Introducing perforation when applying turbo codes

A A Sidorenko
International Academy of Informatization, 3 Leningradsky Ave., Moscow, Russia

E-mail: kotofeykin@mail.ru

Abstract. The problem of adapting the degree of redundancy introduced in the process of error-correcting coding to the changing characteristics of the data transmission channel is urgent. Turbo codes, used in a variety of digital communication systems, are capable of correcting multiple errors occurring in the data transmission channel. The article compares the decoding efficiency for various options for introducing perforation into the code sequence generated by the turbo code encoder. Based on the comparison results, recommendations were made on the most appropriate option for the introduction of perforation.

1. Introduction
Errors occur during data transfer and storage. Due to fading, natural and artificial interference, data transmitted over the radio channel are especially vulnerable [1]. In order to prevent data loss, error correcting codes with error correction are currently widely used. The essence of error-correcting coding is to introduce redundancy into the transmitted (or stored) data. During the encoding process, the information sequence is converted into a code sequence with structural redundancy. Structural redundancy is used to identify and correct errors. The amount of redundancy determines the potential correcting ability of the code.

The introduction of redundancy reduces the real data transfer rate. An urgent problem is the problem of adapting the amount of redundancy to changes in the characteristics of the data transmission channel. One of the simplest adaptation methods is code perforation, that is, the exclusion (puncturing) of one or another number of transmitted bits from the data stream. Puncturing is used in both block and convolutional codes, and can also be used in turbo codes based on punched codes [2]. As a rule, the parameters of the code are calculated based on the worst state of the data transmission channel, and the introduction of perforation makes it possible to increase the data transmission rate in a favourable state of the channel.

2. Methods
Of all the error-correcting codes, the turbo codes came closest to the Shannon limit [3]. Turbo codes are currently used in a variety of data transmission systems. The most common options for constructing a turbo code based on a systematic convolutional code (hereinafter - SCC) [3]. As can be seen from the structural diagram of the \( N \) - dimensional turbo code encoder (figure 1), the transmitted data stream consists of a sequence of information bits and \( N \) sequences of check bits [4]. Which of the sequences is most appropriate to perforate? This publication will substantiate the answer to this question based on the consideration of the results of decoding the code sequence generated by the turbo encoder, consisting of two SCC encoders and an interleaver (figure 2). Decoding will be
performed in accordance with the principle of the maximum posterior probability of the information bit (hereinafter referred to as MAP, from the English maximum a posteriori). The decoding process in accordance with the MAP algorithm is detailed in various sources [4, 5]. The calculation of the reliability of information symbols at the output of the decoder for three types of perforation will be considered: perforation of a sequence of information bits, a sequence of check bits of the first code, a sequence of check bits of the second code. The article applies the provisions of the theory of error-correcting coding in the part concerning decoding of convolutional codes and turbo codes [3].

Figure 1. Block diagram of the \( N \)-dimensional turbo code encoder.

Figure 2. Block diagram of a turbo code encoder based on a convolutional encoder.

3. Results

Let's describe the process of creating a turbo code sequence. During one coding cycle, three code bits are formed at the output of the turbo code. The first code bit is an information bit, the second code bit is a check bit from the output of the first encoder, the third code bit is a check bit from the output of the second encoder.

Let 1100 be an information sequence. The coding process by the first encoder is schematically shown in Figure 3. In the figure, the state of the lattice nodes 00, 10, 01, 11 are designated for convenience by the letters a, b, c, d, respectively. As a result, a sequence of check bits of the first encoder is formed: 1001. The decoding process of the convolutional code is facilitated if the final state of the memory cells is known, as a rule, corresponding to their zeroing (state a). For this, a so-called
"terminating" sequence is added at the end of the information bits [6]. We will assume that when decoding the sequence formed by the first encoder, the final state is known.

Figure 3. The encoding process by the first encoder.

Suppose that during interleaving, the first bit changes with the third, and the second with the fourth, then the sequence 0011 arrives at the input of the second encoder, and a sequence of check bits 0010 is formed at its output.

Based on the results of the operation of the first and second SCC encoders, the output sequence of the turbo code encoder is formed: 110 100 001 010. Let the transmission of bits be carried out by bipolar signals, then the transmitted sequence will have the form 11-1 1-1-1 -1-11 -11-1.

The MAP decoding algorithm is most efficient when using the soft output of the detector [7]. It is obvious that the comparison of the results of decoding when puncturing the information sequence or a sequence of check bits will differ when puncturing a symbol having a value of 0.1 or 0.9 in the absence of puncture. To exclude this kind of distortion of the research results, we will use the hard decision of the detector about the value of the received symbol ("1" or "-1"), and assign the punctured bits to the value "0" at the decoder input. The block diagram of the turbo code decoder is shown in figure 4. At each time instant $k$ (except for $k=5$) the detected values of the information bit $x_k$ and check bits of the first and second encoders $y_{k1}$ and $y_{k2}$, respectively, arrive at the input of the turbo code decoder.

Figure 4. Block diagram of the turbo code decoder.
Consider the MAP decoding algorithm using formulas (1) - (6) from the source [8]. The detected values of the information bit \( x_k \) and the check bit of the first encoder \( y_{k1} \) are fed to the input of the first decoder at each time instant \( k \) (except for \( k = 5 \)). Decoding begins with calculating the metrics of the decoder trellis branches (figure 5) according to the formula:

\[
\delta^{i,m}_k = 0.5 \cdot \exp (x_k \cdot u^i_k + y_{k1} \cdot v^i_k),
\]

where the parameter \( i \) takes the value 0 or 1, depending on which bit the given branch corresponds to the arrival (for branches going straight or up from the node \( i = 0 \), for branches going down from the node \( i = 1 \)), \( u_k \) and \( v_k \) are the values of the bits generated by this branch, \( m \) is a parameter indicating the state of the lattice node (taking values: \( a, b, c, d \)).

After calculating the values of the branch metrics for all \( m \) and \( k \) (except for \( k = 5 \)), the metrics of the forward pass along the decoder lattice are calculated:

\[
a_k^m = \sum_{j=0}^1 \delta_k^{j,b(j,m)} \cdot a_k^{b(j,m)}. \tag{2}
\]

The initial state of the path along the lattice \( m = a \). Then, for \( k = 1 \) and \( m = a \), the metric of the direct pass is equal to 1, and all other metrics of the direct pass (for \( m = b, c, d \)) are equal to zero.

After calculating the forward pass metrics, the backward pass metrics are calculated along the decoder lattice from state \( k = 5 \) to state \( k = 1 \):

\[
\beta_k^m = \sum_{j=0}^1 \delta_k^{j,m} \cdot \beta_k^{f(j,m)}. \tag{3}
\]

Due to the use of the terminating sequence, encoding by the first encoder ends in the known state \( m = a \). Then, for \( k = 5 \) and \( m = a \), the back pass metric is equal to 1, and all other back pass metrics (for \( m = b, c, d \)) for \( k = 5 \) are zero.

Having the values of all three types of metrics, it is possible to calculate the logarithmic likelihood function for all information bits:

\[
L_k(d_k) = \ln \left( \frac{\sum_m a_k^m \delta_k^{i,m} \beta_{k+1}^{f(i,m)}}{\sum_m a_k^m \delta_k^{i,0} \beta_{k+1}^{f(0,m)}} \right). \tag{4}
\]

Similar calculations by formulas (1) - (4) are carried out for the second decoder. The detected and interleaved values of the information bit and the detected values of the check bit of the second encoder \( y_{k2} \) are fed to the input of the second decoder at each time instant \( k \) (except for \( k = 5 \)). When calculating the metrics of the backward pass, it is necessary to bear in mind that the final state on the lattice is unknown, therefore, for \( k = 5 \), all the metrics of the backward pass are equal to 1.

After calculating the logarithmic likelihood function for all information bits at the output of the second decoder, and interleaving calculation results, the first full decoding iteration is considered complete. It becomes possible to calculate the value of the soft output of the turbo code decoder:
where \( L_i(d_i) \) is the channel measurement of the symbol value, \( L_1(d_i) \) is the a priori value of the logarithmic likelihood function (in our case, according to the results of passing the first decoder), \( L_e(d_k) \) is the value of the logarithmic likelihood function according to the decoding results (in our case according to the results of passing the second decoder).

To achieve the required reliability of the solution, several iterations can be performed, and each subsequent iteration must be performed taking into account the logarithmic likelihood function for all information bits obtained at the previous iteration. To compare the results of decoding with different variants of the introduction of perforation, we restrict ourselves to the results of one full iteration. Ultimately, a tough decision is made about the value according to the following criteria:

\[
\begin{align*}
    &d_k = 1, \text{if } L(d_k) > 0 \\
    &d_k = 0, \text{if } L(d_k) < 0
\end{align*}
\]  

Based on the results of considering the decoding process, an assumption can be made regarding the most appropriate perforation of a particular sequence, based on the following reasoning. Information bits are involved in the decoding process in both decoders, and their perforation will negatively affect the reliability of the decoding results of both decoders. As mentioned earlier, due to the use of the termination sequence, the encoding by the first encoder ends in a known state, and therefore the decoding of this sequence gives more reliable results than the sequence generated by the second encoder. Hence follows the assumption about the greatest expediency of perforation of the sequence formed from the check bits of the second encoder (as the least valuable).

Consider decoding the sequence 11-1 1-1-1 -1-11 11-1 detected without errors for three options for the introduction of perforation, defined earlier.

When all information bits are punctured, the sequence at the input of the turbo code decoder is 01-1 0-1-1 0-11 01-1. Then the input of the first decoder receives the sequence 01 0-1 0-1 01, and the input of the second decoder 0-1 0-1 01 0-1.

Let's calculate the output of the first decoder. The results of calculating the metrics of the branches according to the formula (1) are shown in table 1. The results of calculating the metrics of the forward pass according to the formula (2) are shown in table 2. The results of calculating the metrics of the reverse pass according to the formula (3) are shown in table 3.

\[
L(d_k) = L_e(d_k) + L'_e(d_k) + L_e(d_k),
\]  

\[
(5)
\]

| Table 1. Values of branch metrics. |   |   |   |   |
|-----------------------------------|---|---|---|---|
| \( k \)                           | 0 | 1 | 0 | 1 |
| \( m=a \)                         | 0.184 | 1.359 | 1.359 | 0.184 | 1.359 | 0.184 | 1.359 |
| \( m=b \)                         | 1.359 | 0.184 | 0.184 | 1.359 | 0.184 | 1.359 | 0.184 |
| \( m=c \)                         | 1.359 | 0.184 | 0.184 | 1.359 | 0.184 | 1.359 | 0.184 |
| \( m=d \)                         | 0.184 | 1.359 | 1.359 | 0.184 | 1.359 | 0.184 | 1.359 |

| Table 2. Values of forward pass metrics. |
|------------------------------------------|
| \( m=a \) | 1 0.184 0.25 0.386 |
| \( m=b \) | 0 1.359 0.034 0.386 |
| \( m=c \) | 0 0 0.25 2.517 |
| \( m=d \) | 0 0 1.847 0.386 |

| Table 3. Values of back pass metrics. |
|---------------------------------------|
| \( m=a \) | 0.386 0.25 0.184 1 |
| \( m=b \) | 2.517 0.25 0 0 |
As a result of the operation of the first decoder using formula (4), the following results of calculating the logarithmic likelihood function were obtained: $L_c(d_1) = 3.875$; $L_c(d_2) = 3.875$; $L_c(d_3) = -\infty$; $L_c(d_4) = -\infty$. The output of the first decoder is generated.

The output of the second decoder is calculated similarly, taking into account the above notes. As a result of the operation of the second decoder, the following results of calculating the logarithmic likelihood function were obtained: $L(d_1) = 1.325$; $L(d_2) = 0$; $L(d_3) = 0$; $L(d_4) = 0$.

Calculation results by formula (5) of the soft output of the turbo code decoder based on the results of one complete iteration are shown in table 4. The hard output of the turbo code in accordance with criterion (6) does not contain errors.

**Table 4. Results of calculating the soft output of the turbo code decoder.**

| $k$ | $L_c(d_k)$ | $L_e'(d_k)$ | $L_e(d_k)$ | $L(d_k)$ | $d_k$ |
|-----|------------|-------------|------------|----------|-------|
| 1   | 0          | 3.876       | 1.325      | 5.201    | 1     |
| 2   | 0          | 3.876       | 0          | 3.876    | 1     |
| 3   | -\infty   | 0           | -\infty    | 0        | 0     |
| 4   | -\infty   | 0           | -\infty    | 0        | 0     |

When perforating the check bits of the first encoder, the input sequence of the turbo decoder looks like this: 10-1 10-1 -101 -10-1. Then the input of the first decoder receives the sequence 10 10 -10 -10, and the input of the second decoder is -1-1 -1-1 11 1-1. The results of calculating the soft output of the turbo code decoder are shown in table 5. The hard output of the turbo code in accordance with criterion (6) does not contain errors.

**Table 5. Results of calculating the soft output of the turbo code decoder.**

| $k$ | $L_c(d_k)$ | $L_e'(d_k)$ | $L_e(d_k)$ | $L(d_k)$ | $d_k$ |
|-----|------------|-------------|------------|----------|-------|
| 1   | 1          | 2           | 4.897      | 7.897    | 1     |
| 2   | 1          | 2           | 3.843      | 6.843    | 1     |
| 3   | -1         | -\infty    | -6.794     | -\infty  | 0     |
| 4   | -1         | -\infty    | -5.567     | -\infty  | 0     |

With perforation of the check bits of the second encoder, the input sequence of the turbo decoder looks like this: 110 1-10 -1-10 -110. Then the input of the first decoder receives the sequence 11 1-1 -1-1 -11, and the input of the second decoder -10 -10 10 10. The results of calculating the soft output of the turbo code decoder are shown in table 6. Hard output of the turbo code in accordance with the criterion (6) contains no errors.

**Table 6. Results of calculating the soft output of the turbo code decoder.**

| $k$ | $L_c(d_k)$ | $L_e'(d_k)$ | $L_e(d_k)$ | $L(d_k)$ | $d_k$ |
|-----|------------|-------------|------------|----------|-------|
| 1   | 1          | 7.307       | 2          | 10.307   | 1     |
| 2   | 1          | 7.307       | 2          | 10.307   | 1     |
| 3   | -1         | -\infty    | -2         | -\infty  | 0     |
| 4   | -1         | -\infty    | -2         | -\infty  | 0     |

Comparative results of calculating the soft output of the turbo code decoder for different types of perforation are shown in table 7. The most reliable are the results of decoding the sequence with perforation of the check bits of the second encoder. The least confidence is the results of decoding a sequence with perforated information bits.
Table 7. Output values of the turbo code decoder for all perforation options.

| Bit no. | Perforation types | Transmitted bit value |
|---------|-------------------|-----------------------|
|         | information bit perforation | first encoder check bit perforation | second encoder check bit perforation |
| 1       | 5.201             | 7.897                 | 10.307                 | 1       |
| 2       | 3.876             | 6.843                 | 10.307                 | 1       |
| 3       | -∞                | -∞                    | -∞                    | 0       |
| 4       | -∞                | -∞                    | -∞                    | 0       |

Let us consider the results of decoding the received perforated sequence in the presence of erroneous bits in it for the previously indicated three options for introducing punctures. Let each of the sequences (which make up the output of the turbo code encoder) not subjected to perforation contain one erroneous bit.

When all information bits are perforated and there are errors in each of the check bit sequences, the sequence at the input of the turbo decoder has the form 011 0-1-1 0-11 0-1-1, where the erroneous bits are underlined. Then the input of the first decoder receives the sequence 01 0-1 0-1 -1, and the input of the second decoder 01 0-1 01 0-1.

When perforating the check bits of the first encoder and the presence of errors in the information sequence and the sequence of check bits of the second encoder, the sequence at the input of the turbo code decoder has the form 101 10-1 101 -10-1, where the erroneous bits are underlined. Then the input of the first decoder receives the sequence 10 10 10 -10, and the input of the second decoder 11 1-1-1 11 1-1.

When the check bits of the second encoder are punctured and there are errors in the information sequence and the check bit sequence of the first encoder, the sequence at the input of the turbo decoder has the form 110 1-10 1-10 -10-1-10, where the erroneous bits are underlined. Then the input of the first decoder receives the sequence 11 1-1 1-1 -1-1-1, and the input of the second decoder is 10-10 10 10.

Comparative results of calculating the soft output of the turbo decoder for different types of perforation in the presence of errors in non-perforated sequences are shown in table 8. As in cases where there are no errors, the comparison results confirm the hypothesis that it is most expedient to introduce perforation into the sequence of check bits of the second encoder.

Table 8. Output values of the turbo code decoder for all perforation options.

| Bit no. | Perforation types | Transmitted bit value |
|---------|-------------------|-----------------------|
|         | information bit perforation | first encoder check bit perforation | second encoder check bit perforation |
| 1       | 1.449             | 5.273                 | 7                         | 1       |
| 2       | -1.073            | 4.023                 | 6.3                       | 1       |
| 3       | -∞                | -∞                    | -∞                       | 0       |
| 4       | -∞                | -∞                    | -∞                       | 0       |

4. Conclusions

The simplest method for adapting the correcting ability of a turbo code is perforation. The data stream at the output of the turbo encoder is a mixture of a sequence of information bits and several sequences of check bits. Comparison of the results of decoding when perforating different sequences allows us to conclude that it is most expedient to introduce perforation into a sequence of check bits, the end of which is not protected by a terminating sequence. There is no need to make significant changes to the MAP algorithm to work with punched code sequences.
References

[1] Kharkevich A A 2018 *Anti-interference* (Moscow: Lenand)
[2] Sidorenko Alexander A 2019 *J. Phys.: Conf. Ser.* 1399 033011
[3] Sklar Bernard 2001 *Digital Communications Upper Saddle River* (NJ: Prentice Hall)
[4] Robert H. Morelos-Zaragoza 2006 *The Art of Error Correcting Coding* 2d edition (John Wiley & Sons)
[5] Yeung R W 2008 *Information Theory and Network Coding* (Springer)
[6] Richardson T and Urbanke R 2008 *Modern Coding Theory* (Cambridge University Press)
[7] Moon T K 2005 *Error Correction Coding Mathematical Methods and Algorithms* (Jhon Wiley and Sons)
[8] Kudryashev B D 2016 *Foundations of Coding Theory* : a tutorial (SPb .: BHV-Petersburg)