao, “Compression of correlated binary sources using turbo codes,” *IEEE Communications Letters*, pp. 417–419, October 2001.

[4] N. Dütsch and J. Hagenauer, “Combined incremental and decremental redundancy in join source-channel coding,” In *International Symposium on Information Theory and its Applications*, Parma, Italy, October 10-13, 2004.

[5] J. Hagenauer, N. Dütsch, J. Barros, A. Schaefer, “Incremental and Decremental Redundancy in Turbo Source-Channel Coding,” In *1st International Symposium on Control, Communications and Signal Processing, Hammamet, Tunisia*, March 2004, p. 595-598

[6] N. Dütsch, “Decremental Redundancy in Turbo Source Coding,” In *13th Joint Conference on Communications and Coding (JCCC 2004), Galtür, Austria*, March 2004

[7] URL: Data-Compression.com