A New Upper Bound on the Minimum Distance of Turbo Codes

Alberto Perotti, *Member, IEEE*, and Sergio Benedetto, *Fellow, IEEE*

***Abstract*— In this paper, a new upper bound on the minimum distance of turbo codes is derived. The new bound is obtained by construction of an undirected graph which reflects the characteristics of the constituent codes and the interleaver. The resulting expression shows that the minimum distance of a turbo code grows approximately with the base-3 logarithm of the information word length. The new bound is easy to compute, applies to rate k_0/n_0 constituent encoders, and often improves over existing results.**

***Index Terms*— Minimum distance, performance bounds, permutation graphs, turbo codes.**

I. Introduction

Turbo-like codes, in their parallel [1] and serial [2] versions, are widely known to be very powerful error correcting codes. Their amazing performance are based on a code construction that privileges very low error event multiplicities, rather than large distances. For this reason, they tend to exhibit an "error floor" that limits their coding gain at very low bit error probabilities [3].

The minimum distance of a turbo-like code is largely determined by the interleaver. In order to avoid the occurrence of low-weight codewords, several methods have been proposed for the design of permutations [4] [5], and some of them are based on graph theory [6]. Moreover, as shown in [7] and [8], the logarithmic dependence on the information word length of the minimum distance is due to the presence of the interleaver.

While average bounds on the maximum-likelihood performance of turbo-like schemes are known [2], [9], based on the concept of the so-called "uniform interleaver", the evaluation of the minimum distance of a scheme using an actual interleaver is a computationally intensive task.

Upper bounds on the minimum distance of parallel concatenated scheme based on known constituent codes as functions of the interleaver size have been proposed in [7], [10]–[12]. Among these, the bounds in [7] and [13] are based on graph theory, and both result in a logarithmic expression. In [7], the time required to evaluate the bound is $O(K^2)$, where K is the interleaver length; therefore, it becomes impractical for large values of K. For this reason, in the same article a simpler bound is proposed whose complexity is independent from K. In [13], a similar result is obtained which confirms the logarithmic growth of the minimum distance in a two-branch turbo code to be due to the presence of the interleaver.

In this paper, a new upper bound on the minimum distance of parallel concatenated convolutional codes with given information word length and constituent encoder memory is derived. This result is based on the construction of an undirected graph whose characteristics are determined by the constituent encoders and interleaver. The graph construction is similar to that proposed in [7] and [13]. However, our approach discards the artificial index partition introduced by both papers, and this permits to determine the length of the error events participating in the generation of a codeword more accurately, and hence to derive a tighter upper bound. Thanks to these modifications, the new bound significantly improves upon the results presented in [7] and [13] for several example turbo codes.

Manuscript received ....

A. Perotti and S. Benedetto are with CERCOM (Center for Multimedia Radio Communications) - Politecnico di Torino, I-10129 Torino, Italy (e-mail: {alberto.perotti, sergio.benedetto}@polito.it)

A correspondence between the set of cycles in the graph and a subset of the turbo codewords is established, and the minimum distance is estimated through the length of the shortest cycle. It is shown that the minimum distance of turbo codes grows approximately with the base-3 logarithm of the information word length. The new bound is fairly simple to compute, and proves to be tighter than the previously known results [7], [10], [11], and [13] for a wide range of parameter values. Moreover, specific results corresponding to different variants of turbo codes are provided: either tail-biting and independently terminated codes are considered. Results for turbo codes with k_0/n_0 ($k_0 > 1$) constituent encoders and turbo codes with symbol interleaving are provided as well.

In Sec. II, the reference coding scheme is described. Sec. III introduces the basic notations of graph theory and defines some basic properties of graphs. Sec. IV describes the construction of the extended permutation graphs we will use to derive the bound. Sec. V describes the derivation of the upper bound. Sec. VI shows how the extended permutation graph is modified to take into account the independent termination of the constituent codes. In Sec. VII, the obtained results are compared with the previously known results. Finally, in Sec. VIII high-rate turbo codes are considered: both $1/n_0$ constituent codes with puncturing and the use of high-rate constituent codes are analyzed.

II. Reference coding scheme

The main characteristics of the code, such as the information word length K, the memory of the constituent encoders ν and the permutation performed by the interleaver, determine the minimum distance of the concatenated code.

In this paper, we will refer to a coding scheme (see Fig. 1) where two equal recursive, convolutional systematic encoders (see Fig. 2 for an example of a 4-state encoder) are concatenated in parallel through an interleaver. The information bit sequence $\mathbf{u} = \{u_0, \ldots, u_{K-1}\}$ is sent to the first encoder, which generates the systematic bit sequence $\mathbf{u}$ and the coded bit sequence $\mathbf{c}_1$. The sequence $\mathbf{u}$ is interleaved according to a permutation $\mathbf{\Pi} : i \to \pi(i) = \pi_i$, $i = 0, \ldots, K-1$. The interleaved sequence $\tilde{\mathbf{u}}$ is sent to the second constituent encoder, which generates the coded bit sequence $\mathbf{c}_2$. The output of each constituent encoder is punctured to obtain the desired coding rate.

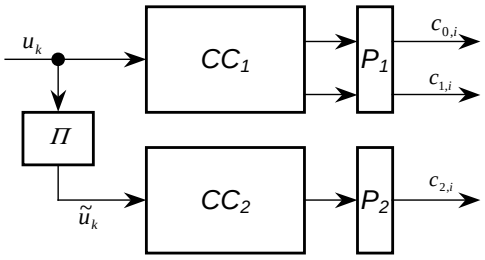


Fig. 1. Reference coding scheme.

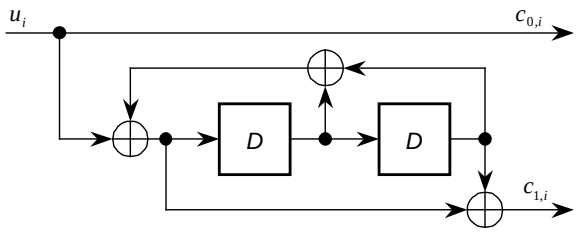


Fig. 2. A 4-state systematic, recursive convolutional encoder.

Following the characterization carried out in [7], the constituent encoders are described by the two parameters α and β, which express the Hamming weight of the constituent code sequence $\mathbf{c}_i$ when the information sequence consists of two 1 separated by $kp-1$ zeroes. Here, $p = 2^\nu - 1$ is the *encoder period*. This sequence causes a constituent encoder to leave the all-zero state, and return to it after kp trellis steps:

$$\mathbf{u} = D^i + D^{i+kp}$$

The resulting output Hamming weight is:

$$w_H(\mathbf{c}) \leq \alpha k + \beta \leq kp + 1 \tag{1}$$

where α and β depend on the encoder connections and puncturing pattern.

Here, we will consider periodic puncturing patterns. The systematic bit sequence is not punctured, while the code sequences are punctured to obtain the desired rate. In general, if $\mathbf{p}_i$ is the puncturing pattern applied to the output of CC_i, for a turbo code of rate $q/(q+1)$ we have:

$$\mathbf{p}_1 = 1 + D^{2q} + D^{4q} + \dots \tag{2}$$
$$\mathbf{p}_2 = D^q + D^{3q} + D^{5q} + \dots \tag{3}$$

where a 1 in $\mathbf{p}_i$ means that the corresponding bit in $\mathbf{c}_i$ is selected, otherwise it is eliminated.

Table I (taken from [10] and here extended) shows a set of example parameter values that will be used to compare our results with [7]. The encoder connections are in octal form. The numerator corresponds to the feedforward connections and the denominator corresponds to the feedback connections.

TABLE I
PARAMETERS α, β OF THE CONSIDERED ENCODERS.

ν	**Connections**	p	R			
			$^1/_3$	$^1/_2$	$^2/_3$	$^7/_8$
2	$5_8/7_8$	3	2, 2	1, 2	-	-
3	$17_8/13_8$	7	4, 2	2, 2	1, 2	-
4	$35_8/23_8$	15	8, 2	4, 2	2, 2	1, 2

Given the set of information words $\mathcal{U}$, and the set of permutations of length K, Π_K, we perform the derivation in the following way:

$$d_m \leq \max_{\Pi_K} \min_{\mathbf{u}\in\mathcal{U}'} w_H(CC_1(\mathbf{u})) + w_H(CC_2(\tilde{\mathbf{u}})) \tag{4}$$

where $\mathcal{U}' \subseteq \mathcal{U}$, and d_m is the minimum distance of the turbo code. When tail-biting codes are considered, $\mathcal{U}'$ contains information words with even Hamming weight. When the constituent codes are independently terminated, the content of $\mathcal{U}'$ will be defined in Sec. V.

III. GRAPH NOTATIONS

A graph G is defined by its set of vertices V and set of edges E. An edge can be considered as the association between two vertices $v_i \in V$ and $v_j \in V$, and is usually indicated using the following notation:

$$e = (v_i, v_j)$$

We will consider here only *undirected* edges.

The *degree* of a vertex $v \in V$ is the number of edges incident in v, and is indicated with $d(v)$.

A graph G is called *r-regular* when its vertices have all degree r:

$$\forall v \in V : d(v) = r.$$

3-regular graphs are called *cubic*.

A *cycle* is a subgraph $C = (V', E')$ of $G = (V, E)$, where $V' \subseteq V$ and $E' \subseteq E$, and the following properties are satisfied: (1) every vertex in V' has 2 incident edges, and (2) C is connected, i.e., a path exists between any two vertices of C. The *length* of a cycle C is defined as:

$$l(C) = |V'| = |E'|.$$

The *girth* of G is the length of its shortest cycle:

$$g(G) = \min_{C\in\mathcal{C}}[l(C)] \tag{5}$$

where $\mathcal{C}$ is the collection of all the cycles of G.

IV. EXTENDED PERMUTATION GRAPHS

The construction of a joint model including the characteristics of the constituent encoders and the interleaver permits to analyze the concatenated encoder as a whole. One way to construct such a model is to consider the permutation graph [14] associated to the interleaver and modify it according to the constituent encoders characteristics.

Initially, we will consider *tail-biting* [15] as the trellis termination technique, which consists of forcing each constituent encoder to start and end the block encoding in the same state. Therefore, every trellis path starting and ending in the same state belongs to the code. The trellis can be considered *circular* and this results in a highly regular extended permutation graph. We will then extend the construction to the case of independently terminated constituent encoders, which is the most widely adopted termination technique.

The extended permutation graph $G_{TB} = (V, E)$ is defined on the following set of vertices:

$$V = V_1 \cup V_2$$

where

$$V_1 = \{v_{I,j}, j = 0, \dots, K-1\}$$
$$V_2 = \{v_{O,j}, j = 0, \dots, K-1\}$$

V_1 (resp. V_2) corresponds to the K bits at the interleaver input (resp. output). The set of edges E is defined as follows:

$$E = E_1 \cup E_\Pi \cup E_2$$

where

$$E_1 = \{(v_{I,j}, v_{I,((j+1) \mod K)}),\ j = 0, \dots, K-1\}$$
$$E_\Pi = \{(v_{I,j}, v_{O,\pi_j}),\ j = 0, \dots, K-1\}$$
$$E_2 = \{(v_{O,j}, v_{O,((j+1) \mod K)}),\ j = 0, \dots, K-1\}$$

where E_1 (resp. E_2) contains the edges corresponding to CC_1 error events (resp. CC_2 error events), and E_Π contains the edges corresponding to the interleaver.

In Fig. 3 an example of extended permutation graph for memory 1 constituent encoders is shown.
The resulting graph is *cubic*.

Two key considerations allow to exploit the characteristics of the extended permutation graph for the derivation of our result:

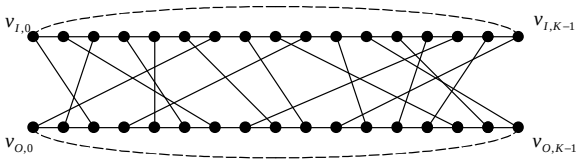


Fig. 3. An example of permutation graph for memory $\nu = 1$ constituent encoders.

1) If the constituent encoders are accumulators, i.e., 2-state recursive convolutional encoders, almost all the cycles in the permutation graph can be associated to codewords[1].
2) The length of a cycle is related to the Hamming weight of the associated codeword.

In fact, if $C \subseteq G_{TB}$ is a cycle, we can associate to it an information word $\mathbf{u}$ in the following way:

$$u_k = \begin{cases} 1 & \Leftrightarrow \quad (v_{I,k}, v_{O,\pi_k}) \in C \\ 0 & \Leftrightarrow \quad (v_{I,k}, v_{O,\pi_k}) \notin C \end{cases} \tag{6}$$

i.e., every edge in $C \cap E_\Pi$ corresponds to a 1 in the codeword. Clearly, since $|C \cap E_\Pi|$ is even[2], only information words with *even* Hamming weight are considered. This defines the set $\mathcal{U}'$ of (4).

About the second consideration, the following statements can be formulated:

1) Every vertex $v \in C$ can be associated to a non-zero trellis transition in the corresponding constituent encoder (see Fig. 4).
2) Every path consisting of k consecutive edges in $C \bigcap E_1$ (resp. $C \bigcap E_2$) corresponds to an error event of length $1 + kp$ trellis steps in CC_1 (resp. CC_2).

The two considerations lead to slightly different expressions for the upper bound. However, their values are very close to each other. In the following, we will consider the second statement, since it allows an easier extension to the case $\nu > 1$.

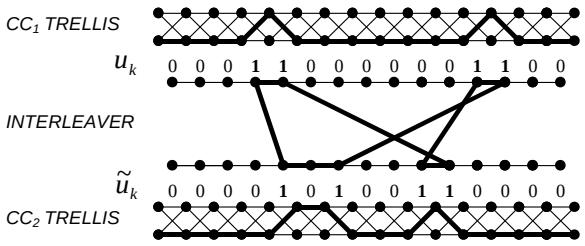


Fig. 4. Correspondence between codewords and cycles.

A. Extension to memory ν constituent encoders

When constituent encoders with memory $\nu > 1$ are considered, the permutation graph must be modified to reflect their properties.

Consider an information word $\mathbf{u}$. If $u_i = 1$, the first constituent encoder leaves the all-zero state at step i. If $u_{i+kp} = 1$, $k \in \mathbb{Z}^+$, the constituent encoder returns to the all-zero state, causing the error event to terminate. This behavior is modelled by connecting vertices $v_{I,i}$ (resp. $v_{O,i}$) and $v_{I,((i+p) \mod K)}$ (resp. $v_{O,((i+p) \mod K)}$) with an edge. The extended permutation graph G_{TB} for period p constituent encoders is defined by its set of edges

$$E = E_1 \cup E_\Pi \cup E_2$$

where

$$\begin{aligned} E_1 = &\quad \{(v_{I,i}, v_{I,(i+p) \mod K}),\ i = 0, \ldots, K-1\} \\ E_\Pi = &\quad \{(v_{I,i}, v_{O,\pi_i}),\ i = 0, \ldots, K-1\} \\ E_2 = &\quad \{(v_{O,i}, v_{O,(i+p) \mod K}),\ i = 0, \ldots, K-1\} \end{aligned}$$

The extended permutation graph for memory $\nu = 3$ constituent encoders is shown in Fig. 5. This graph is cubic, and its girth can be easily bounded from above.

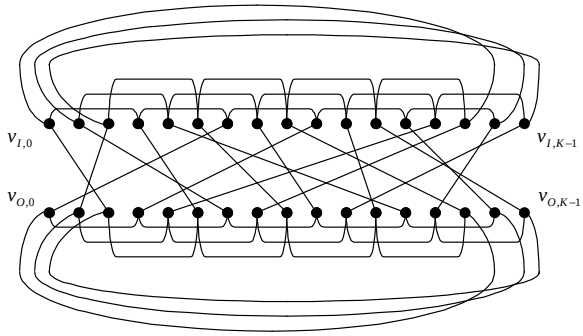


Fig. 5. Extended permutation graph for tail-biting terminated turbo codes.

Similarly to the case $\nu = 1$, cycles correspond to codewords, and their length can be used to estimate the weight of the associated codeword. However, before performing the estimation, an improvement can be carried out: given a cycle C, the edges in $C \cap E_\Pi$ do not correspond to error events. The contribution of these edges to the cycle length would result in an overestimation of the codeword weight, and hence in a looser upper bound. Therefore, we perform a further modification on G_{TB}: a new graph G'_{TB} is obtained from G_{TB} by contracting the edges in E_Π. As shown in Fig. 6, the contraction of an edge incident in two degree-3 vertices results in a new degree-4 vertex.

The resulting graph $G'_{TB} = (V', E')$ is 4-regular. Its number of vertices is $|V'| = K$, and its number of edges is $|E'| = 2K$. The set of edges $E' = E'_1 \cup E'_2$ is defined as

$$\begin{aligned} E'_1 = &\quad \{(v_{I,\pi_i}, v_{I,\pi_{(i+p) \mod K}}),\ i = 0, \ldots, K-1\} \\ E'_2 = &\quad \{(v_{O,i}, v_{O,(i+p) \mod K}),\ i = 0, \ldots, K-1\} \end{aligned}$$

The main advantage we obtain is that *all* the edges in G'_{TB} correspond to error events. Therefore, *all* the edges in *any* cycle $C \subset G'_{TB}$ correspond to error events.

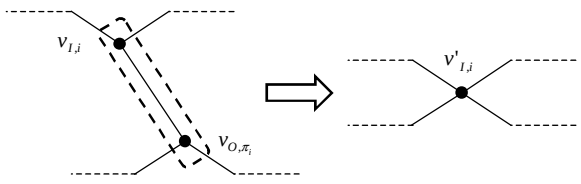


Fig. 6. Contraction of an edge.

An upper bound on the girth of G'_{TB} is derived in App. I. Applying (20), we obtain:

[1]We will see later that, for tail-biting codes, some cycles do not correspond to codewords. However, under certain constraints, this will not invalidate our result.

[2]This property is due to the bipartite nature of the set of vertices V. Suppose that a path in G contains an odd number of edges in E_Π. Then, if the initial vertex $v_i \in V_1$ (resp. $v_i \in V_2$), then the final vertex $v_f \in V_2$ (resp. $v_f \in V_1$). Therefore, the path cannot be closed.

$$g(G'_{TB}) \leq g_{ub}(G'_{TB}) \tag{7}$$

where

$$g_{ub}(G'_{TB}) \leq 2\log_3(K+1) - 1. \tag{8}$$

Eq. (8) provides a preliminary insight on the asymptotical behavior of the minimum distance turbo codes:

$$d_m \leq w \log_3(K)$$

where the w coefficient corresponds to the Hamming weight of error events corresponding to edges in G'_{TB}.

V. The upper bound

In order to derive an upper-bound on the minimum distance of the code, we must consider that every edge sequence in $G'_{TB} \cap E_1$ and $G'_{TB} \cap E_2$ consisting of k consecutive edges corresponds to an error event of length $1 + kp$ trellis sections.

Using (1), we can write the output weight corresponding to an error event of length $1 + kp$ for a *systematic* constituent encoder:

$$w_H(\mathbf{c}_1) \leq k\alpha + \beta$$

Hence, we can define the *maximum equivalent weight* of an error event of length $kp + 1$ as

$$w_{e,\max} \triangleq \max_{k>0} \left[\frac{k\alpha + \beta}{k} \right]. \tag{9}$$

which is clearly obtained for $k = 1$:

$$w_{e,\max} = \alpha + \beta. \tag{10}$$

In order to estimate the Hamming weight of the information word, we consider a cycle C' of length l in G'_{TB} which results from applying edge contraction to cycle C in G_{TB}. Since two edges in E_Π cannot be adjacent, the length of C' cannot exceed $2l$. Therefore, the maximum weight of the information sequence corresponding to the considered cycle is l. ¿From this consideration, and from (8) and (10), we obtain the following result:

$$d_m \leq (w_{e,\max} + 1)\ g_{ub}(G'_{TB}) \tag{11}$$

or:

$$d_m \leq (\alpha + \beta + 1)\,[2\log_3(K+1) - 1] \tag{12}$$

This result depends only on few relevant and easily derived parameters of the turbo code. Moreover, its computation is of very low complexity. However, a simpler expression can be used when the parameters α and β of the constituent encoders are not available. This simpler expression derives from the second part of (1):

$$d_m \leq (p+2)\,[2\log_3(K+1) - 1]. \tag{13}$$

This result only requires the knowledge of ν, the memory of the constituent encoders.

A. Considerations on some types of cycles and their associated codewords

Some types of cycles in G_{TB} and G'_{TB} deserve a further analysis: for example, two edges may be partially overlapping (see Fig. 7). The information word associated to this pair of edges corresponds to the modulo-2 sum of two weight-2 information sequences:

$$\mathbf{u} = \mathbf{u}_1 + \mathbf{u}_2 = (D^i + D^k) + (D^j + D^l)$$

and, as long as the constituent codes are linear, the resulting code sequence is the modulo-2 sum of the two error events:

$$\mathbf{c}_1 = CC_1(\mathbf{u}_1) + CC_1(\mathbf{u}_2).$$

This results in a longer error event starting in position i and terminating in position l.

In this case, the codeword weight is overestimated, because, in the overlapping region, trellis transition are counted twice (or multiple times, if more than two error events are overlapping). This situation results in a possibly looser, but still valid, upper bound.

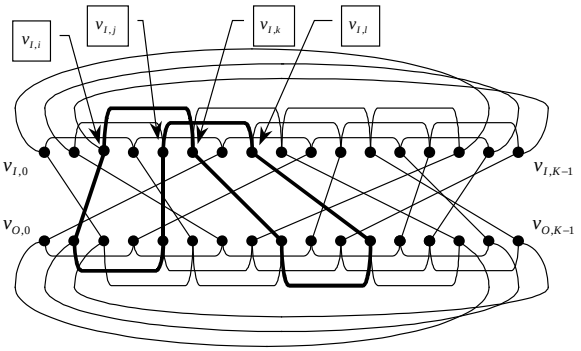


Fig. 7. An example of cycle with partially overlapping error events.

For tail-biting codes, another class of cycles in G_{TB} has to be considered: cycles containing no edges in E_Π (see Fig. 8). Since these cycles do not correspond to codewords (or correspond to information words with zero Hamming weight), they could invalidate our result. However, their length is at least $\frac{K}{p}$ (the minimum length is obtained when K is an integer multiple of p), and, under the following constraint:

$$\frac{K}{p} \geq 2\log_3(K+1) - 1 \geq g(G_{TB}) \tag{14}$$

none of these cycles can be the shortest in G_{TB}. Therefore, we are sure that the shortest cycle contains at least two edges in E_Π, and hence it corresponds to a codeword.

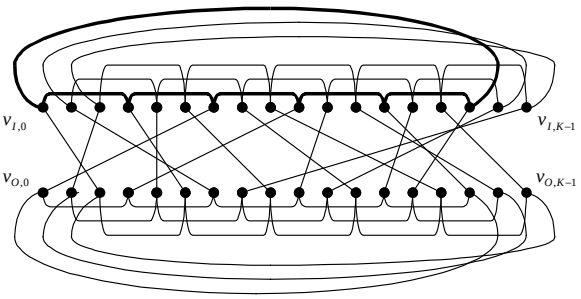


Fig. 8. An example of cycle containing no crossing edges.

Tab. II shows the minimum values of K for which the upper bound for tail-biting codes is valid for different values of ν.

TABLE II
MINIMUM VALUES OF K FOR WHICH THE UPPER BOUND FOR TAIL-BITING CODES IS VALID.

ν	$K_{\min}$
2	11
3	41
4	115
5	289

VI. TERMINATED TURBO CODES

When constituent codes are terminated, our model must be adapted to the new structure. First, we construct a graph leading to a result independent of the termination technique, and then we propose a graph for independently terminated turbo codes resulting in a tighter upper bound.

A. General result on terminated turbo codes

By performing a simple modification on G_{TB}, we obtain an extended permutation graph G_T for terminated turbo codes which is independent on the adopted termination technique and then we will apply edge contraction to obtain a graph G'_T.

The trellis of independently terminated turbo codes cannot be considered *circular*, therefore edges

$$e_i = (v_{I,i}, v_{I,((i-p) \mod K)}),\ i = 0, \ldots, p-1$$

and

$$e_j = (v_{O,j}, v_{O,((j-p) \mod K)}),\ j = 0, \ldots, p-1$$

must be removed (see Fig. 9).

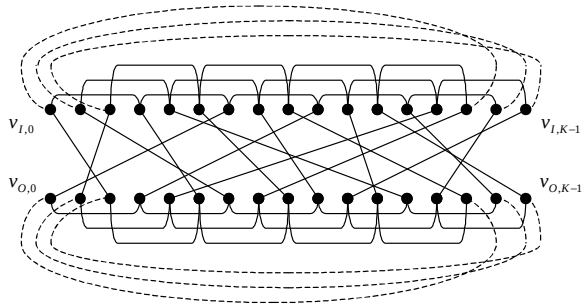


Fig. 9. Modifications of G_{TB} to obtain the graph for terminated turbo codes G_T. The dashed edges are removed.

Now, we apply edge contraction on the set C_Π. The cycles in the resulting graph still correspond to codewords, and their length is still related to the codeword weight.

Due to edge contraction, the obtained graph G'_T contains some vertices whose degree is reduced to 3 or 2. The latter corresponds to the following situation:

$$\begin{aligned} \exists i \quad \in \quad & \{0, \ldots, p-1, K-p, \ldots K-1\} : \\ & \pi_i \in \{0, \ldots, p-1, K-p, \ldots K-1\} \end{aligned}$$

Taking into account these considerations, in App. II an upper bound on the girth of a graph with the characteristics of G'_T is derived. Applying (25) with $n = K$ and $m = 2p$, we obtain:

$$\begin{aligned} d_{m,T} \quad \leq \quad & 2(\alpha+\beta+1) \times \\ & \times \quad \left[\log_3\left(2K - 1 - 4\left\lfloor\frac{p}{2}\right\rfloor\right) + \left\lfloor\frac{p}{2}\right\rfloor\right] \end{aligned} \tag{15}$$

which is an upper bound on the minimum distance of *terminated* turbo codes, independently of the adopted termination technique.

Being a general result with respect to the adopted termination technique, this result is looser than the result obtained for tail-biting codes. Indeed, G_{TB} can be considered as a special case of G_T for tail-biting codes.

Comparing (15) and (12), we can see that the difference between the two results becomes negligible for $K \to \infty$. In this sense, (12) can be considered as an asymptotic result for independently terminated turbo codes.

This result, besides being general with respect to the adopted termination technique, does not suffer from the limitation corresponding to (14), since all the cycles of G_{TB} contain at least two edges in E_Π.

B. Independent termination of constituent codes

A trellis termination technique often used in parallel concatenated convolutional codes consists in forcing both encoders to the all-zero state independently at the end of each block. This technique implies the existence of certain types of codewords generated by information words of weight 1, which consist of error events ending in the trellis termination. These codewords have not been taken into account in the previous analysis. Unfortunately, it is not straightforward to construct a graph modelling this behavior. However, in order to show how the independent termination of the encoders determines the value of the upper bound, we will refer to a slightly different termination rule which introduces such codewords: after the K information bits are sent to each encoder, another $p-1$ steps are added. The information bits u_i corresponding to the termination are determined by the following rule:

$$u_i = \overset{K-1}{\underset{\substack{j=0 \\ j \bmod p = i \bmod p}}{\overset{\oplus}{\sum}}} u_j, \quad K \leq i < K + p - 1 \tag{16}$$

where $\overset{\oplus}{\sum}$ indicates a modulo-2 summation.

We will initially perform modifications on G_{TB} to obtain an extended permutation graph G_{IT} for independently terminated turbo codes, and then we will apply edge contraction to obtain G'_{IT}.

The trellis of independently terminated turbo codes cannot be considered *circular*, therefore edges

$$e_i = (v_{I,i}, v_{I,((i-p) \mod K)}),\ i = 0, \ldots, p-1$$

and

$$e_j = (v_{O,j}, v_{O,((j-p) \mod K)}),\ j = 0, \ldots, p-1$$

must be removed. Moreover, since the termination consists of ν further trellis steps at the end of each block, the corresponding part of the graph must be modified. The chosen approach is to extend the graph beyond the termination by adding $2(p-1)$vertices, and $4(p-1)$ additional edges corresponding to error events spanning beyond the end of the *information* word. Moreover, an edge $(v_{I,K+p-2}, v_{O,K+p-2})$ is added to take into account error events forced to terminate after the end of the information word. This modification results in considering also some odd-weight information words in $\mathcal{U}'$. In Fig. 10, the removed and added edges are highlighted, and Fig. 11 shows the resulting graph G_{IT}.

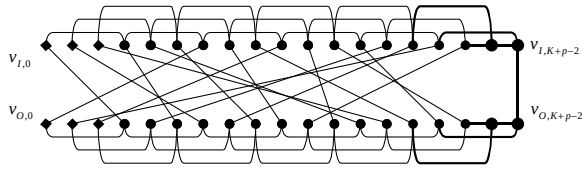


Fig. 10. Modifications of G_{TB} to obtain the graph for terminated turbo codes. Thick edges and bold vertices are added. Square vertices have their degree reduced.

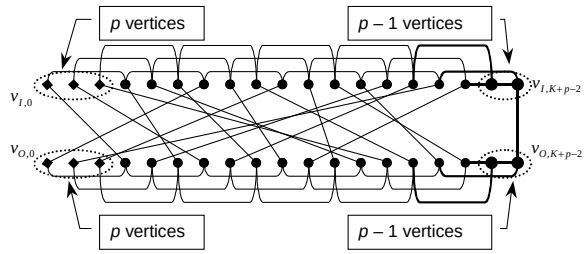


Fig. 11. Extended permutation graph for independently terminated turbo codes.

Now, we apply edge contraction on the set C_Π. Moreover, the $2(p-1)$ termination vertices are pairwise contracted, resulting in $p-1$ degree-4 vertices (see Fig. 12). Since contraction does not involve edges corresponding to error events, the cycles in the resulting graph still correspond to codewords, and their length is still related to the codeword weight.

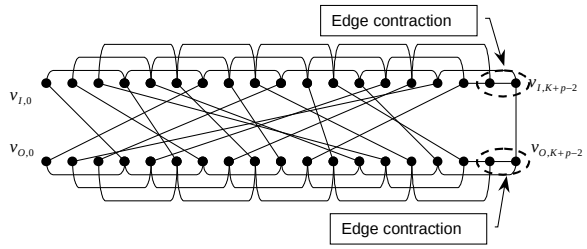


Fig. 12. Edge contraction applied to termination vertices of an extended permutation graph.

If we assume that

$$\pi_i > p, \quad 0 \le i < p \tag{17}$$

then G'_{IT} contains $2p$ degree-3 vertices corresponding to the first p bits in $\mathbf{u}$ and $\tilde{\mathbf{u}}$. If we release (17), then G'_{IT} contains at most $2p$ degree 2 vertices. In either case, it is straightforward to note that G'_{IT} can be transformed in a 4-regular graph by adding the following p edges in G_{IT}:

$$(v_{I,i}, v_{O,i}),\ i = 0, \ldots, p-1$$

and not contracting them. Therefore, the girth of G'_{IT} can be upper bounded as in App. II, where a 4-regular graph with some of missing edges has been considered.

Applying (25) with $n = K + p - 1$ vertices and $m = p$ missing edges, we obtain:

$$\begin{aligned} d_{m,IT} \quad &\le \quad 2(\alpha+\beta+1) \times \\ &\times \quad \left[\log_3\left(2K + 2p - 3 - 4\left\lfloor\frac{p}{4}\right\rfloor\right) + \left\lfloor\frac{p}{4}\right\rfloor\right] \end{aligned} \tag{18}$$

which is an upper bound on the minimum distance of *independently terminated* turbo codes.

Comparing (18) and (12), we can see that the difference between the two results becomes negligible for $K \to \infty$. In this sense, (12) can be considered as an asymptotic result for independently terminated turbo codes.

VII. Comparison with the previously known results

This section shows the obtained results for independently terminated turbo codes compared to the previously known upper bounds [7] [13].

In pursuing a result for the wide class of parallel concatenated convolutional codes with more than two constituent encoders, the authors of [13] introduced an artificial partitioning of the interleaver indices. Moreover, they modelled the multiple concatenation as a hypergraph whose characteristics depend on the artificial partitioning. This is, in our opinion, the main cause of the difference between the results in [13] and our results.

In [7], a partitioning according to the residue classes modulo the period of the constituent encoders is performed. Moreover, an artificial index partitioning (Quasi-Uniform Index partitioning) is performed: a given QUI partitioning excludes from the search all the codewords consisting of error events that cross the partition boundaries.

The derivation of our results discards any artificial index partitioning. This permits to estimate the length of the error events participating in the generation of a codeword more accurately, and hence to derive a tighter upper bound.

In Fig. 13, the result corresponding to terminated turbo codes (15) is compared to [13, Theorem 2]. Since in [13, Theorem 2] puncturing is not considered, in order to perform a fair comparison the rate $^1/_3$ code has been chosen. Our results improve over [13, Theorem 2] of roughly two orders of magnitude for all the considered parameter ranges.

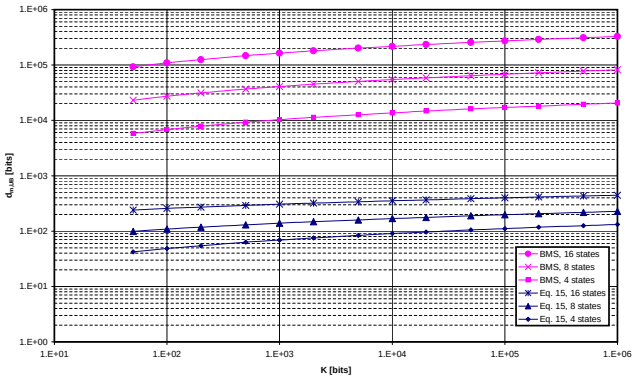


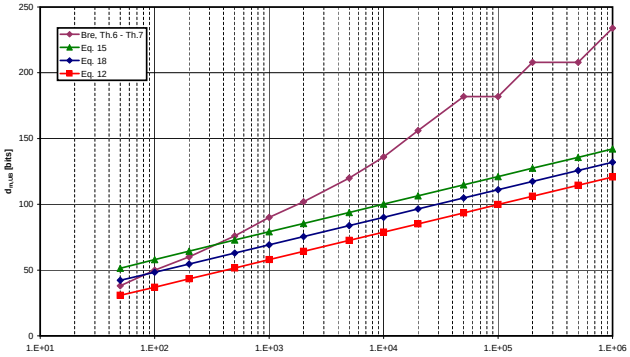
Fig. 13. Result (15) compared to [13, Theorem 2] (labelled BMS).

Fig. 14 shows the new upper bound (15) for independently terminated codes compared to the logarithmic upper bound in [7]. The upper bound for tail-biting codes (12) is always tighter than the one in [13, Theorem 2], while the upper bound for terminated codes (15) is tighter than the one in [7] for a wide range of block lengths.

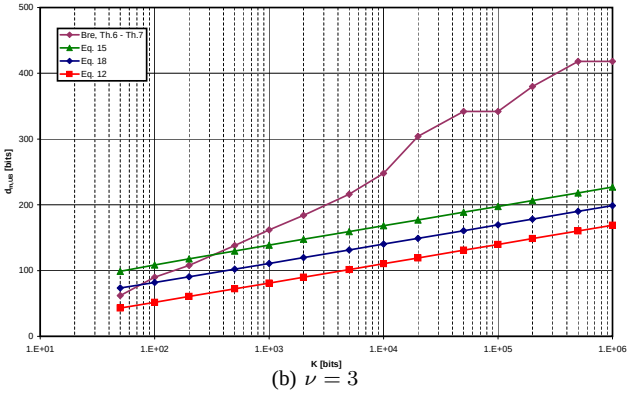
VIII. High-rate turbo codes

In order to obtain high rates ($R \ge 1/2$), the following techniques are usually applied:

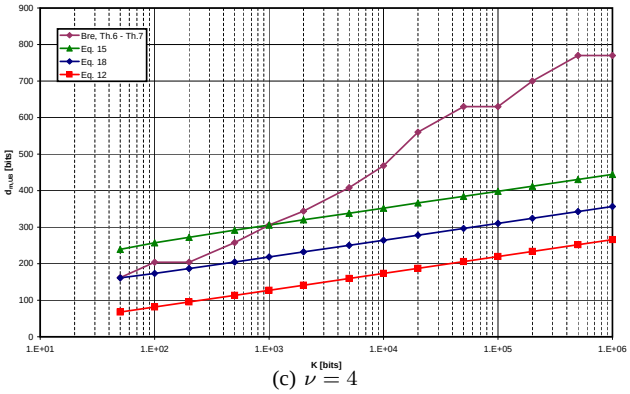
1) Puncturing of the coded bit sequence at the output of the constituent encoders (usually performed when the constituent code rate is $1/n_0$)



(a) $\nu = 2$

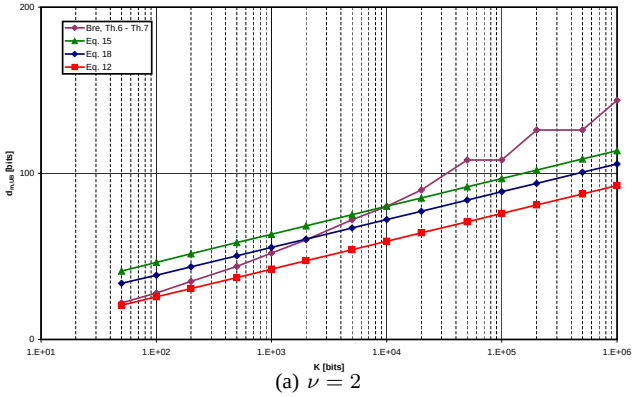


(b) $\nu = 3$

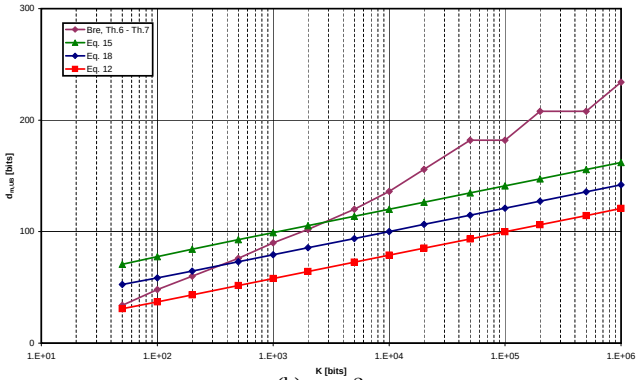


(c) $\nu = 4$

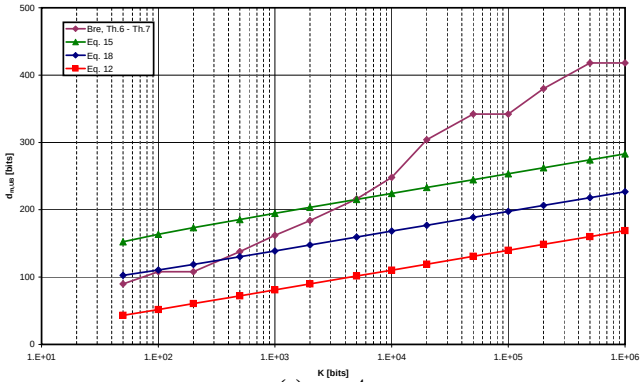
Fig. 14. Upper bounds (12), (15) and (18) for rate $^1/_3$ codes.



(a) $\nu = 2$



(b) $\nu = 3$



(c) $\nu = 4$

Fig. 15. Upper bounds (12), (15) and (18) for rate $^1/_2$ codes.

2) Use of rate k_0/n_0, $k_0 > 1$ constituent encoders.

The two techniques can be applied jointly as well; however, we will analyze them separately.

A. Punctured high-rate turbo codes

Puncturing consists in eliminating some bits in the code sequence at the output of the constituent encoders, and is usually performed applying a periodic puncturing pattern. The elimination of coded bits results in a reduced value of parameters α and β (see Tab. I). Moreover, due to the characteristics of the iterative decoding algorithm used to decode turbo codes, systematic bits are seldom punctured. This results in a larger number of eliminated coded bits rather than information bits. Therefore, in this case, the contribution of information bits to the codeword weight becomes significant. Moreover, while the upper bound in [7] counts the *exact* Hamming weight of the considered information words, (12), (15) and (18) do not allow such accurate estimation. This consideration explains the reason why the upper bound of [7] is tighter for some punctured high-rate turbo codes.

Fig. 15 shows the obtained results for rate $^1/_2$ turbo codes.

B. Rate k_0/n_0 constituent codes

Here, constituent codes with rate k_0/n_0, $k_0 > 1$, are considered. The reference constituent encoder scheme is shown in Fig. 16. For a detailed description of this encoder structure, see [16].

Let k_0 be the number of bits per input symbol. The i-th symbol in the information bit word is $\mathbf{u}_i$. The trellis diagram has 2^{k_0} outgoing edges from each state. We assume the recursive encoder to be characterized by a primitive feedback polynomial.

Under these hypotheses, similarly to rate $1/n_0$ encoders, when the encoder is forced to leave the *all-zero* state by a single non-zero input symbol $\mathbf{u}'_i$, it starts walking through all the non-zero states and never returns to the zero state unless a symbol $\mathbf{u}_{i+jp} = \mathbf{u}'_i$, $j \in \mathbb{Z}^+$ forces it to return to the all-zero state.

By considering an input symbol $\mathbf{u}'_i : w_H(\mathbf{u}'_i) = 1$, we can state that two 1 whose distance in the input *bit* sequence is a multiple of $k_0 p$ cause an error event to start (with the leftmost 1) and terminate (with the rightmost 1).

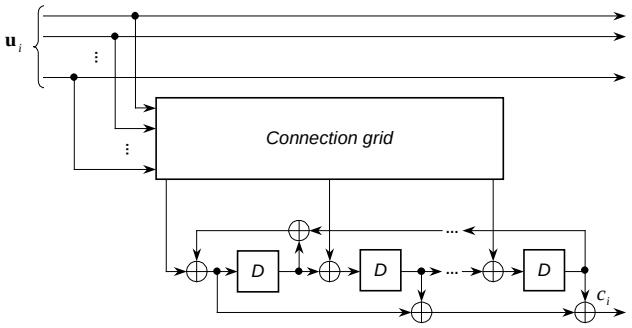


Fig. 16. Rate k_0/n_0 recursive convolutional encoder.

We will initially define the extended permutation graph G_{TB} for rate k_0/n_0 encoders, and then apply edge contraction to obtain G'_{TB}. The extended permutation graph G_{TB} is defined by connecting vertices $v_{I,i}$ and $v_{I,(i+k_0p) \mod K}$:

$$\begin{aligned} E_1 = &\quad \{(v_{I,i}, v_{I,(i+k_0p) \mod K}),\ i = 0, \ldots, K-1\} \\ E_\Pi = &\quad \{(v_{I,i}, v_{O,\pi_i}),\ i = 0, \ldots, K-1\} \\ E_2 = &\quad \{(v_{O,i}, v_{O,(i+k_0p) \mod K}),\ i = 0, \ldots, K-1\} \end{aligned}$$

Clearly, an extended permutation graph for rate $1/n_0$ constituent encoders is obtained when $k_0 = 1$.

With respect to rate $1/n_0$ codes, the resulting extended permutation graph is again 4-regular, and the number of vertices is unchanged, therefore its girth is unchanged. However, parameters α and β are changed according to the encoder connections. In fact, as long as the input symbols are 0,the state trajectory of a k_0/n_0 encoder is equal to that of a rate $1/n_0$ encoder with the same memory and feedback connections. Moreover, if the feedforward connections are equal, also the output sequence is equal; therefore, their parameters α and β are equal.

Example 1: Consider two rate $1/2$ turbo codes. The first code is obtained by puncturing two 8-state recursive convolutional encoders. From Tab. I, we have $\alpha_1 = 2$ and $\beta_1 = 2$.

The second code is obtained by concatenating in parallel two rate $2/3$, 8-state recursive convolutional encoders. In this scheme, puncturing is not performed. Suppose that the constituent encoders are based on a memory $\nu = 3$ shift register with feedback connections 13_8 and feedforward connections 17_8. Since these connections are equal to those of a rate $1/2$ convolutional encoder, its parameters can be derived from Tab. I, row $\nu = 3$, col. $R = 1/3$, that corresponds to *unpunctured* constituent codes. We have $\alpha_2 = 4$ and $\beta_2 = 2$.

Clearly, the upper bound for the second code is larger than the upper bound for the first code, as shown in Fig. 17.

C. Symbol interleaving

Symbol interleaving consists of performing a permutation on the set of symbols instead of bits. Indeed, a permutation

$$\Pi : i \to \pi_i, \quad i \in \left\{0, \ldots, \frac{K}{k_0} - 1\right\}$$

(K an integer multiple of k_0) operating on k_0-bit symbols is equivalent to a permutation

$$\Pi' : i \to \pi'_i, \quad i \in \{0, \ldots, K-1\}$$

having the following property:

$$(i \mod k_0) = (\pi'_i \mod k_0)$$

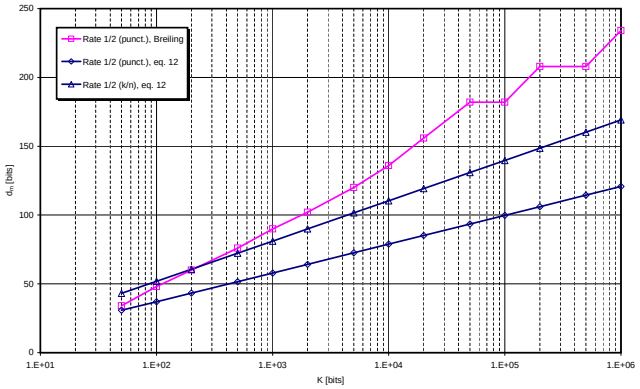


Fig. 17. Upper bounds for a rate $1/2$ 8-state turbo code of Example 1.

By partitioning the set of vertices V_1 (resp. V_2) in k_0 classes $V_{q,1}$ (resp. $V_{q,2}$), $q = 0, \ldots, k_0 - 1$, defined as follows:

$$\begin{aligned} v_{I,i} \in V_{q,1} \quad &\Leftrightarrow \quad (i \mod k_0) = q \\ v_{O,j} \in V_{q,2} \quad &\Leftrightarrow \quad (j \mod k_0) = q \end{aligned}$$

we note that every edge in E_1 and E_2 connects vertex pairs belonging to the same class:

$$\begin{aligned} e_i = (v_{I,i}, v_{I,j}) \quad &\Leftrightarrow \quad j = i + k_0 p \\ &\Leftrightarrow \quad (j \mod k_0) = (i \mod k_0) \end{aligned}$$

Clearly, this is true also for edges in E_2 and, due to symbol interleaving, in E_Π as well. Therefore, no edge connects vertex pairs belonging to different equivalence classes, and the resulting extended permutation graph can be partitioned in k_0 disjoint subgraphs containing K/k_0 vertices each.

The girth of G'_{TB}, which is obtained from G_{TB} by contracting the edges in E_Π, can be upper bounded choosing at random one of the k_0 subgraphs and applying (20) with $n = K/k_0$. The resulting upper bound is:

$$d_{m,SI} < (\alpha + \beta + 1)\left[2\log_3\left(\frac{K}{k_0} + 1\right) - 1\right] \qquad (19)$$

If $K/k_0 \gg 1$, the $+1$ term in the argument of the logarithm can be neglected. By comparing (19) with (12), we note that symbol interleaving causes a reduction of $(\alpha+\beta+2)\log_3(k_0)$ (see Fig. 18). Since k_0 is typically low ($k_0 \lesssim 10 \ \rightarrow \ \log_3(k_0) \lesssim 2$), if the block length is large, this reduction is negligible also for low information word lengths.

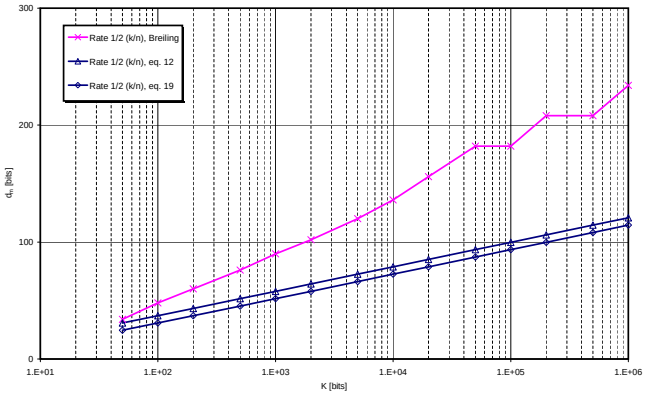


Fig. 18. Effect of symbol interleaving on a rate $1/3$ tail-biting turbo code.

IX. Conclusions

We have shown in the paper that a connection between parallel concatenated convolutional codes and the topological properties of certain associated graphs can be established. The connection has been used to obtain an upper bound on the minimum distance of parallel concatenated convolutional codes. The proposed method has been used both for tail-biting and independently terminated turbo codes. Moreover, it has been extended to rate k_0/n_0 constituent codes with bit- and symbol-interleaving. The new bound is easy to compute, and improves over known bounds for a wide range of constituent codes parameters.

Appendix I
The girth of 4-regular graphs

Here, we derive an upper bound on the girth of 4-regular graphs. The result is obtained following the derivation of the *naive bound* in [17].

Theorem 1: The girth g of a 4-regular graph with $|V| = n$ vertices is

$$g \leq 2\log_3(n+1) - 1 \tag{20}$$

Proof: Choose an edge $e = (v_i, v_j) \in E$ at random. If $g > 3$, vertices v_i, v_j and their neighboring vertices must be distinct (levels $l = 0$ and $l = 1$ in Fig. 19); otherwise, a cycle of length 3 would be present. Therefore, the number of vertices of G is $n \geq 2 + 2 \times 3$. If $g > 5$, also the neighbors of the previously considered vertices must be distinct (levels 0 to 2 in Fig. 19). Therefore, the number of vertices of G is $n \geq 2(1 + 3 + 3^2)$. Extending to higher odd values of g, we obtain:

$$\begin{aligned} g > 3 &\Rightarrow n \geq 2 + 2 \times 3 \\ g > 5 &\Rightarrow n \geq 2(1 + 3 + 3^2) \\ \ldots &\Rightarrow \ldots \\ g > g_o &\Rightarrow n \geq 2\left(1 + 3 + \ldots + 3^{\frac{g_o-1}{2}}\right) = 3^{(g_o+1)/2} - 1 \end{aligned}$$

therefore:

$$g_o \leq 2\log_3(n+1) - 1 \tag{21}$$

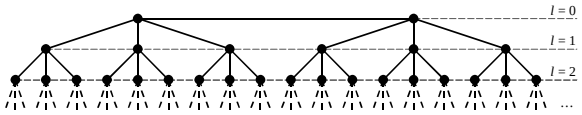


Fig. 19. Construction for the derivation of the upper bound on the girth (odd values).

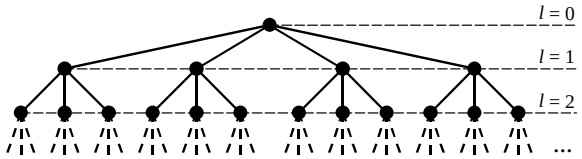


Fig. 20. Construction for the derivation of the upper bound on the girth (even values).

If g is even, we operate in a similar way, but choosing as a starting point a vertex instead of an edge (see Fig. 20. We obtain:

$$\begin{aligned} g > 2 &\Rightarrow n \geq 1 + 4 \\ g > 4 &\Rightarrow n \geq 1 + 4(1 + 3) \\ \ldots &\Rightarrow \ldots \\ g > g_e &\Rightarrow n \geq 1 + 4\left(1 + 3 + \ldots + 3^{\frac{g_e}{2}-1}\right) = \\ &\qquad = 1 + 2(3^{\frac{g_e}{2}} - 1) = 2 \times 3^{\frac{g_e}{2}} - 1 \end{aligned}$$

therefore:

$$g_e \leq 2\log_3(n+1) - 2\log_3 2 \tag{22}$$

From (21) and (22), we obtain:

$$g \leq \max(g_e, g_o) = 2\log_3(n+1) - 1 \tag{23}$$

and this proves the theorem.

□

Appendix II
The girth of an almost-4-regular graph

In the following, we derive an upper bound on the girth of a graph $G = (V, E)$ which results from removing a small number m of edges from a 4-regular graph with the additional constraint that the minimum degree of G is not lower than 2.

Theorem 2: Let G be a graph on $|V| = n$ vertices and $|E| = 2n - m$ edges. Then its girth is

$$g \leq g_{ub}^{(4-)} \tag{24}$$

where

$$g_{ub}^{(4-)} = 2\log_3\left(2n - 1 - 4\left\lfloor\frac{m}{4}\right\rfloor\right) + 2\left\lfloor\frac{m}{4}\right\rfloor. \tag{25}$$

Proof:

As in App. I, we derive the upper bound following the derivation of the *naive bound* in [17].

Suppose we construct the graph by removing m edges form a 4-regular graph. Starting from a randomly selected edge, we construct the spanning tree resulting form a breadth-first visit of the original 4-regular graph. If we stop our construction at level L, the total number of vertices in the tree is

$$n_L = 2(1 + 3 + \ldots + 3^L) = 3^{L+1} - 1$$

Now, we count the total number of vertices of the same spanning tree, considering that there are m missing edges. If such missing edges are not in the spanning tree[3], they do not result in a reduced tree height, and consequently in a reduced girth. Hence, we restrict to the case where each missing edge is placed between level l_i and level $l_i + 1$, $i = 0, \ldots, m-1$, and compute the number of vertices in a tree with L levels of vertices:

$$\begin{aligned} n'_L &= n_L - \sum_{i=0}^{m-1}\left(1 + 3 + 3^2 + \ldots + 3^{L-l_i-1}\right) \\ &= n_L - \sum_{i=0}^{m-1}\frac{3^{L-l_i} - 1}{2} \\ &= 2\sum_{i=0}^{L} 3^i - \sum_{i=0}^{m-1}\frac{3^{L-l_i} - 1}{2} \end{aligned} \tag{26}$$

It is straightforward to note that, for a given number of levels L, the minimum number of vertices is obtained when levels l_i, $i =$

[3]A spanning tree contains only $|V| - 1$ edges.

$0, \ldots, m-1$ are minimum. For a fixed number of vertices $|V|$, this turns into the largest height for the spanning tree, hence in the largest girth.

Considering that, starting from a 4-regular graph, there can be at most 4 missing edges between each pair of consecutive levels (otherwise, the minimum degree would be reduced below 2), we obtain:

$$\begin{aligned} n'_L &= 2\sum_{i=0}^{L} 3^i - 2\sum_{i=0}^{\lfloor m/4 \rfloor - 1} \left(3^{L-i} - 1\right) - a \\ &= 3^{L-\lfloor m/4 \rfloor + 1} - 1 + 2\left\lfloor \frac{m}{4} \right\rfloor - a \\ &\geq \frac{3^{L-\lfloor m/4 \rfloor + 1}}{2} + \frac{1}{2} + 2\left\lfloor \frac{m}{4} \right\rfloor = n''_L \end{aligned} \tag{27}$$

where

$$a = (m \mod 4)\frac{3^{L-\lfloor m/4 \rfloor} - 1}{2}.$$

This results from considering $(m \mod 4) \leq 3$. The girth of G is related to the number of levels needed to visit all the vertices:

$$g \leq 2L_m + 2 \tag{28}$$

where

$$L_m = \inf\left\{L : n'_L \geq n\right\} \tag{29}$$

Since $n''_L \leq n'_L$, we derive the following result:

$$g \leq 2\log_3\left(2n - 1 - 4\left\lfloor \frac{m}{4} \right\rfloor\right) + 2\left\lfloor \frac{m}{4} \right\rfloor. \tag{30}$$

which proves the theorem.

□

ACKNOWLEDGMENTS

The authors would like to thank Dr.-Ing. Marco Breiling for his valuable support.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: turbo codes," in *Proceedings of the International Conference on Communications*, vol. 2, Geneva, Switzerland, 1993, pp. 1064–1070.

[2] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design and iterative decoding," *IEEE Transactions on Information Theory*, vol. 44, no. 3, pp. 909–926, May 1998.

[3] R. Garello, P. Pierleoni, and S. Benedetto, "Computing the free distance of turbo codes and serially concatenated codes with interleavers: Algorithms and applications," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 5, pp. 800–812, May 2001.

[4] S. Dolinar and D. Divsalar, "Weight distributions for turbo codes using random and nonrandom permutations," *TDA Progress Report*, vol. 42-122, pp. 56–65, April-June 1995.

[5] C. Fragouli and R. D. Wesel, "Turbo-encoder design for symbol-interleaved parallel concatenated trellis-coded modulation," *IEEE Transactions on Communications*, vol. 49, no. 3, pp. 425–435, March 2001.

[6] P. O. Vontobel, "On the construction of turbo code interleavers based on graphs with large girth," in *Proceedings of the International Conference on Communications*, New York, USA, 2002, pp. 1408–1412.

[7] M. Breiling, "A logarithmic upper bound on the minimum distance of turbo codes." [Online]. Available: http://www.lnt.de/~breiling/Research/Publications/bre01b.pdf

[8] D. Truhachev, M. Lentmaier, and K. S. Zigangirov, "Some results concerning design and decoding of turbo-codes," *Problems of Information Transmission*, vol. 37, no. 3, pp. 190–205, July-September 2001.

[9] 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, March 1996.

[10] M. Breiling and J. B. Huber, "Upper bound on the minimum distance of turbo codes," *IEEE Transactions on Communications*, vol. 49, no. 5, pp. 808–815, May 2001.

[11] ——, "Combinatorial analysis of the minimum distance of turbo codes," *IEEE Transactions on Information Theory*, vol. 47, no. 7, pp. 2737–2750, Nov 2001.

[12] N. Kahale and R. Urbanke, "On the minimum distance of parallel and serially concatenated codes," submitted to IEEE Transactions on Information Theory. [Online]. Available: http://lthcwww.epfl.ch/~ruediger/papers/weight.ps

[13] L. Bazzi, M. Mahdian, and D. Spielman, "The minimum distance of turbo-like codes." [Online]. Available: http://www-math.mit.edu/~mahdian/tc3.ps

[14] S. Even, A. Pnueli, and A. Lempel, "Permutation graphs and transitive graphs," *Journal of the Association for Computing Machinery*, vol. 19, pp. 400–410, 1972.

[15] G. Solomon and H. van Tilborg, "A connection between block and convolutional codes," *SIAM Journal on Applied Mathematics*, vol. 37, pp. 358–369, October 1979.

[16] C. Berrou, M. Jézéquel, C. Douillard, and S. Kerouédan, "The advantages of non-binary turbo codes," Cairns, Australia, 2001, pp. 61–63.

[17] N. L. Biggs, "Constructions for cubic graphs with large girth," *Electronic Journal of Combinatorics*, vol. 5, 1998.

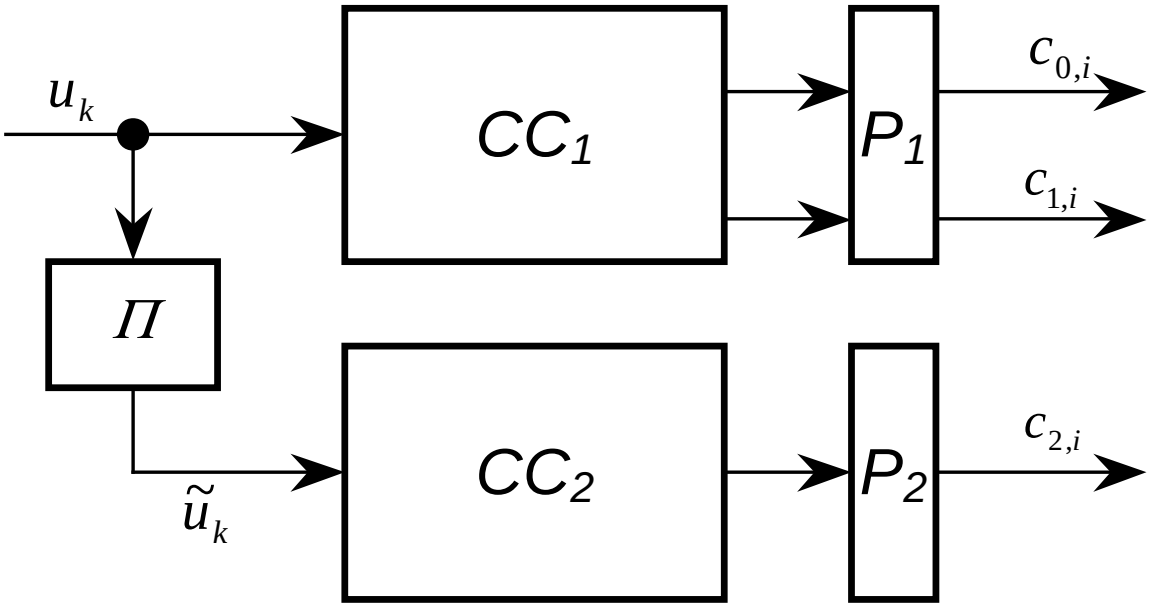


Fig. 1. Reference coding scheme.

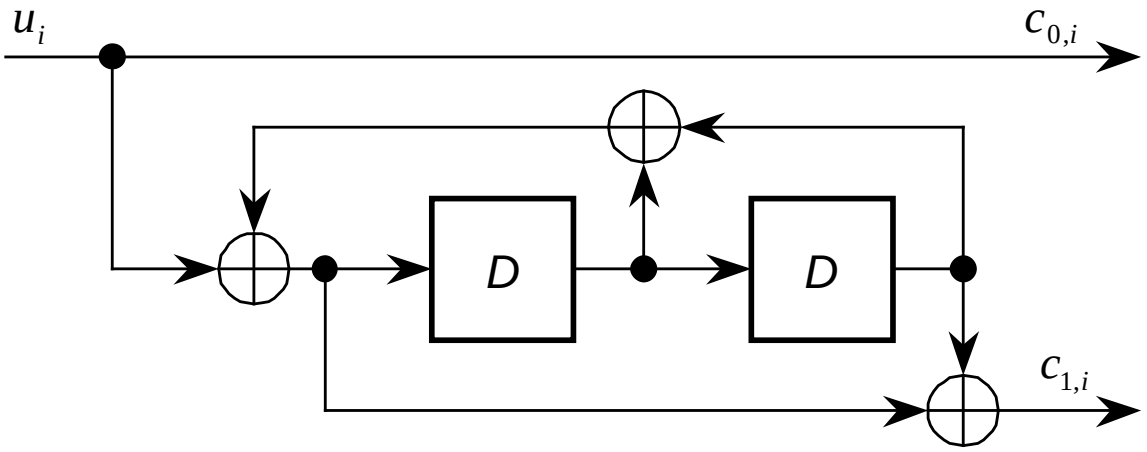


Fig. 2. A 4-state systematic, recursive convolutional encoder.

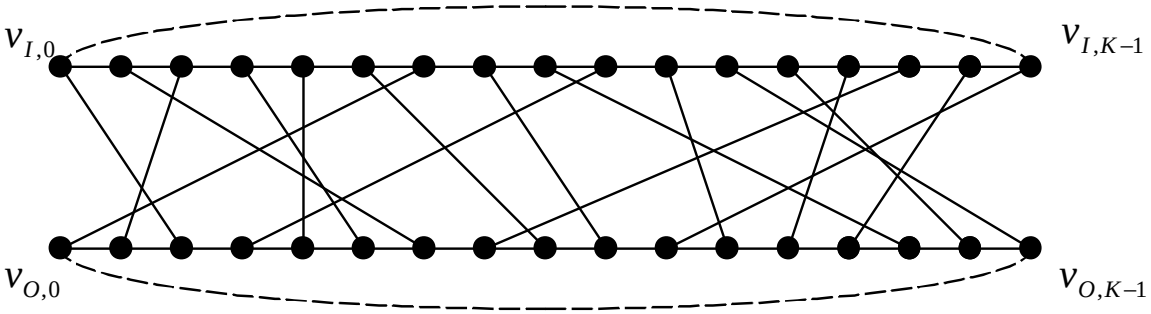


Fig. 3. An example of permutation graph for memory $\nu = 1$ constituent encoders.

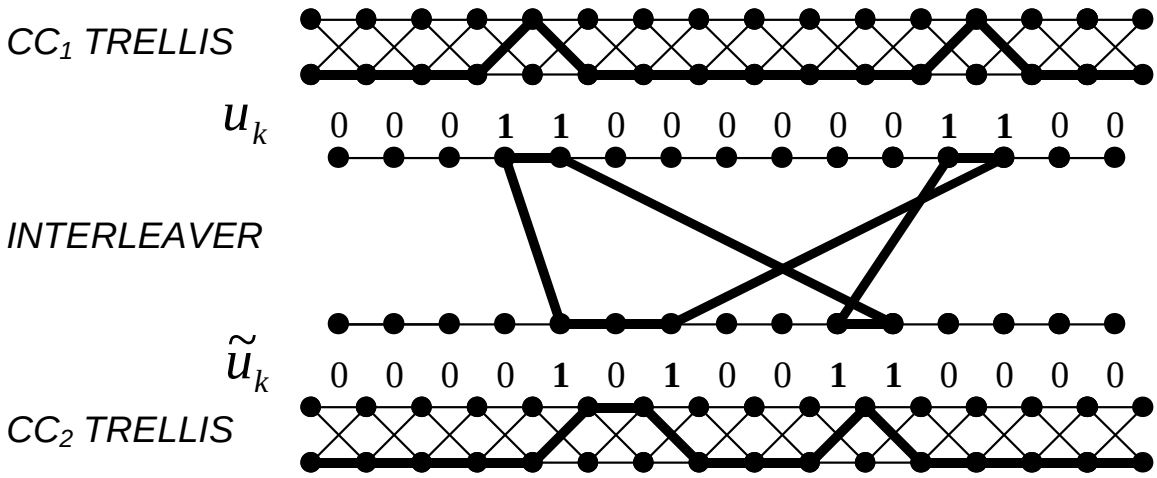


Fig. 4. Correspondence between codewords and cycles.

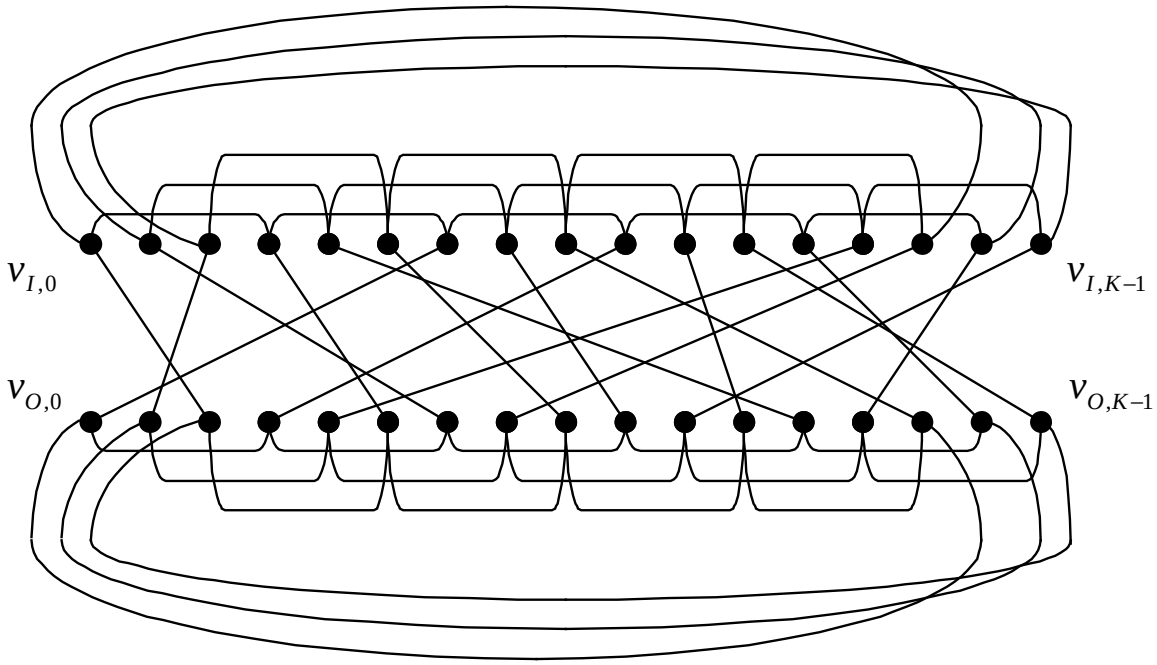


Fig. 5. Extended permutation graph for tail-biting terminated turbo codes.

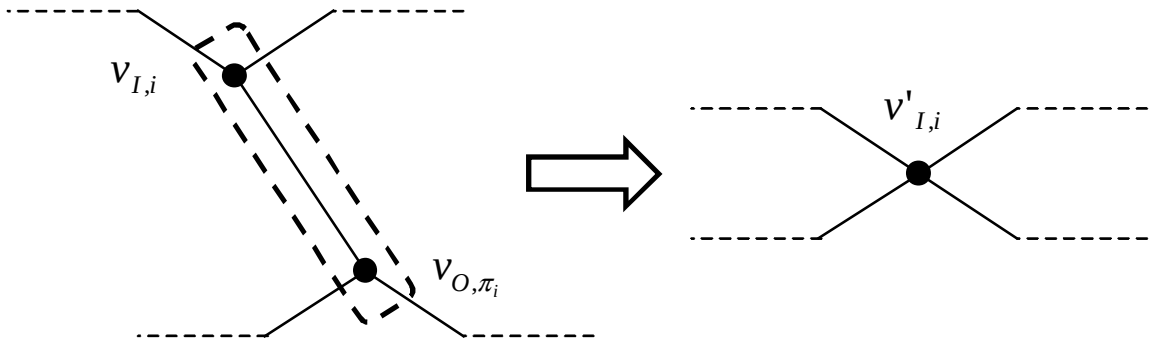


Fig. 6. Contraction of an edge.

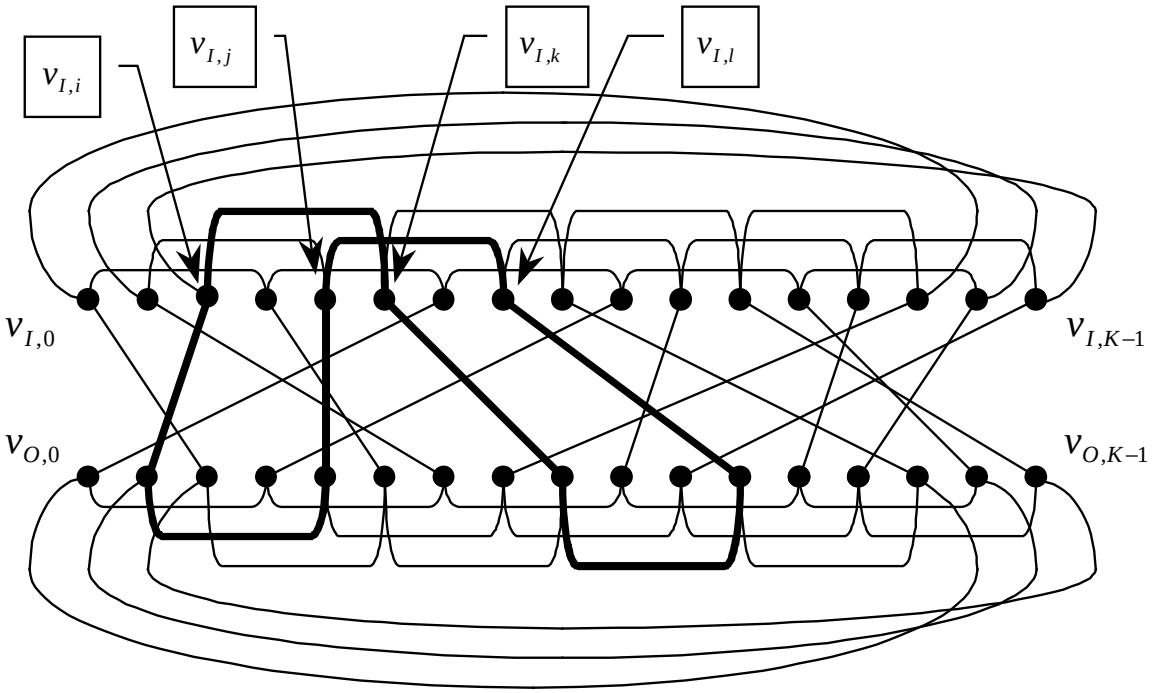


Fig. 7. An example of cycle with partially overlapping error events.

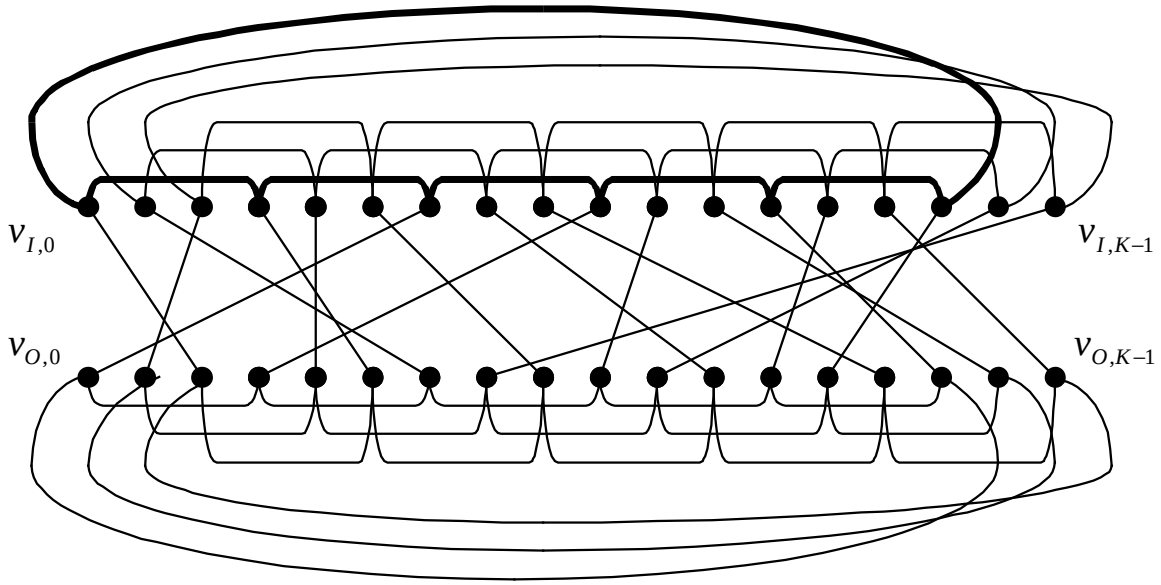


Fig. 8. An example of cycle containing no crossing edges.

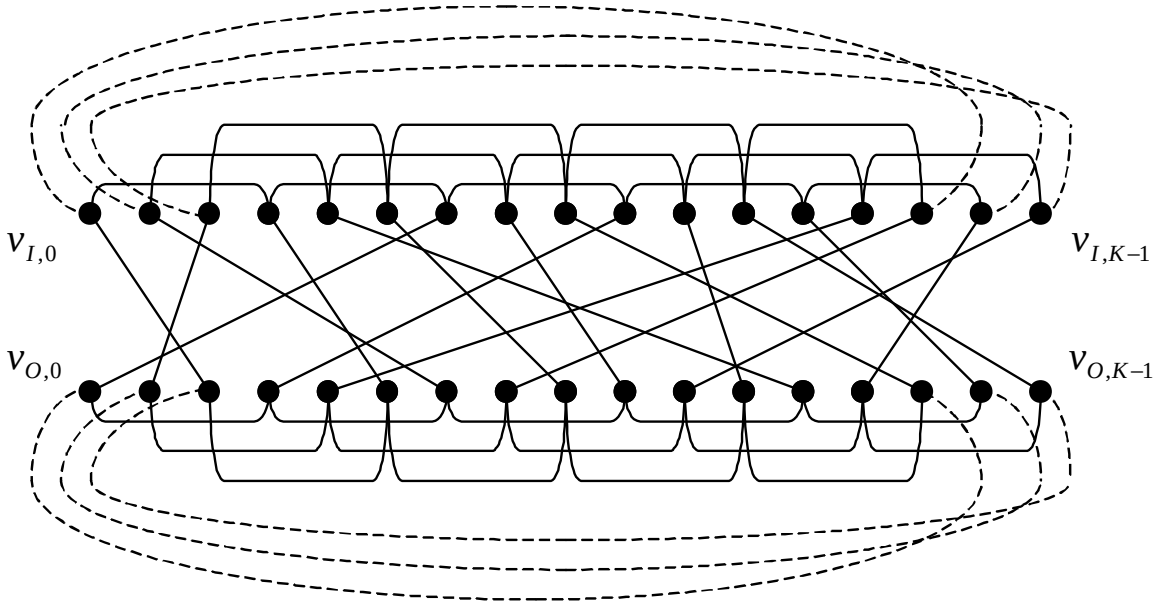


Fig. 9. Modifications of G_{TB} to obtain the graph for terminated turbo codes G_T. The dashed edges are removed.

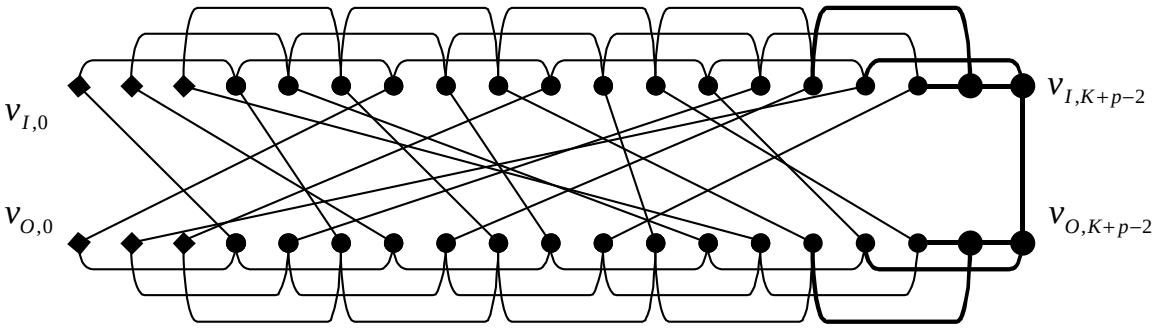


Fig. 10. Modifications of G_{TB} to obtain the graph for terminated turbo codes. Thick edges and bold vertices are added. Square vertices have their degree reduced.

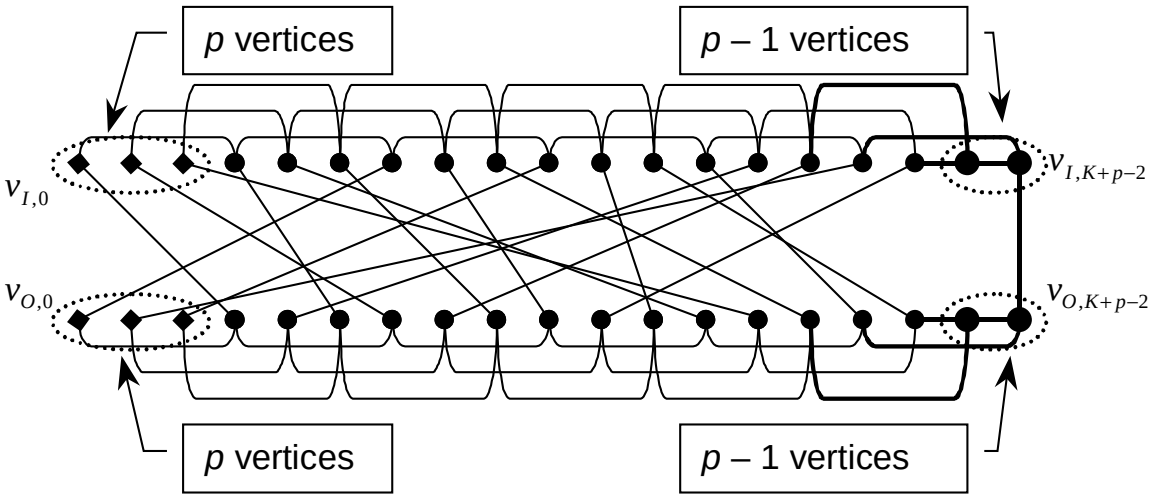


Fig. 11. Extended permutation graph for independently terminated turbo codes.

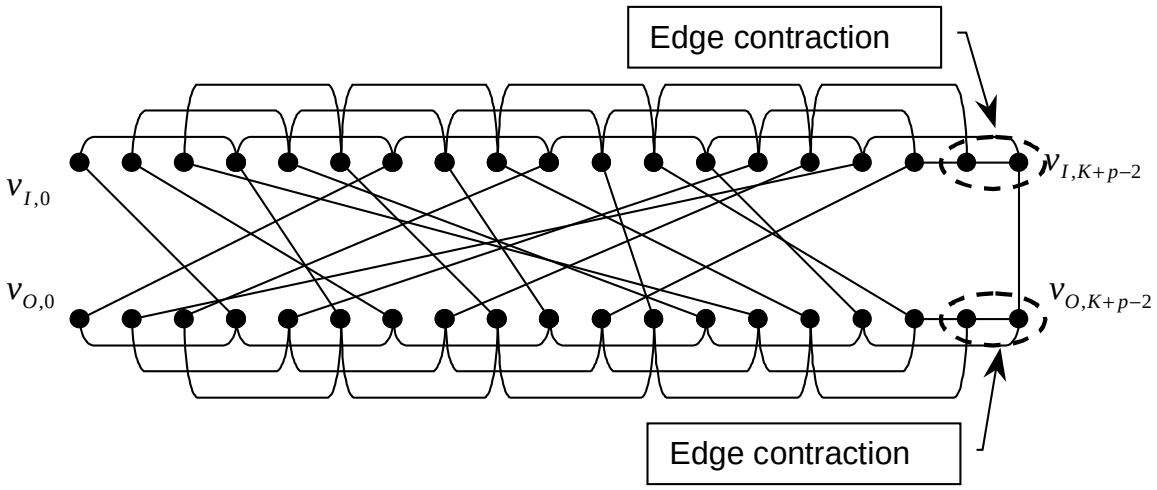


Fig. 12. Edge contraction applied to termination vertices of an extended permutation graph.

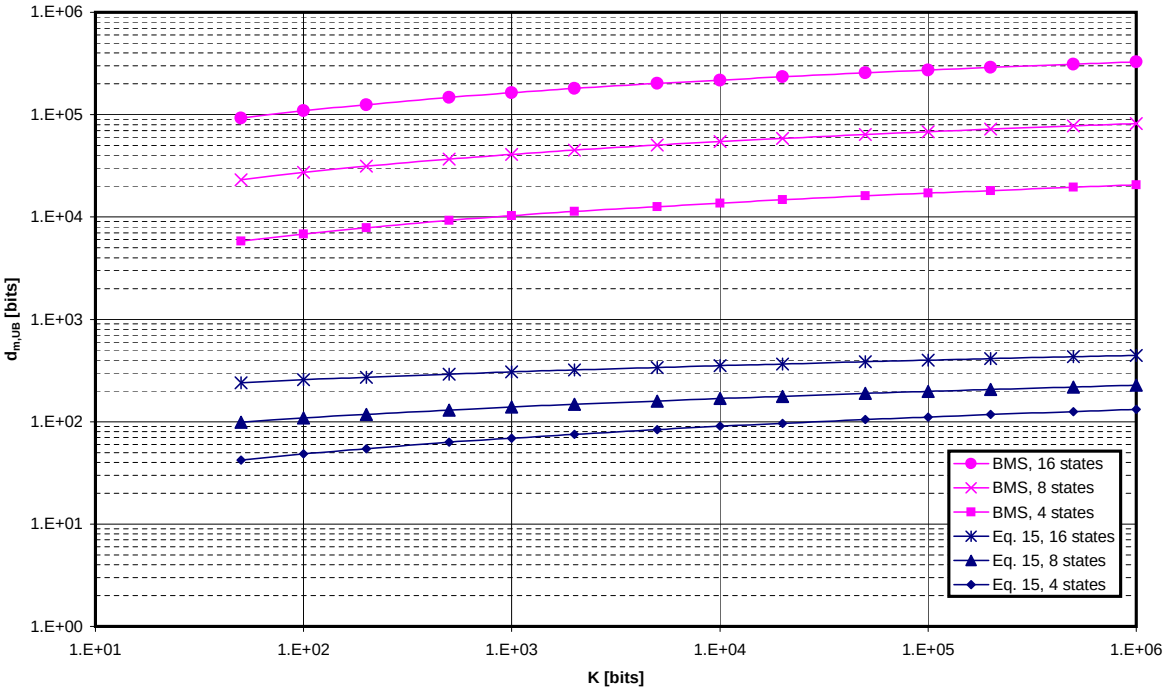
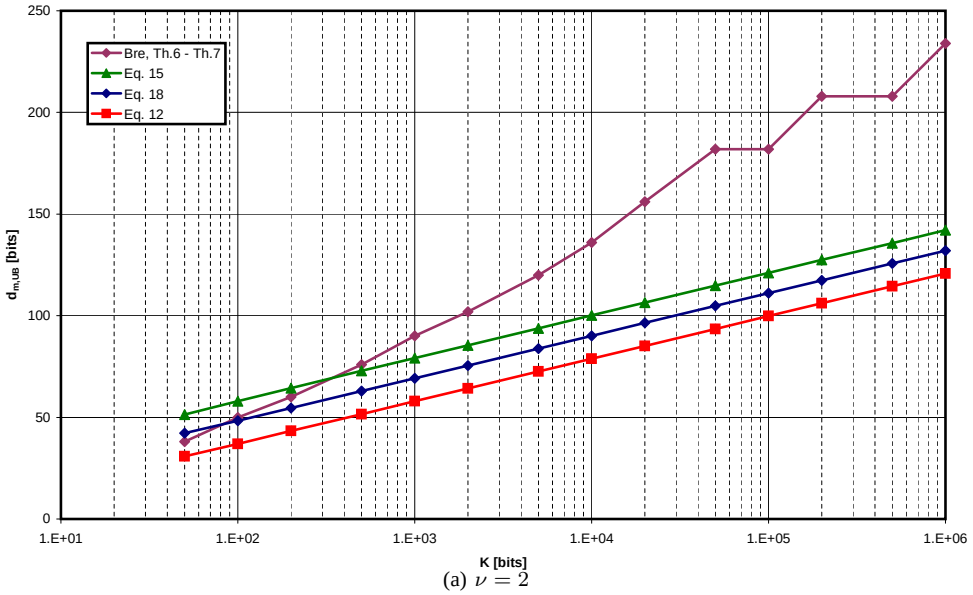
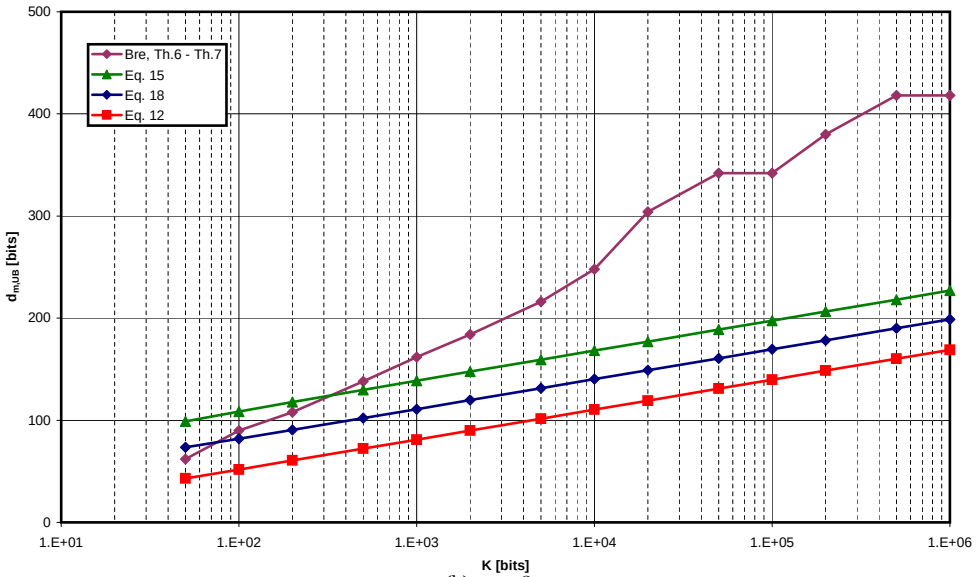


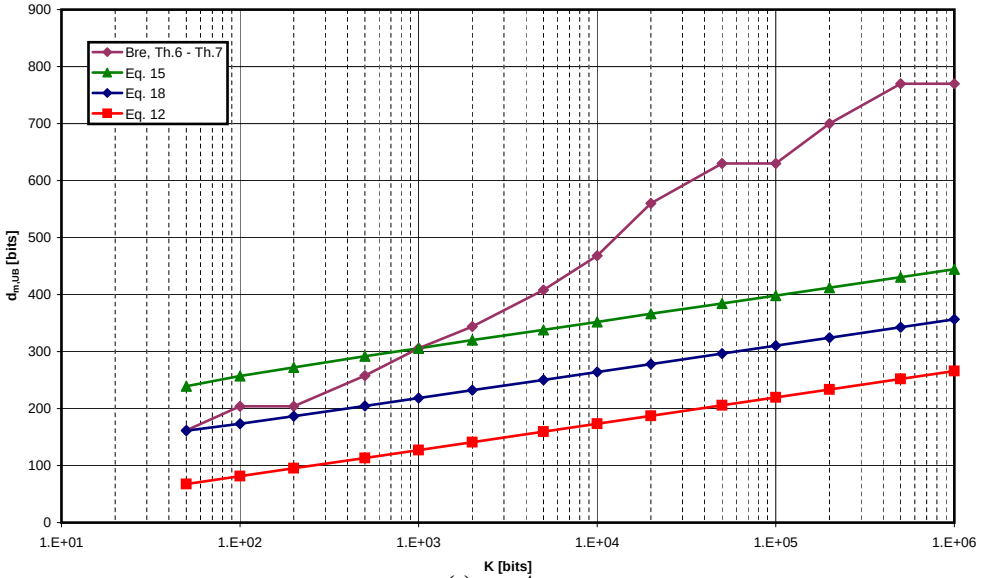
Fig. 13. Result (15) compared to [13, Theorem 2] (labelled BMS).



(a) $\nu = 2$

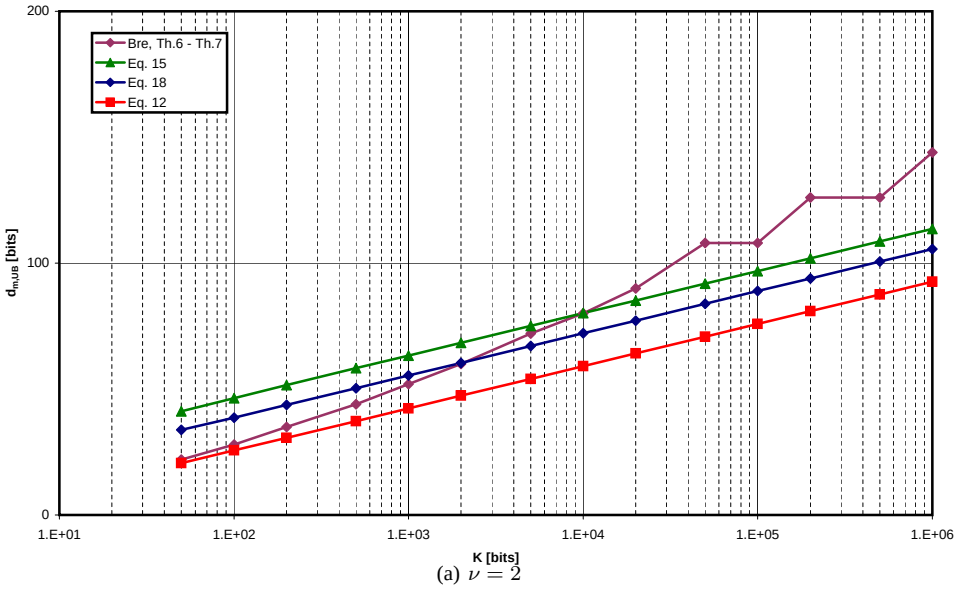


(b) $\nu = 3$

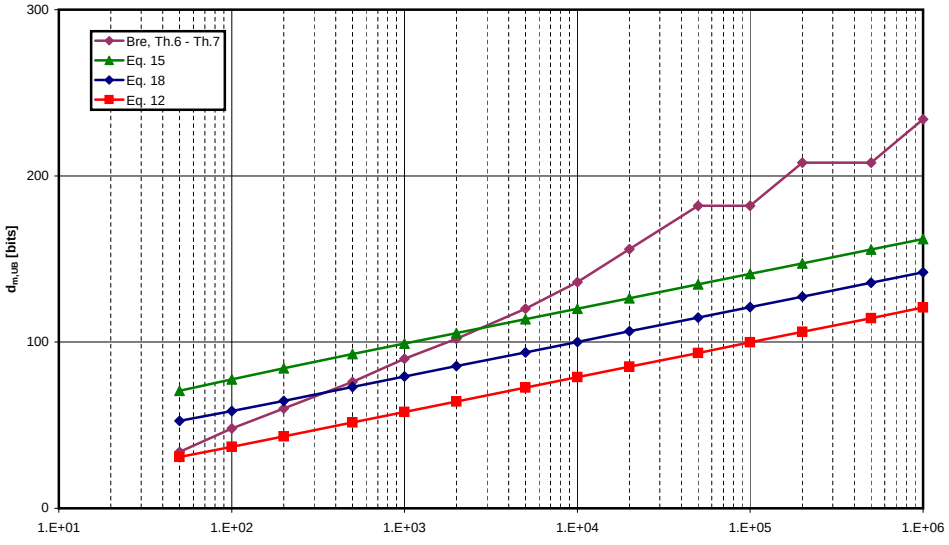


(c) $\nu = 4$

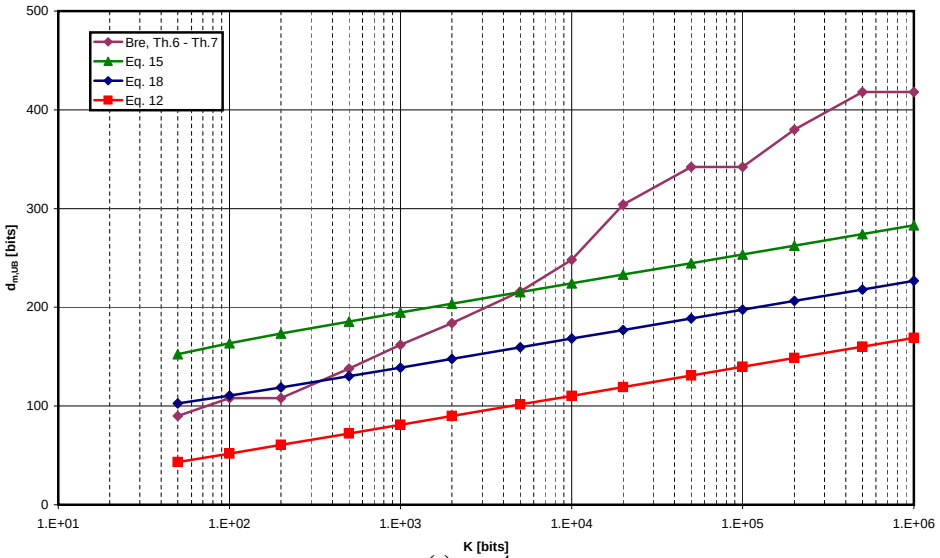
Fig. 14. Upper bounds (12), (15) and (18) for rate $^1/_3$ codes.



(a) $\nu = 2$



(b) $\nu = 3$



(c) $\nu = 4$

Fig. 15. Upper bounds (12), (15) and (18) for rate $^1/_2$ codes.

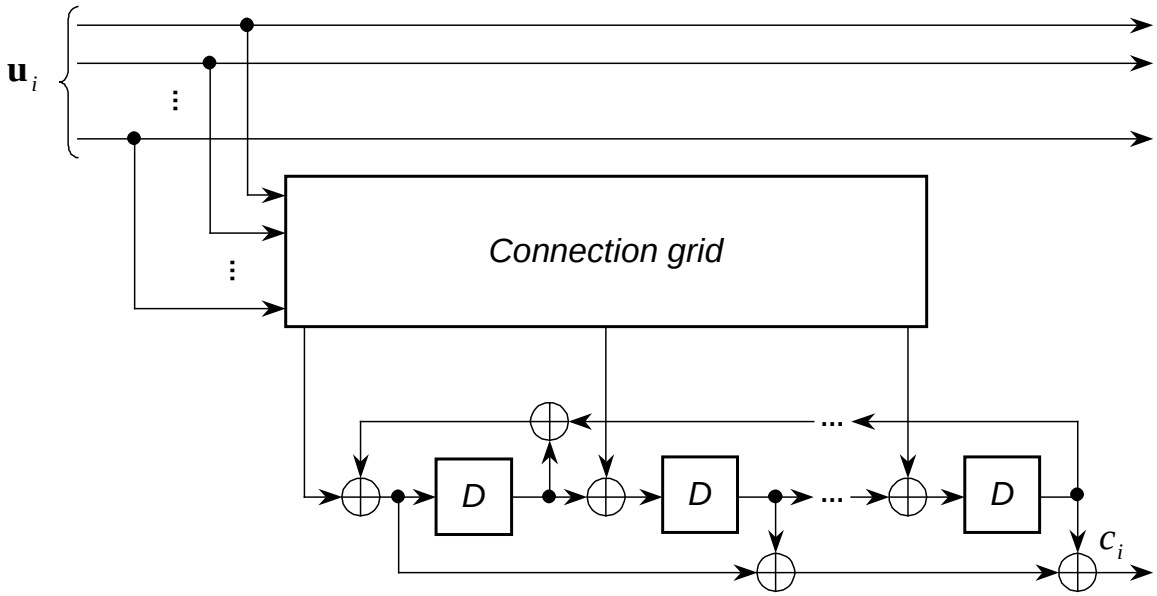


Fig. 16. Rate k_0/n_0 recursive convolutional encoder.

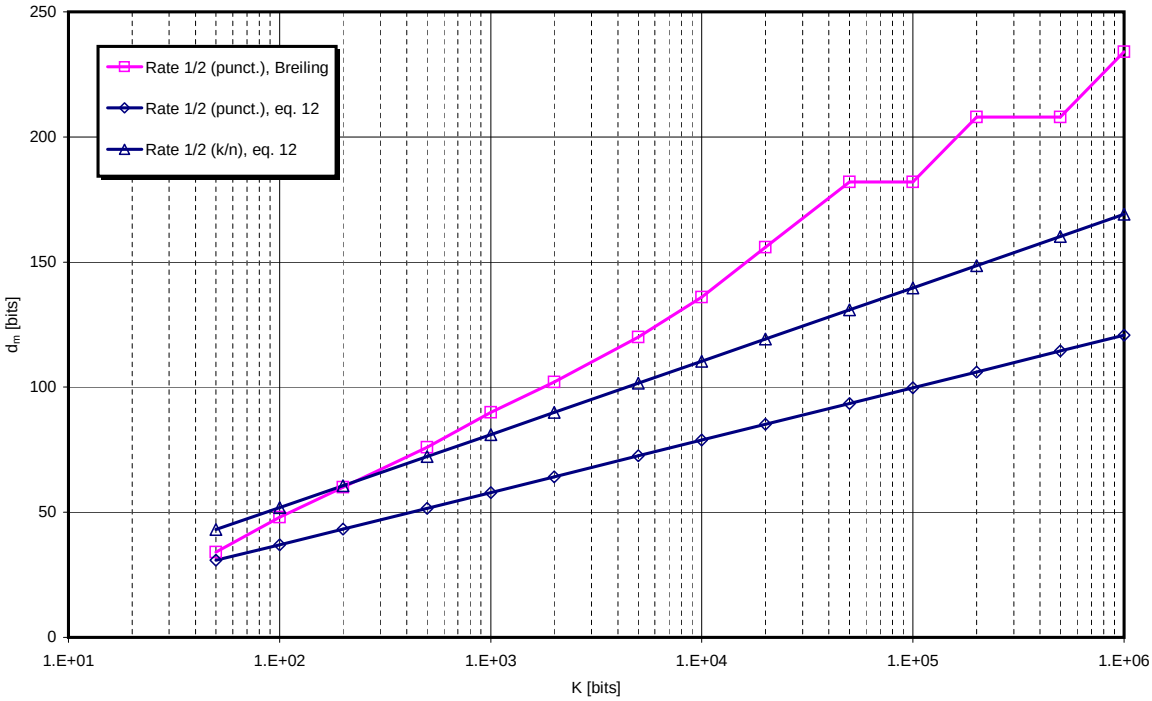


Fig. 17. Upper bounds for a rate $1/2$ 8-state turbo code of Example 1.

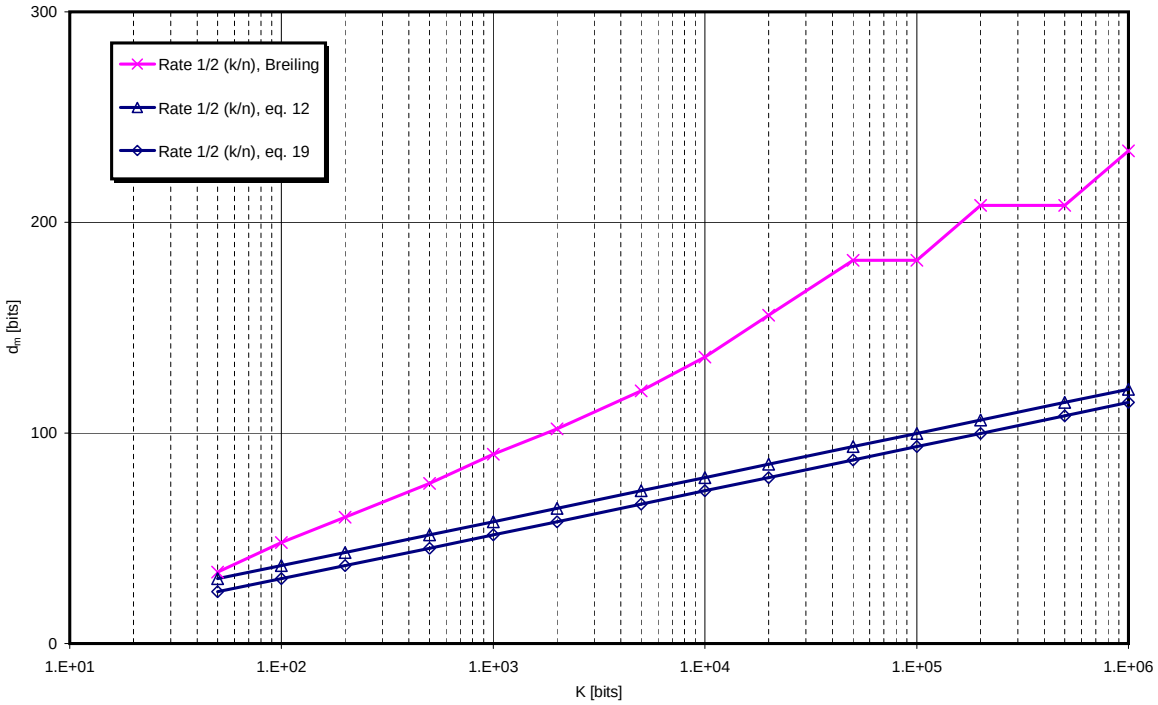


Fig. 18. Effect of symbol interleaving on a rate $1/3$ tail-biting turbo code.

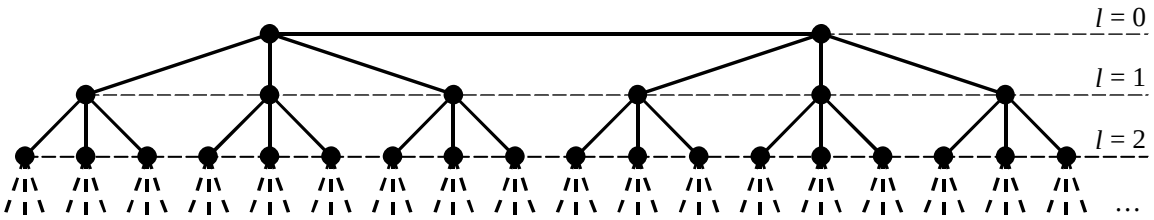


Fig. 19. Construction for the derivation of the upper bound on the girth (odd values).

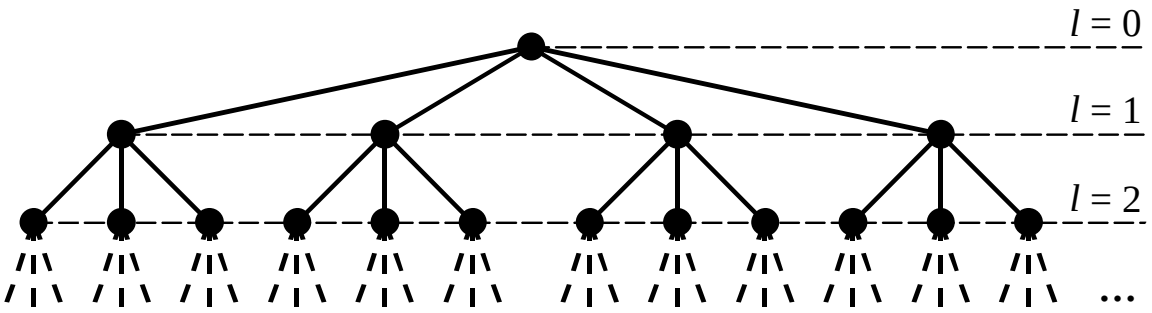


Fig. 20. Construction for the derivation of the upper bound on the girth (even values).