Fifteenth International Workshop on Algebraic and Combinatorial Coding Theory
June 18-24, 2016, Albena, Bulgaria *pp. 307–312*

A binary block concatenated code based on two convolutional codes [1]

Igor Zhilin zhilin@iitp.ru
IITP RAS
Victor Zyablov zyablov@iitp.ru
IITP RAS
Dmitry Zigangirov zig@iitp.ru
IITP RAS

Abstract. We propose a concatenated code construction based on convolutional codes. We prove that minimum distance of this construction equals product of free distances of component codes.

1 Introduction

Using known codes to create a concatenated construction is an old known way of creating a new code that usually has higher length and performs closer to Shannon limit. Several ways of creating such codes are known, such as product codes, concatenated codes, generalized concatenated and generalized error-locating codes, (partial) unit memory codes, turbo codes and so on. On of concatenation methods that is used in ITU-T Recommendation G.975.1 [2] for LDPC super FEC code consists of using just two codes and encoding an information vector by outer (n_B, k_B) code and then encoding this result with inner (n_A, k_A) code, where $k_A = n_B$. In other words, is consists of applying two codes sequentially, possibly with symbol permutation in between.

We consider similar coding scheme but with convolutional component codes. We propose a lower bound of code distance for this code.

It is worth noting that this construction differs from product convolutional codes [4]: in [4] authors made *convolutional* code based on two convolutional codes. We consider a *block* code that uses *terminated* convolutional codes as component codes and derive its block minimal distance.

2 Construction Description and Encoding

Let us describe construction in general. We consider this construction with two component codes that can be the same code.

[1]This work has been supported by RScF, research project No. 14-50-00150.

Let us write information symbols of this code as matrix:

$$\mathbf{I} = \underbrace{\left.\begin{array}{|c|}\hline\\\hline\end{array}\right\}k_A}_{k_B} \tag{1}$$

Encoding of the code is done as follows. At first we encode information matrix in row-wise order by outer code:

$$\mathbf{I}_A = Enc_B(\mathbf{I}) = \underbrace{\left.\begin{array}{|c|}\hline\\\hline\end{array}\right\}k_A}_{n_B} \tag{2}$$

The result is stored in a matrix in row-wise order. Information sequence is terminated in a usual way. White cells represent information symbols and grey cells represent parity-check symbols. It is worth noting that all this matrices are processed row-by-row by a single convolutional encoder. This differs, for example, from the work [5] where authors considered encoding all rows by several independent encoders.

Then we encode this matrix in column-wise order by inner code:

$$\mathbf{C} = Enc_A(\mathbf{I}_A) = \underbrace{\left.\begin{array}{|c|}\hline\\\hline\end{array}\right\}n_A}_{n_B} \tag{3}$$

The result is also stored in a new matrix in column-wise order, this matrix is a codeword of this construction.

3 A bound on code distance

Let us introduce some designations:

$R_A = b_A/c_A$ — rate of inner code,

d_A — free distance of inner code in binary symbols,

f_A — maximum length of word (information sequence) of inner code that has weight d_A in blocks of code,

R_B, b_B, c_B, d_B, f_B — the same for outer code.

Let us consider code construction where $n_A \geq f_A c_A$, $n_B \geq f_B c_B$. That means that the longest word of minimal weight of outer/inner code fits in a single row/column (probably with wrapping).

At first let us consider a naive case. Suppose that information matrix $\mathbf{I}$ generates a word of minimal weight in a first row of $\mathbf{I}_A$. That means that the first row of $\mathbf{I}_A$ stores a word of weight d_B, all other rows equal zero. Since $n_B \geq f_B c_B$ this codeword occupies only the fist row.

Now we can consider encoding of the inner code. If we consider convolutional code trellis, each time encoder touches zeroth state and goes trough several subsequent zeroth state, it starts new independent codewors.

Any non-zero symbol at the input of the outer code can not yield code sequence of weight less than d_A and a sequence of such weight cant be longer than f_A symbols. At first we can suppose that columns are high enougth that their encoding is done independently. Then code sequences of the inner code that are generated by 1s in the first row will be placed in different columns without overlapping. There are at least d_B such sequences and each sequence has weight of at least d_A, which gives us lower bound on the weight of such codeword: $d \geq d_A d_B$.

Such code combination always exists. That means that this bound is strict, $d = d_A d_B$.

This naive description does not account for dependency between subsequent encoded columns. Let us now describe bound that accounts for these dependencies.

Theorem 1. *Exist such sizes n_A and n_B that Hamming distance of binary block concatenated code based on two convolutional codes is $d = d_A d_B$, where d_A and d_B are free distances of inner and outer codes respectively.*

Proof. Encoding of the outer code is anyway just a plain encoding of the convolutional code. The first encoder takes information matrix $\mathbf{I}$ and encodes it in row-wise order as shown in Eq. 1.

Any non-zero information matrix gives a matrix $\mathbf{I}_A$ of weight greater or equal to free distance of outer code d_B. We have to chose $n_B \geq f_B c_B$ to guarantee that if the word has minimal weight then it fully fits in a single row (up to an arbitrary shift that would cause wrapping). That means that each non-zero element of $\mathbf{I}_A$ is placed in different column. Look of such arrangement is shown in Eq. 2 by colored matrix elements.

Now we need to consider encoding of outer codes. We have n_B columns, at least d_B of them contain non-zero symbols. Encoding by the outer code is also done sequentially but in column-wise order. We need to consider two options: when the encoder goes through two consequent zero states while encoding consequent columns or not.

- At first let us consider an option when decoder goes through two consequent zero states in between encoding of consequent columns. That is a simpler case since we can immediately say that encoding of the first and of the second symbol of outer code codeword (or, equally, column of $\mathbf{I}_A$) is independent. That means that each such symbol generates a sequence

of weight of at least free distance of inner code d_A. In order achieve this we need to choose $n_A \geq f_A c_A + 1$ where addition of 1 is needed because of possible wrapping.

- The second option is that the outer code generates such matrix $\mathbf{I}_A$ that decoder of the inner code doesn't go through two consequent zero states. That means that these two parts are dependent and we cannot use free distance to estimate weight of the resulting codeword. To estimate it we need active distances [3]. Active distances lower bound weight of a code sequence generated by a coder that does not pass through two consequent zero states. Authors proved that convolutional codes with active distances that grow with sequence length and lower-bounded by a linearly increasing function exist and also showed a couple of examples of known codes where increasing active distances is seen, see fig. 1.

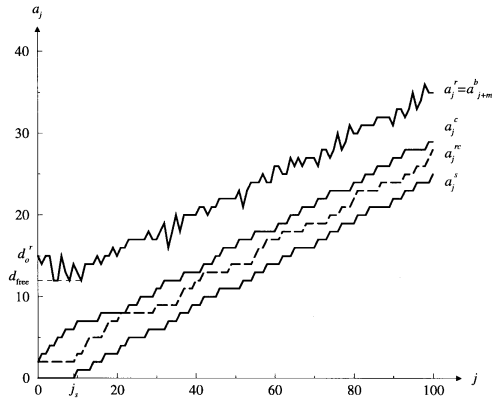

Figure 1: Example of active distances from [3].

Since we need two consequent columns to have weight of at least $2d_A$, three columns to have weight $3d_A$ and so on, we need to choose such n_A that row active distance

$$a_j^r \geq sd_A, s \in \overline{1, n_B}, \tag{4}$$

where $j = sn_A/c_A$. That can be easily done since a_j^r can be lower-bounded by linearly increasing function. Suppose that

$$a_j^r \geq uj + v \tag{5}$$

Then we need

$$uj + v = u(sn_A/c_A) + v = usn_A/c_A + v \geq sd_A \tag{6}$$

Asymptotically that yields $un_A/c_A \geq d_A$ or, equivalently,

$$n_A \geq d_A c_A / u \tag{7}$$

It can also be evaluated for all $s \in \overline{1, n_B}$ and maximum of n_A from each inequality needs to be chosen.

That means that if we choose such n_A, then average weight of each non-zero column will be more than or equal to d_A. Average here is used in sense that if we have s consequent dependent columns, we does not guarantee that $wt(each\ column) \geq d_A$, but guarantee that $wt(all\ columns) \geq sd_A$.

That gives us lower bound on the code distance, $d \geq d_A d_B$.

Example before the theorem proves that exist a codeword of weight strictly $d_A d_B$. Therefore minimal distance of this construction with appropriate n_A and n_B is $d = d_A d_B$. □

It is worth noting that this theorem is only valid for convolutional component codes. Since we do not know local properties of minimal-weight words of block code, we do not know what code distance of similar construction based on block codes would be.

4 Conclusion

We proposed a lower bound on code distance for binary block concatenated code based on two convolutional codes. This construction differs from other constructions of concatenated codes based on convolutional codes from one side in sense that it is a block code (unlike convolutional code in [4]) and uses one convolutional code for sequentially encoding all rows and one convolutional code for sequentially encoding all columns (unlike code in [5] that uses separate convolutional codes for all rows and columns).

References

[1] E. L. Blokh, V. V. Zyablov, "Generalized concatenated codes," *Svyaz', Moscow (1976).*

[2] "Forward Error Correction for High Bit Rate DWDM Submarine Systems", *ITU-T recommendation G.975.1, Feb. 2004.*

[3] S. Host, R. Johannesson, K. Sh. Zigangirov, V. V. Zyablov, "Active Distances for Convolutional Codes," *IEEE Transactions on Information Theory, Vol. 45, No. 2, March 1999.*

[4] M. Bossert, C. Medina, V. Sidorenko, "Encoding and distance estimation of product convolutional codes," *Proceedings. International Symposium on Information Theory, 2005. ISIT 2005., Adelaide, SA, 2005, pp. 1063-1067.*

[5] O. Gazi, A. O. Yilmaz, "Turbo Product Codes Based on Convolutional Codes," *ETRI Journal, Volume 28, Number 4, August 2006*