On the Design of Punctured Low Density Parity Check Codes for Variable Rate Systems

Marco Baldi and Franco Chiaraluce

Original scientific paper

***Abstract*— The authors face the problem of designing good LDPC codes for applications requiring variable, that is adaptive, rates. More precisely, the object of the paper is twofold. On one hand, we propose a deterministic (not random) procedure to construct good LDPC codes without constraints on the code dimension and rate. The method is based on the analysis and optimization of the local cycles length in the Tanner graph and gives the designer the chance to control complexity of the designed codes. On the other hand, we present a novel puncturing strategy which acts directly on the parity check matrix of the code, starting from the lowest rate needed, in order to allow the design of higher rate codes avoiding additional complexity of the co/decoding hardware. The efficiency of the proposed solution is tested through a number of numerical simulations. In particular, the puncturing strategy is applied for designing codes with rate variable between 0.715 and 0.906. The designed codes are used in conjunction with M-QAM constellations through a pragmatic approach that, however, yields very promising results.**

***Index Terms*— LDPC codes, variable rate systems, puncturing, error rate performance**

I. INTRODUCTION

Low Density Parity Check (LDPC) Codes are the "state-of-the art" in the current scenario of error correcting codes. They have been introduced by Gallager in the sixties [1], but for a long period they suffered scarce interest. In recent years, however, many authors have shown how these codes can outperform other solutions in approaching the Shannon limit [2], when combined with efficient decoding algorithms based on 'belief propagation'. Since then, several methods have been proposed in the literature for the design of LDPC codes. At the same time, these codes have begun to appear in important telecommunication standards: a relevant example, in this sense, is provided by the proposal to include an LDPC code in the DVB (Digital Video Broadcasting) standard [3]. LDPC codes are currently under investigation also by CCSDS (Consultative Committee for Space Data Systems) for future space missions [4].

For a given code length n and rate R, it is not so difficult to design an LDPC code with good performance: it should have BER/FER (Bit Error Rate/Frame Error Rate) curves characterized by sudden "waterfall" (i.e. starting at small signal-to-noise ratios) and no error floor (or, at least, very low error floor, which means "negligible" for the application of interest). The point is that such a result could need rather complex solutions, often unpractical to implement. Thus, the goal is to design LDPC codes with (sometimes sub-)optimal performance but with limited complexity. The situation becomes more involved in those applications that require variable rate codes; this is typical, for example, in radio links, where the system throughput can be related to the channel conditions: less redundancy added when the channel is good, more redundancy added when the channel is bad. Radio links generally adopt M-ary modulation schemes, with the aim to limit the bandwidth occupancy, i.e., to have high spectral efficiency; then the code rate is adjusted in conjunction with the spectral features of the modulation adopted, in such a way as to satisfy the constraints on the bandwidth.

In a variable rate system using LDPC codes, a straightforward but inefficient solution may consist in using a different code for each required rate. This way, each code can be optimized, thus providing very good performance in any operational condition. As a drawback, however, a solution of this kind implies the need for a complete change of the co/decoding system any time updating the rate is requested. This, in turn, yields a proliferation of components, that is complex hardware and redundant processing time. Therefore, from this point of view, such a solution is quite unpractical.

A much more practical approach consists in using puncturing to adapt the rate. In this case only one (or a limited number of) code(s) is included in the system, with parameters suitable to ensure the lowest rate; the higher rates are then obtained by adopting a proper puncturing strategy. In its classic implementation, puncturing selectively eliminates a number of bits in the codewords produced by the code with the lowest rate (mother code). A puncturer device is added, for such purpose, at the encoder output: the value of n is reduced and the value of R is increased accordingly. The problem of puncturing is that it may require optimization and this is not, in general, a simple task, due to the number of degrees of freedom in the choice of the puncturing pattern. Moreover, again thinking in terms of classic implementation, the belief propagation algorithm, that operates on the Tanner graph representing the code, must be adapted to manage the missing bits at the receiver; this is usually done by assigning a null value to the Log-Likelihood Ratio (LLR) of these bits. Consequently, a significant performance degradation often results.

In this paper, we propose a different puncturing strategy where, instead of erasing bits, we eliminate rows and columns from the parity check matrix of the mother code; this permits to associate a reduced graph to the code with higher rate; such a graph is a subset of the original graph, and there is no need to force "artificial" values for the LLRs of the not transmitted bits. The resulting co/decoding system remains nearly as simple as that involved in classic puncturing, only

Manuscript received June 01, 2005; revised November 07, 2005 and January 09, 2006. The paper was presented in part at the Conference on Software, Telecommunications and Computer Networks (SoftCOM) 2005.

Authors are with the Dipartimento di Elettronica, Intelligenza Artificiale e Telecomunicazioni (DEIT), Università Politecnica delle Marche, via Brecce Bianche, 60131 Ancona, Italy, email: {m.baldi, f.chiaraluce}@univpm.it

1845-6421/05/5156

requiring the "switch-off" of some routing paths.

Performance evaluation of a number of codes with variable rate designed through this new puncturing strategy (called *pseudo-puncturing* in the following) is one of the targets of the present paper. As another target, however, we propose a new procedure, named Local Cycles Optimization (LCO), to design good LDPC codes without constraints on the code size and rate. As mentioned above, the design of good LDPC codes is an involved and, in some respects, still open problem. The "goodness" of a code is related to a number of aspects, often contrasting one each other, like encoding and decoding complexity, and error rate performance. Our design method, which is deterministic (not random), relies on the analysis and optimization of the local cycles length in the Tanner graph and gives the designer the chance to control complexity. Actually, the LCO algorithm will be presented in the first part of the paper, as it will be used in the second part, together with other well-known techniques like the Progressive Edge Growth (PEG) algorithm, to design variable rate codes.

The organization of the paper is as follows. In Section II we describe the LCO algorithm with examples of application. In Section III we discuss the way to match binary codes with M-ary modulations. Section IV introduces the question of variable rate codes design and its possible approaches, fixing attention, in Section V, on the pseudo-puncturing mechanism. Section VI deals with an explicit example of application, while section VII outlines other solutions found in the literature, that are compared with the proposed one. Finally, section VIII concludes the paper.

II. The LCO algorithm for the design of LDPC codes

A. General design issues

The design of an LDPC code consists in a suitable choice of its parity-check matrix. A designed parity-check matrix **H** is efficient if it is able to ensure good performance, yielding BER/FER curves characterized by sudden waterfall and very low error floor. Unfortunately, these two requirements are often contrasting. On one hand, it is known that the minimum distance of a code specified by its parity-check matrix corresponds to the smallest number of columns of **H** that sum to the null vector, i.e. the kernel rank, and this number generally increases when the column weight increases or, that is the same, when the matrix sparseness is reduced[1]. On the other hand, in most cases, a sudden start of the waterfall region would benefit by a more sparse matrix [5]. Searching for the best compromise is not simple, even because performance is determined by the whole degree distribution in the Tanner graph representing the code.

It is qualitatively reasonable that to have a small overlapping factor between the columns of **H** can help to increase the kernel rank. In this sense, there is a strict relationship with the need to avoid short length cycles in the Tanner graph of the code, that requires small overlapping as well, since short length cycles damage the efficiency of the 'belief propagation' decoding method. Moreover, it is well known that the local cycles minimum length determines a lower bound on the minimum distance [6]. All these aspects provide useful information, and should be taken into account in the arrangement of the parity check matrix.

Besides performance issues, the design of an LDPC code has to take into account the requirement to have limited encoding and decoding complexity. As an example, one of the most popular methods for LDPC encoding, which is based on Gaussian elimination followed by back-substitution, exhibits a complexity sometimes increasing as the square of the code size. This is due to the fact that Gaussian elimination, involving processing of the matrix rows and columns (instead of simple permutations) may be responsible for the reduction of the sparse character of matrix **H**. As a consequence, if encoding complexity must be a premium, encoding should be realized by avoiding Gaussian elimination, but using algorithms based only on rows and columns permutations which preserve the sparse character of the parity-check matrix [7]. In some cases, these techniques lead to a complexity that increases linearly with the code dimension. Further results in the same direction have been obtained by introducing the so-called LU factorization technique to increase the efficiency of the encoding stage [8].

An approach that surely leads to an encoding complexity linearly increasing with the code dimension consists in designing directly lower triangular matrices: in this case the sparse character is always preserved in the encoding stage too. A way of pursuing this aim is to divide the parity-check matrix in two blocks, one random and one deterministic, the latter having lower triangular form. Such an approach has been first introduced with the so-called 'semi-random codes' [9].

Based on these premises, in the next subsection we present a 'fully' deterministic technique to construct LDPC matrices, which is able to consider all the needs described above. The proposed algorithm permits to design codes without constraints on the dimension or the rate.

B. The LCO algorithm

The core of the LCO algorithm is a set of procedures which serve to compute the local cycles length in the Tanner graph of a code. This knowledge is necessary to estimate the length of the shortest cycle for each variable node, also known as the local 'girth'.

It can be noticed that, whilst in the PEG algorithm, described in [6], edges are inserted in the Tanner graph through a 'best effort' approach, the LCO method uses a 'brute force' procedure, more similar to the one presented in the 'look-ahead-enhanced version' of the PEG algorithm, recently proposed [10]. In spite of this, however, the code construction always employs reasonable computation time, even for codeword lengths as large as 10k bits. Furthermore, employing such a technique, the code design is straightforward: in most cases only one attempt is needed to design a good code, so the computation overload due to the brute-force approach can be justified considering that the algorithm is executed only once.

[1]This holds on condition that the number of symbols '1' in each column of the parity-check matrix remains less than the number of symbols '0', but this is certainly true for LDPC codes

Algorithm 1 Local Cycles Optimization

initialize the parity check matrix with all symbols '0'
do the first matching by inserting multiple diagonals
for $(k = 1; k < d_{vmax}; k++)$
  for $(i = 1; i \leq n; i++)$
    if $(d_v[i] < d_{vtarget}[i])$
      set the value of j_{min} to consider the possible constraint on the lower triangular form for the generated matrix
      for $(j = j_{min}; j \leq r; j++)$
        if $(\nexists\ edge(i,j))$ **and** $(d_c[j] < d_{ctarget}[j])$
          simulate $edge(i,j)$
        end if
      end for
      if $\exists\ j$ that satisfies the girth length constraint
        find j_{best} and **assign** $edge(i, j_{best})$
      end if
    end if
  end for
end for

The deterministic approach, here adopted, permits to maintain a complete control on the girths, contrary to what may happen when using a random approach. A routine implementing the LCO is represented by Algorithm 1.

The notation used needs explanation: d_{vmax} is the variable nodes maximum degree, n is the number of columns in the parity check matrix, $d_v[1 \ldots n]$ is the current column weights vector, $d_{vtarget}[1 \ldots n]$ is the target column weights vector, r is the number of rows in the matrix, $d_c[1 \ldots r]$ is the current row weights vector, $d_{ctarget}[1 \ldots r]$ is the target row weights vector, $edge(i,j)$ represents an edge connecting the variable node i with the check node j (i.e. setting to 1 the (j,i)-th symbol in the matrix). $d_{vtarget}[1 \ldots n]$ and $d_{ctarget}[1 \ldots r]$ are fixed in advance, according with the total edge number to be inserted or with specific degree distributions the code designer would like to impose.

When the algorithm starts, the graph is initialized with the insertion of an edge for each variable node, which is equivalent to put multiple diagonals in the matrix. This task is performed in such a way that a whole diagonal is positioned in the right part of the matrix, to ensure the lower triangular form. As an option, the last diagonal can be doubled, to increase the initial column degree of the right square block, similarly to what done in [9].

Actually, for increasing flexibility, the choice to have a lower triangular form is not obliged: by a suitable choice of the parameter j_{min}, one can decide to insert symbols '1' above the rightmost diagonal, or not. For the former case, in particular, it is sufficient to set $j_{min} = 1$; otherwise, j_{min} is properly calculated by the algorithm.

A key point concerns finding j_{best}, that is the index of the most suitable control node to connect with the variable node v_i. This is done through the Algorithm 2.

The algorithm has to establish if the girth length constraint is satisfied or not. For this purpose, a scanning procedure is used which recursively runs through the Tanner graph and reports the length of the local cycles. The local girth length found must be higher than a threshold, imposed by the designer, for the constraint to be satisfied, otherwise the edge is not inserted.

Algorithm 2 Evaluation of j_{best}

given i as the variable node under analysis
considering $j \in [j_{min}, r]$
find $(j : edge(i,j)$ does not create a local cycle) **and** $(d_c[j]$ is minimum)
if found
  $j_{best} = j$
else
  find $j : edge(i,j)$ creates the longest local cycle
  if $edge(i,j)$ creates a local cycle that satisfies the girth length constraint
    $j_{best} = j$
  else
    $j_{best} = 0$
  end if
end if

C. Code Examples

Nine LDPC codes, with $n = 2640$ and $R = 1/2$, have been designed by using the LCO technique with different initial settings; four of them have full (in the sense of non-triangular) parity-check matrices and the other five have lower triangular parity-check matrices. For the sake of comparison, two further codes, designed by using different techniques but with the same parameters, have been considered: the first one was obtained by using the PEG algorithm [6], while the second one is a Margulis code [11], well recognized as a valuable benchmark in the literature. The number of edges and the local cycles length distribution, for each code considered, are shown in Table I where, for subsequent discussion, the codes have been properly numbered, and LCO codes with lower triangular parity-check matrices (LT) are distinguished from those with full matrices (F).

The performance of the whole set of codes has been compared according with the criteria explained in the following subsection. Some results of this comparison are then reported and discussed in subsection II-E.

D. Comparison Criteria

Three parameters will be evaluated for each considered code: encoding complexity, decoding complexity and error rate performance. Based on these parameters the choice of a code should be always made, however, safeguarding the peculiarities of any specific application.

1) Encoding Complexity: Encoding complexity obviously depends on the encoding technique adopted. Since none of the codes considered allows particular encoding procedures (as quasi cyclic codes do, for example), we will refer to the common 'back substitution' technique, together with 'Gaussian elimination', if necessary.

TABLE I
COMPARISON OF LDPC CODES WITH $n = 2640$ AND $R = 1/2$

Code	Design	No. of edges	Number of Local Cycles with Length: 8	10	12	14	≥ 16
C_1	LCO (LT)	7061	0	0	2639	0	1
C_2	LCO (LT)	7127	0	0	2627	0	13
C_3	LCO (F)	7141	0	0	2627	0	13
C_4	LCO (F)	7220	0	0	2640	0	0
C_5	LCO (F)	7229	0	0	2640	0	0
C_6	LCO (LT)	7480	0	2218	421	0	1
C_7	LCO (F)	7920	0	2640	0	0	0
C_8	LCO (LT)	13005	2639	0	0	0	1
C_9	LCO (LT)	13534	2639	0	0	0	1
C_{10}	PEG	7899	0	2631	0	0	9
C_{11}	Margulis	7920	2640	0	0	0	0

In this case, there is a considerable difference between the encoding stage of a code characterized by a full parity check matrix and that of a code with a lower triangular matrix: the former requires Gaussian elimination to be performed on its parity check matrix, that usually implies the loss of the sparseness character of the matrix itself, while the latter permits to bypass this process thus preserving the matrix sparseness.

In the worst cases, that however are not so rare, we can presume that, after application of the Gaussian elimination, matrix **H** becomes dense. This has implications on the complexity, as explained below.

To estimate the encoding complexity, we can consider the total number of sums needed to solve the linear system of parity check equations. In evaluating this quantity, we refer to lower triangular matrices, either those designed just in this form or those obtained by Gaussian elimination. It can be easily verified that, in both cases, the total number of sums equals the number of symbols '1' in the matrix minus its rows number, each parity equation providing one redundancy bit.

For dense lower triangular matrices, we can consider that, except for the rightmost all-zero triangle, they have approximately half the symbols equal to '1' and half equal to '0'; in this case, we denote by *ECD* the encoding complexity (*EC*) of a codeword given by

$$\begin{aligned} ECD &= \frac{1}{2}\frac{(n+k)\cdot r}{2} - r = \\ &= \frac{n^2}{4}\left(1-R^2\right) - n\left(1-R\right) \end{aligned} \quad (1)$$

where the relationships $r = n \cdot (1 - R)$ and $k = n \cdot R$ have been used to express the information size k and the redundancy size r in terms of the codeword size n and the code rate R. It follows immediately that the encoding complexity, in this case, increases as $O\left(n^2\right)$.

For sparse lower triangular matrices, the number of symbols '1' in the matrix simply coincides with the number of edges E in the Tanner graph; so the following equation results, where *ECS* is the encoding complexity for the sparse case:

$$\begin{aligned} ECS &= E - r = \\ &= n \cdot d_{vaverage} - n\left(1-R\right) = \\ &= n\left(d_{vaverage} + R - 1\right) \end{aligned} \quad (2)$$

In Eq. (2), we have considered that $E = n \cdot d_{vaverage}$, where $d_{vaverage}$ represents the average number of symbols '1' in each column of the matrix. $d_{vaverage}$ is always low (typically less than 6, for the code length here considered) and does not depend on n or, more precisely, depends on n in a sub-linear way; so we can say that the encoding complexity, in this case, increases as $O\left(n\right)$.

2) Decoding Complexity: Although exact evaluation of hardware/software decoding complexity for LDPC implementation strongly depends on advanced design issues like the parallelization degree, the routing strategies and the memory requirements, a complexity measure which is independent of the final implementation and significant enough for fair comparison between different codes is needed.

It can be proved that some important features, like the memory requirements, the number of elementary operations at each variable node for each iteration and the number of elementary operations at each check node for each iteration, are all proportional to the number of edges E in the Tanner graph. Moreover, complexity should be referred to each decoded information bit; this implies that E is multiplied by the number of iterations and then divided by k, the information bits per decoded codeword.

In conclusion, we can express the decoding complexity per information bit as follows

$$DC = \frac{I_{MAX} \cdot E}{k} \quad (3)$$

where I_{MAX} represents the maximum number of iterations scheduled for decoding.

3) Error Rate Performance: A further criterion we have considered is the code performance over the AWGN (Additive White Gaussian Noise) channel. For this purpose, all codes have been simulated by means of a software written in C++ language and their performance has been analyzed in terms of Bit Error Rate (BER) and Frame Error Rate (FER) curves. The modulation scheme adopted is Binary Phase Shift Keying.

Figs. 1 and 2 show the performance of LCO codes with a full parity-check matrix, whilst Figs. 3 and 4 show the performance of LCO codes with a lower triangular parity check matrix. All figures also report, for the sake of comparison, the performance of the PEG code and the Margulis code.

For an immediate comparison, we can fix (at least) one target FER value and consider the signal-to-noise ratio it requires for different codes. A typical working point is FER $= 10^{-4}$, so that we have compared the values of $\widetilde{E_b/N_0} = \left(E_b/N_0 \,\text{at FER} = 10^{-4}\right)$

Some applications may require to work at lower FER values, so a lower target should be chosen as well. Some codes could require too high signal levels for reaching the target, because of the appearance of an error floor. For the specific application, the use of these codes should be obviously avoided.

E. Codes Comparison

The comparison criteria defined in the previous subsection have been applied to the considered codes, and the results obtained are summarized in Table II. For the Margulis code, the value of I_{MAX} is not available.

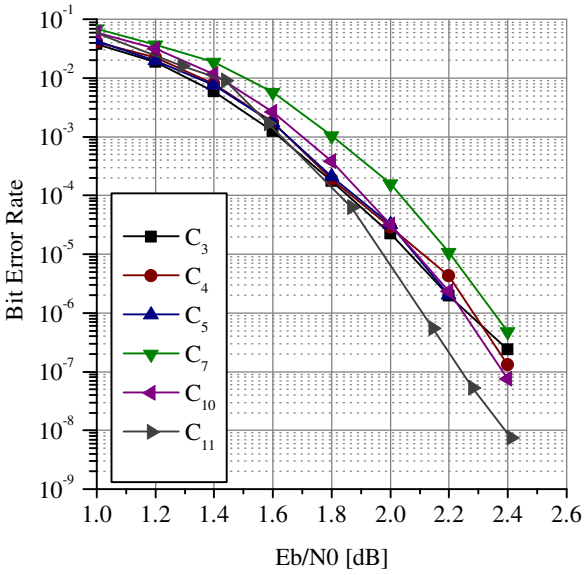


Fig. 1. BER curves for the codes generated with a full parity-check matrix

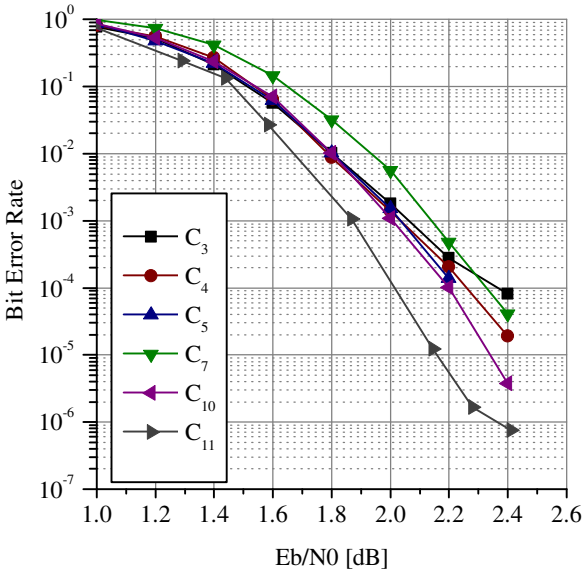


Fig. 2. FER curves for the codes generated with a full parity-check matrix

TABLE II
CONSIDERED CODES COMPARISON

Code	Design	EC	I_{MAX}	DC	$\widetilde{E_b/N_0}$
C_1	LCO (LT)	5741	30	160.5	2.35 dB
C_2	LCO (LT)	5807	30	162.0	2.50 dB
C_3	LCO (F)	1305480	30	162.3	2.35 dB
C_4	LCO (F)	1305480	30	164.1	2.25 dB
C_5	LCO (F)	1305480	30	164.3	2.25 dB
C_6	LCO (LT)	6160	30	170.0	2.60 dB
C_7	LCO (F)	1305480	30	180.0	2.30 dB
C_8	LCO (LT)	11685	30	295.6	2.85 dB
C_9	LCO (LT)	12214	30	307.6	2.95 dB
C_{10}	PEG	1305480	30	179.5	2.20 dB
C_{11}	Margulis	1305480	-	-	2.05 dB

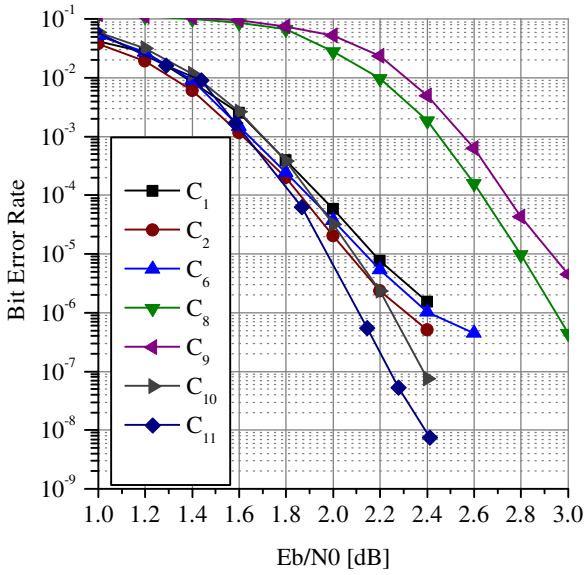


Fig. 3. BER curves for the codes generated with a lower triangular parity-check matrix

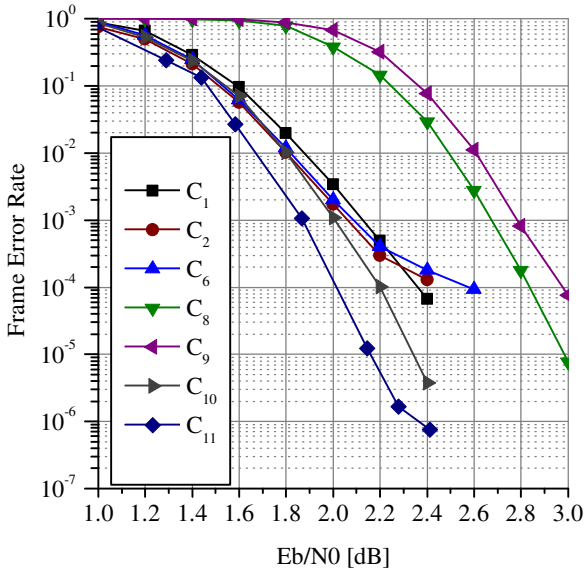


Fig. 4. FER curves for the codes generated with a lower triangular parity-check matrix

As probably expected, the results demonstrate that none of the codes can be considered “the best” in absolute terms. For the sake of clarity, the results of Table II are also reported in pictorial form, in the histograms of Fig. 5 (each expressed as a percentage of the corresponding maximum value).

The minimum signal-to-noise ratio is ensured by the Margulis code that, however, has the maximum encoding complexity (together with other codes). On the other hand, the superiority of the Margulis code may become questionable for lower error rates because of the appearance of an error floor (see Figs. 2 and 4).

The codes designed with the LCO method exhibit a rather low decoding complexity (with the exception of codes 8 and 9)

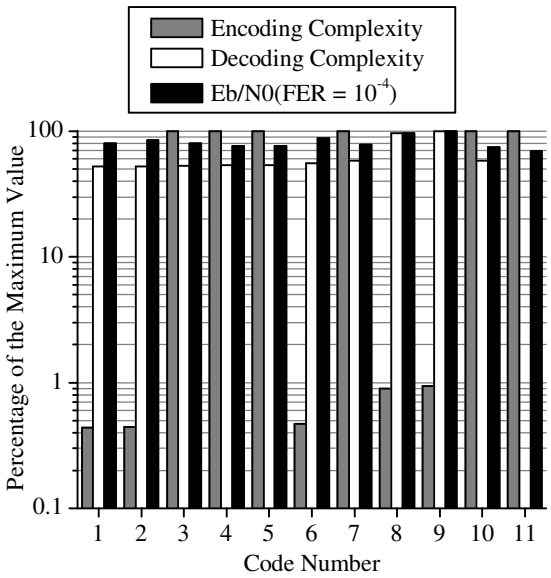


Fig. 5. Codes comparison

and a low encoding complexity when the parity-check matrix is designed directly in the lower triangular form.

This is probably one of the main advantages of the LCO method: the possibility to design a code with a parity-check matrix suitable for easy encoding and with acceptable, though non-optimum, error rate performance. Code C_1, in particular, seems a good result: it has the minimum value of *EC* and *DC*, but remains at 0.3 dB from the minimum E_b/N_0 found for the Margulis code.

III. Application to M-ary modulation schemes

The LCO algorithm presented in the previous section has been used to design binary codes, which are "naturally" suited for binary modulation schemes. On the other hand, in many important applications, M-ary modulation schemes are preferred to increase the spectral efficiency. Consequently, in these schemes, there is the need to match encoding and modulation. In principle, this problem is conceptually not simple as one should take into account that performance, in an M-ary scheme, is dominated by the Euclidean distances, which may not be proportional to the Hamming distances. As a consequence, the best binary code could not lead to the best code over the M-ary constellation, and code optimization should be done directly in the multilevel domain [12]. In [13], for example, an interesting method has been proposed to design codes, directly over the group of integers modulo 8, that is Z_8, suitable for applications to 8-PSK constellations. In this paper, however, we consider higher order modulations, suitable for applications requiring very large spectral efficiency. For high order modulations, it is known that M-QAM is generally preferred to M-PSK because of its more favorable error rate performance. M-QAM is different from M-PSK: for both constellations the Euclidean distance is not proportional to the Hamming distance but, whilst for M-PSK, with $M > 4$, the constellation is not matched to a binary group, for M-QAM

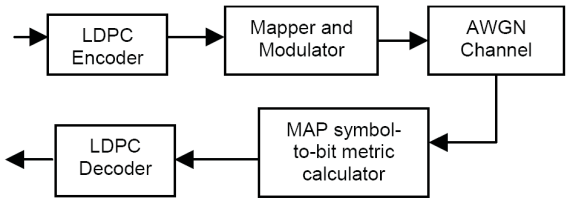


Fig. 6. Block diagram of the LDPC-coded scheme adopted in conjunction with $M-$QAM constellation

the constellation is matched to a binary group, at least up to the boundary effects. This has implications on the design, where the impact of the more rigorous approach is expected to be lower (for example there is no need to build a group code over Z_M, the binary LDPC code generating a geometrically uniform Euclidean space code by itself). Moreover, decoding complexity in the multilevel domain is higher than for binary codes.

For these reasons, we have adopted a more pragmatic approach, based on the scheme shown in Fig. 6. Looking at the figure, we can say that, whilst the transmitter blocks are rather obvious, with the mapper and modulator block that maps groups of $m = \log_2 M$ code bits into a symbol of the bi-dimensional constellation (more will be said in the following as regards the labeling strategy adopted), a fundamental element at the receiving side is the symbol-to-bit metric calculator that computes the LLRs according with a maximum a posteriori (MAP) criterion. The LLR of the coded bit x_i, given the received sample $\mathbf{y}$, is given by [14]:

$$L(x_i) = \ln\left[\frac{\sum_{\mathbf{v}\epsilon\chi_0^i} \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{y}-\mathbf{v}\|^2\right)}{\sum_{\mathbf{w}\epsilon\chi_1^i} \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{y}-\mathbf{w}\|^2\right)}\right] \qquad (4)$$

where $\mathbf{v}$ and $\mathbf{w}$ are two complex symbols of the constellation (i.e. bi-dimensional vectors), σ^2 is the variance of the thermal noise, and χ_b^i is the subset of signals whose label has the value $b\,\epsilon\,\{0,1\}$ at the i-th position. Starting from (4), decoding proceeds, in the LDPC decoder, through an iterative exchange of messages between the variable and check nodes of the Tanner graph representing the code (belief propagation).

The efficiency of this LDPC coded modulation scheme has been already tested in previous literature, in conjunction with Gray labeling. Labeling is the rule by which sequences of bits are univocally assigned to the symbols. Gray labeling is asymptotically optimal in bit-interleaved coded-modulation [15]. It can be used also for the present system where, as we have verified through simulation, it permits to achieve performance better than that obtained by using other labelings, like rotationally invariant labelings. On the other hand, it is well known that Gray labeling is not possible when the number of symbols in the constellation is an odd power of two. This occurs, for example, in the 32-QAM and 128-QAM. In this case we have adopted a quasi-Gray labeling, actually derived through the more general theory of quasi symmetric ultracomposite (SU) labeling [16]. An example of SU labeling for the case of 32-QAM is reported in Fig. 7: it shows minimum deviation from the Gray labeling (limited to the corners for the considered example).

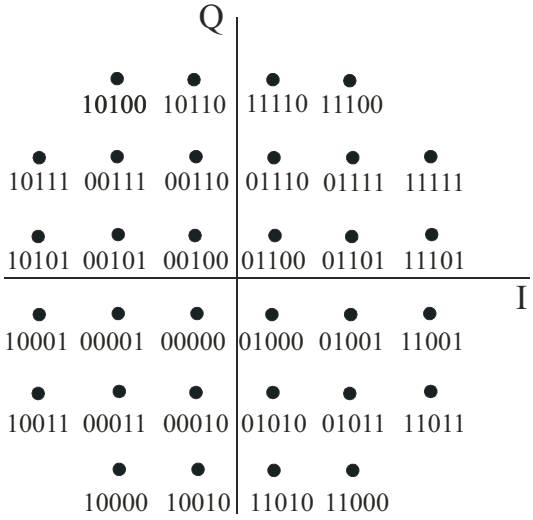


Fig. 7. Example of Quasi SU labeling for the 32-QAM constellation

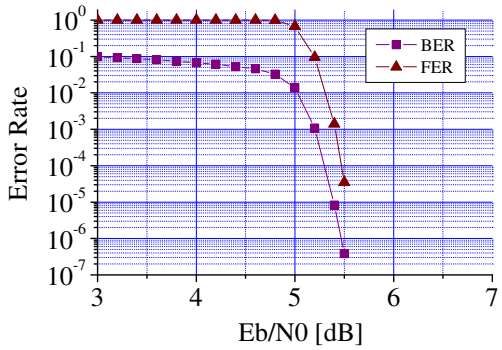


Fig. 8. Example of BER/FER performance for an LDPC code with rate 0.715 matched with the 16-QAM constellation

An example of the performance achievable by using the considered scheme, for an LDPC code with $k = 9720$ and $n = 13600$, matched with a 16-QAM Gray labeled constellation, is shown in Fig. 8. The Tanner graph of the code has $E = 39911$ edges, almost regularly distributed, and a symbols '1' density, defined as the ratio between the number of symbols '1' and the overall number of symbols in the parity check matrix of the code, $d = 3 \cdot 10^{-4}$. As well known, the value of the matrix density d plays a crucial role in fixing the performance of an LDPC code: on one hand, it should be high to have low error floor; on the other hand, it should be low to have sudden water-fall and good performance for small signal-to-noise ratios [5].

The simulated curves have been obtained by assuming a maximum number of iterations $I_{MAX} = 30$ in the decoding process, while reliability of the results has been ensured by the simulation of at least 100 wrong frames as a stop criterion for each point estimation. The same assumptions hold for the results presented in the subsequent sections. To have an idea of the goodness of the results reported in Fig. 8, one should note that a competitor Trellis Code Modulation + Reed Solomon (TCM+RS) system, characterized by the same rate (though with codewords five times shorter) requires a signal-to-noise ratio more than 1.5 dB larger at BER = 10^{-6}.

IV. Design of codes with different rates

Let us suppose that the system must operate with different rates in such a way as to adapt the code and the modulation scheme to the transmission conditions. In this paper we consider code rates ranging between $R_{min} = 0.715$, for a system operating with 16-QAM, and $R_{max} = 0.906$, for a system operating with 128-QAM, and assume several intermediate values. The corresponding spectral efficiency ranges between $\eta_{min} = 2.860$ and $\eta_{max} = 6.342$. Thus, a non trivial problem concerns the choice of the strategy of managing such a widespread range of possibilities.

The most "obvious" solution should consist in designing as many LDPC codes as the number of rates of interest. This way, an optimum code could be found for any operation condition. Such "ad hoc" design strategy, however, is quite unpractical: it requires to change the code any time the rate changes, with a questionable excess of memory occupancy and processing time.

On the contrary, one could aim at using a single co/decoding scheme that adapts itself to the variable operation condition with minimum software/hardware requirements. This apparently ambitious result can been achieved by puncturing. In its classic form, puncturing consists in using only one parity check matrix to produce codewords for all rates of interest. This matrix, denoted as $\mathbf{H_m}$ in the following, has n_m columns and $r_m = n_m - k_m$ rows, where $k_m/n_m = R_{min}$; therefore it is designed for the code with the lowest rate in the specified range: we call this code "mother code" for the considered set. Typically, in the variable rate scheme, the value of k is fixed (that is, $k = k_m$ for all rates); we increase the rate by puncturing a prefixed number of bits (usually, though not necessarily, in the parity check part) of the codewords produced by the mother code. Therefore, the peculiarity of this scheme is that it does not require any architectural updating at the encoder, to face the variable rate, except for the addition of a simple puncturer before the mapper and modulator. At the decoding side, the Tanner graph of the punctured code can be assumed to be identical to that of the mother code; the LLRs of the punctured bits are conventionally set to 0 and, after such initial assignment, decoding proceeds exactly in the same way, independently of the rate value.

Puncturing has been extensively studied in conjunction with many iteratively decoded concatenated codes [17] and it is known it often yields remarkable performance worsening. The main difficulty of puncturing is the huge number of degrees of freedom it implies, for example in regard to the possible puncturing pattern configurations (which is huge as well for non trivial codes). Therefore, classic puncturing of LDPC codes requires optimization, and the development of suitable techniques, for example based on the density evolution method, which permits to avoid testing of all (or a significant proportion of all) possible patterns [18].

In this paper, however, we have considered a modified form of puncturing that has shown much less dependence on the chosen pattern; this modified form has been called "pseudo-puncturing" and is presented in the next section. In Section VI,

instead, some examples are given of codes designed according with this strategy.

V. The pseudo-puncturing mechanism

Pseudo-puncturing consists in eliminating a number of columns, in the right ($r_m \times r_m$) part of the parity check matrix $\mathbf{H_m}$ of the mother code, and a number of rows, both equal to the difference between the codeword length of the mother code and that of the pseudo-punctured one. Columns elimination is equivalent to scale the number of parity bits, whilst rows elimination corresponds to discard parity check equations.

The parity check matrix of the mother code is preferably designed in a lower triangular form: this choice simplifies encoding, which can be done by means of a back substitution procedure. Wishing to have a lower triangular form for the punctured matrix too, one can choose to eliminate pairs of rows and columns that intersect along the rightmost diagonal.

At the decoder side, the Tanner graph of the pseudo-punctured code, compared with that of the mother code, has less variable nodes and less check nodes. This, however, does not imply any architectural modification, but only a "switch-off" of the edges in the Tanner graph connecting punctured nodes. In practice, some messages exchanged between the nodes of the mother code are no longer present while, contrary to the case of classic puncturing, there is no need to force an artificial value for the missing log-likelihood ratios. Moreover, the decoder has to know only the position of the missing edges (that is an information which can be easily derived from the knowledge of the Tanner graph and the pseudo-puncturing pattern).

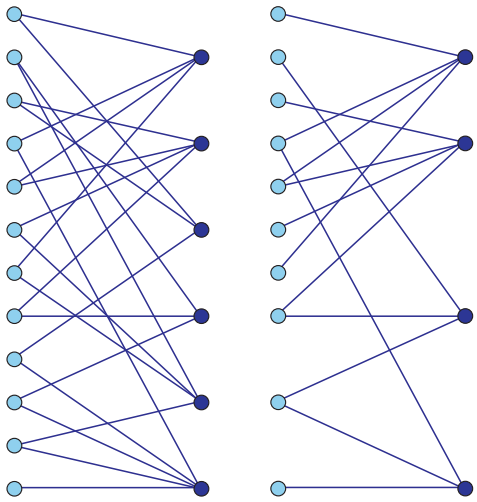
For better understanding the proposed procedure, let us consider a very simple example: the following matrix (5) obviously does not refer to an LDPC code, but it is satisfactory for explicative purposes:

$$\mathbf{H_m} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \quad (5)$$

In spite of its simplicity, matrix (5) is not trivial: the corresponding Tanner graph, shown in Fig. 9 (a), has no 4-length cycles (as the consequence of a maximum overlap between columns equal to 1), while the matrix is in lower triangular form.

The code defined by (5) has $n_m = 12$ and $k_m = 6$ (rate 1/2); starting from it, we can construct a code with $n = 10$ and $k = 6$ (rate 3/5) by eliminating, for example, the ninth and eleventh column and, consequently, the third and fifth row. This way, the new matrix is:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \quad (6)$$



(a) Mother (b) Pseudo-puncutred

Fig. 9. Tanner graphs of the two codes: the mother code with matrix (5) and the pseudo-punctured code with matrix (6)

and the Tanner graph becomes that shown in Fig. 9 (b): nodes and edges are a subset of those in Fig. 9 (a), which confirms the absence of any architectural modification. Like the original graph, also the reduced one has no 4-length cycles; moreover, $\mathbf{H}$ maintains the lower triangular structure. These properties hold regardless of the code size and, therefore, can be extended to larger (and more significant) codes as well. In the next section, we provide examples of long codes designed by pseudo-puncturing.

VI. Examples of codes designed by pseudo-puncturing

We have already stressed in Section III that a good choice of the matrix density d is essential to obtain a good compromise between the waterfall and the error floor behavior. When designing punctured parity check matrixes (in the sense specified in the previous section) for a wide range of rates, this may be a problem: if the mother code has an almost regular matrix (which is the result of the algorithm we have adopted for the codes design), the value of d does not change significantly because of pseudo-puncturing. In other words, we can say that pseudo-puncturing reduces linearly the number of symbols '1' in the parity check matrix (or, that is the same, the number of edges in the Tanner graph).

Due to the constraint on the value of d, in order to obtain pseudo-punctured codes with good performance, it is necessary to start from a mother code whose parity check matrix $\mathbf{H_m}$ has a rather large number of symbols '1' (compared to its "optimum" value); this way, the pseudo-punctured matrixes can have a not too low number of symbols '1' in their turn (that would have favorable effects on the waterfall but catastrophic effects on the error floor).

In the following of this section, we present a practical example of codes designed through the pseudo-puncturing mechanism, for the previously specified rate values range, and matched with QAM constellations with different cardinalities.

TABLE III
EXAMPLES OF CODES WITH $k = 9720$, DESIGNED BY PSEUDO-PUNCTURING

Code	n	R	E	d	QAM	G
C_{12}	13600	0.715	249569	$4.73 \cdot 10^{-3}$	16	-1 dB
C_{13}	12750	0.762	186669	$4.83 \cdot 10^{-3}$	32	-0.9 dB
C_{14}	11656	0.834	111308	$4.93 \cdot 10^{-3}$	16	$+0.8$ dB
C_{15}	11330	0.858	90779	$4.98 \cdot 10^{-3}$	32	$+0.7$ dB
C_{16}	10724	0.906	54646	$5.07 \cdot 10^{-3}$	128	$+1.5$ dB

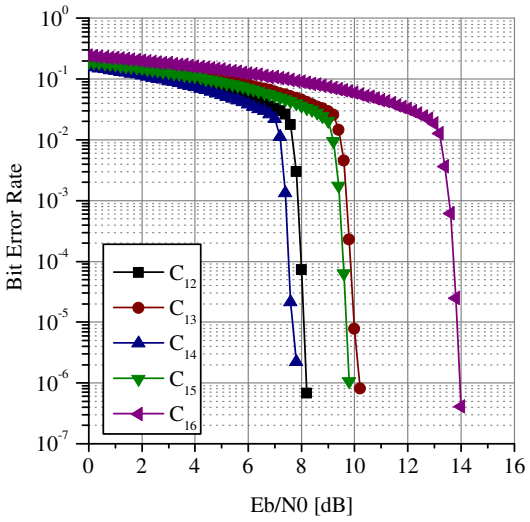


Fig. 10. BER performance of the codes in Table III

The parameters of the considered systems are detailed in Table III. All codes have $k = 9720$; the table specifies also the rate R, the number of edges E and the symbols '1' density d. The last column of the matrix reports the additional coding gain G that the system based on LDPC codes permits to obtain with respect to a TCM+RS system with the same rate, at BER = 10^{-6}. In particular, codes C_{12} and C_{13} are compared with TCM 2D, while codes C_{14}, C_{15} and C_{16} with TCM 4D. The TCM codes are based on convolutional codes with rate $2/3$. C_{12} is the mother code, whose parity check matrix has been punctured with rather arbitrary and simple rules (for example, progressively eliminating the columns at the rightmost side and the rows at the bottom of $\mathbf{H_m}$). In other words, the puncturing pattern has not been optimized, deliberately. In spite of this, the comparison with the TCM+RS system (which is an optimized scheme) shows improvements immediately: for the 128-QAM constellation, the system based on the LDPC code has an additional coding gain $G = 1.5$ dB with respect to the competing TCM+RS system. Thus, it is evident that the codes design, in this specific example, has privileged performance for high order constellations. This is clearly explained by the large number of edges that code C_{12} exhibits, for example in comparison with the analogous code discussed in Section III and whose performance has been plotted in Fig. 8. The latter has a smaller number of edges (or symbols '1' density) and this reflects on a more favorable start of the waterfall region.

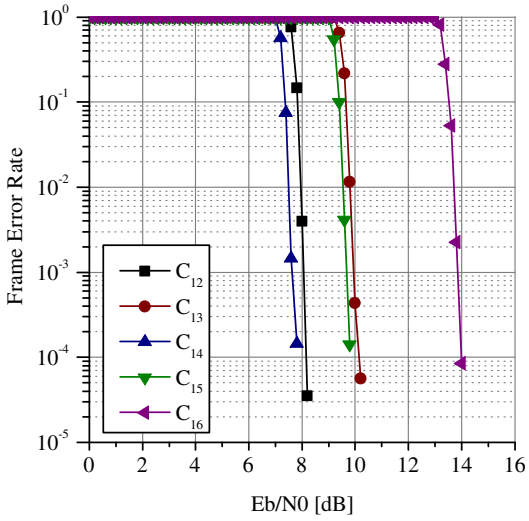


Fig. 11. FER performance of the codes in Table III

The BER and FER curves for codes C_{12}-C_{16} are plotted in Fig. 10 and Fig. 11, respectively. These curves confirm the above considerations: performance of codes C_{12} and C_{13} is worse than expected (and in fact, in Table III, their coding gain G is negative, which means that TCM+RS system is better): their design has been conditioned by the goal to improve performance of codes C_{14}, C_{15} and, most of all, C_{16}, characterized by the highest rates. Anyway, even under such constraint, the performance achieved seems good for all rates, with prompt waterfall and no error floor in the explored region.

Discussion above is based on the coding gain G at BER = 10^{-6}. Anyway, in our simulations we have found that the TCM+RS curves and the LDPC curves have comparable slopes in the explored region of signal-to-noise ratios; this means that the value of the gain remains approximately the same also in the neighborhood of BER = 10^{-6} (for example, up to 10^{-7}).

Aiming at further improving performance in a so wide range of values for R (between 0.715 and 0.906, in the proposed example) it could be convenient to use more than just one mother code; two mother codes, for example, should be a reasonable compromise: the former for the lowest rates and the latter for the highest ones. This way, the constraints in fixing the structure of the parity check matrix for the mother code could be somewhat relaxed and a range of R as large as that considered in our simulations could be covered with very high efficiency.

VII. OUTLINE OF DIFFERENT PUNCTURING STRATEGIES

In this paragraph we present a brief overview of the most significant approaches proposed in previous literature to design rate compatible codes, aiming at emphasizing the novelties and discussing the advantages offered by the pseudo-puncturing solution here proposed.

As stressed in the previous sections, to find an optimum puncturing pattern is usually a very difficult task, quite unfeasible to face in analytical terms. In the case of classic

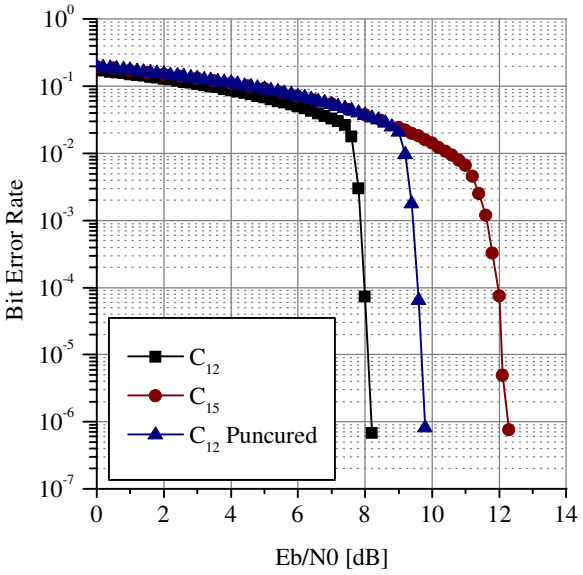


Fig. 12. BER comparison between punctured and pseudo-punctured codes

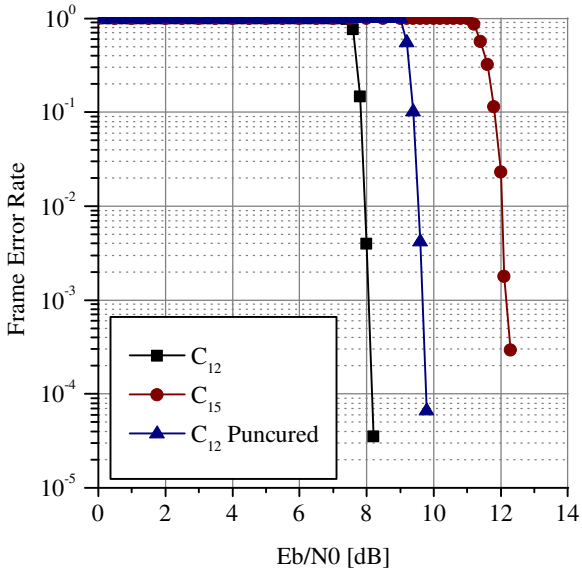


Fig. 13. FER comparison between punctured and pseudo-punctured codes

puncturing, a strategy often pursued in previous literature is that of a random choice of the punctured bits. Random puncturing is very simple to implement (through the adoption of elementary algorithms for random generation of a uniformly distributed integer variable) and does not need, in principle, any optimization. However, such a strategy can yield, at random, quite unsatisfactory results as well, so that it seems advisable to limit somehow the choice, in order to avoid those patterns that will certainly lead to bad performance. A first rule in this sense consists in preventing inclusion, among the punctured bits, of a whole stopping set; otherwise, the decoding algorithm is not able to recover the information relative to those nodes. In the case of lower triangular parity check matrixes, the rule is satisfied by limiting puncturing to the last r bits of the codeword [19], i.e. electing for puncturing the columns in the rightmost side. Any column subset of the triangular portion, in fact, is not a stopping set, because the node corresponding to the leftmost column of this subset has at least one neighbor (due to the topmost '1') that is singly connected to the set.

We have applied such a strategy to code C_{12} in Table III, by puncturing 2270 bits in such a way as to obtain a code with rate R = 0.858, and compared its performance, at a parity of I_{MAX}, with code C_{15} in the table, obtained through pseudo-puncturing. The comparison is shown in Fig. 12 for the BER and in Fig. 13 for the FER. The performance of C_{12} is also shown as a reference. Looking at the figures, it is clear that the pseudo-puncturing strategy outperforms the classic one with constrained random choice of the punctured bits, with more than 2 dB of additional coding gain for this specific example. Moreover, no error floor appears in the pseudo-punctured code curves.

McLaughlin et al. have applied the density evolution algorithm to find optimized puncturing patterns for LDPC codes with rate $0.5 \leq R \leq 0.9$. Though the conceptual value of density evolution is highly recognized, its practical impact may be questionable: basically, it provides asymptotic results that do not match, necessarily, with the real code behavior. In [18], for example, a mother code with block length $n = 131072$ and $R = 0.5$ was designed and properly punctured up to $R = 0.95$; the performance obtained was very good but rather far from the theoretical result, that therefore represents a lower bound, especially for the highest rates. The designed code was irregular, while no discussion was made about encoding complexity. In most applications, the codeword length must be significantly smaller (in radio links, for example, where it is necessary to control delay [20]) whilst to limit encoding complexity is often a mandatory task. The codes we have considered in this paper are about one order of magnitude shorter than that in [18]; at the same time, the parity check matrixes have been designed through criteria that aim at maximizing the girth length in the Tanner graph (so increasing the minimum distance), maintaining almost constant the weight of the parity check matrix columns. This, combined with the lower triangular form, that is achieved at no additional cost, makes encoding of pseudo-punctured codes particularly easy. In [21] the same approach of [18], called intentional puncturing, was applied to shorter codes, but in this case not achieving an equally favorable result: the proposed punctured LDPC code had performance better than that of the randomly punctured code but worse than that of an "ad hoc" design, at least for the medium/low error rates of interest. On the contrary, we have verified, through several numerical examples, that pseudo-punctured codes can be designed, rate-by-rate, that are able to ensure the same performance of "ad hoc" codes, just from the neighborhood of the waterfall knee.

An example is shown in Fig. 14, for the BER, and in Fig. 15, for the FER: the performance of code C_{16}, obtained by pseudo-puncturing, is comparable with that offered by an "ad hoc" code that we have designed for the same rate. This

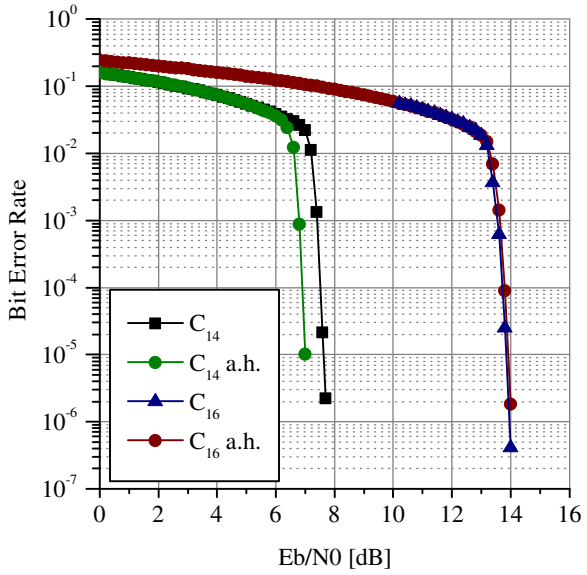


Fig. 14. BER comparison between "ad hoc" (a.h.) and pseudo-punctured codes

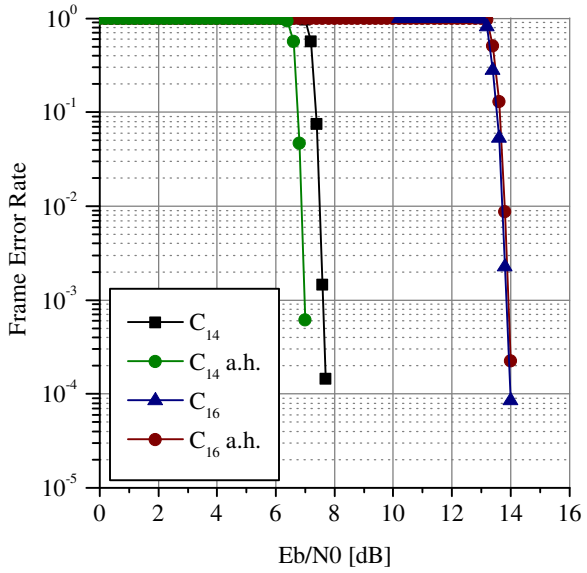


Fig. 15. FER comparison between "ad hoc" (a.h.) and pseudo-punctured codes

demonstrates that pseudo-puncturing may be lossless, in the sense that, for a specific rate, it is not difficult to construct, through a suitable choice of the mother code parameters, a pseudo-punctured code that behaves like a dedicated code. Obviously, this does not mean that, for the same mother code, such an excellent performance is retained for the other rates: in Figs. 14 and 15, for example, code C_{14} shows a penalty with respect to an "ad hoc" code with the same rate ($R = 0.834$).

The density evolution algorithm for punctured codes has been improved in [22] and [23], and its application to find good puncturing patterns has been made even easier. Anyway, best results were found for the binary erasure channel (BEC), while other channels (like the AWGN channel, here of interest) require some form of further refinement and optimization. Thus, the problem of reducing the penalty induced by puncturing in conventional channels remains, and the proposal of alternative approaches, like the pseudo-puncturing here considered, appears justified anyway.

In [19], the density evolution approach has been applied in conjunction with information nulling and parity puncturing techniques. This is a possible answer to the problem claimed in Section VI of obtaining good punctured codes for a wide range of rate values: the designed mother code has an intermediate rate, from which lower rates can be achieved by nulling (i.e., shortening the code) and higher rates by puncturing.

A similar idea justifies the proposal in [24] where, however, the lowest rates are obtained by code extension. Opposite to puncturing, extending consists in adding parity check equations, when necessary. Though interesting in principle, the proposed procedure has a number of drawbacks, namely: 1) it precludes the lower triangular form for the parity matrix of the extended code; 2) it is obtained by adding regular square blocks, of weight 3, in the right bottom corner of the matrix; moreover, the blocks cannot have an arbitrary size: a minimum dimension exists in order to avoid the appearance, inside the block, of short cycles; 3) the added parity equations do not yield an almost uniform weight for the rows. Drawback 1) implies that Gaussian elimination is required, that can cause the loss of the sparse character for the matrix; drawback 2) limits the granularity of the achievable rates, that cannot assume any value [With pseudo-puncturing, instead (and, really, with classic puncturing as well) we can proceed by eliminating a single row and column (a single bit), so that there are much less theoretical limitations on the number of rates or the rate values we can generate.]; finally, drawback 3) is not a favourable choice to have maximum decoding efficiency.

In [25], the extending procedure has been improved in such a way as to provide a more uniform row weight distribution and a stronger dependency between the columns of the parity check matrix of the mother code and the newly added columns. Moreover, the modified extending is deterministic (which makes design and implementation modular and simple), and the PEG algorithm is applied to design the mother code and the extended one with matrixes having lower triangular form (with the advantage it implies, in regard to the encoding complexity, that grows linearly with the code size). In spite of these improvements, however, the procedure is exposed to criticism: granularity of the rate remains limited and, most of all, the PEG algorithm is applied separately to code portions, that does not ensure that the PEG advantages remain when these portions are inserted in the whole code. To the point that, because of the addition of two side-by-side identity matrices at the lower left part of the matrix, 4-length cycles may appear, with their dangerous impact on the code performance. On the contrary, this risk is absolutely absent in our pseudo-puncturing procedure.

Another possible assessment for pseudo-punctured codes would be to compare their performance with that of Tornado/LT/Raptor codes [26], [27], [28]. Such codes have been firstly proposed for channels with erasures, where the informa-

tion bits (or packets) are received as correct or discarded at all (when packets include at least one bit in error). A receiver can successfully decode the original source data once it receives a sufficient number of bits (packets); this is the reason why these codes are basically rate-less. However, erasure codes need the presence of other codes, working at a lower layer, able to act efficiently on the random errors induced by Gaussian noise and, in this other framework, the analysis of LDPC codes, seen as error correcting codes, must be conducted, in more conventional terms, on the AWGN channel, where performance depends on the spectral efficiency.

The applicability of Raptor Codes on the binary symmetric channel, and the AWGN channel, in particular, has been also studied in [29]. In practice, a Raptor Code over an AWGN channel is used in the following way. Given k source bits $x_1 \dots x_k$, these are first pre-coded into a codeword $(y_1 \dots y_n)$. A suitable probability distribution Ω on $\{1 \dots n\}$ is defined; the distribution Ω is sampled to obtain a weight w, and a further uniform sampling is made from the set of binary vectors of weight w to obtain a vector $(v_1 \dots v_n)$. The value of the output symbol is then obtained as $\sum_{i=1}^{n} v_i y_i$. The receiver collects output bits from the channel and records the reliability of each bit. This reliability translates into an amount of the information of the bit. The receiver collects bits until the sum of the informations of the individual bits is $k\,(1+\epsilon)$ where ϵ is an appropriate constant, called the reception overhead. Once reception is complete, the receiver applies belief-propagation decoding to recover the input bits.

Apparently, the concept of rate-less codes looks like the concept of rate-compatible codes and it is certainly true that, in principle, the performance of Raptor Codes could be compared with that of punctured codes. Raptor codes are designed in such a way as to have a reception overhead arbitrary close to zero, thus obtaining nearly capacity-achieving codes. In this sense, they should provide a formidable benchmark. It must be noted, however, that, contrary to the erasure channel, where "universal" Raptor Codes can be found, whose performance comes arbitrarily close to the capacity regardless of the erasure probability, in the AWGN channel the optimum degree distribution depends on the noise variance σ^2. This makes the project of these codes very difficult, and only preliminary examples of Raptor Codes tailored for the AWGN channel have been presented up to now. Actually, it has been verified that Raptor Codes designed for the binary erasure channel do not perform too badly on the AWGN channel [29] as well. This is an interesting starting point but the topic requires further investigation.

Even assuming that the design problem is solved, the management of these codes appears difficult, with a dynamical adjustment of the Tanner graph to comply the changeable rate. Though aware that the puncturing solution may be worse, and intrinsically unable to approach the capacity for variable rates, we feel that its implementation is much simpler, and therefore more realistic in the present scenario.

Finally, for the sake of completeness, we want to mention that in [30] the problem of obtaining multiple rate LDPC codes with a unique architecture has been solved by a different approach, that is maintaining constant the codeword length n. In practice, starting from a mother code with low rate, higher rate codes are obtained by reducing the number of rows in the parity check matrix. Reduction is realized by combining rows linearly, which is equivalent to replace a group of check nodes with a single check node that sums all the edges coming into each of the original check nodes. The mother matrix must be properly designed for this purpose. Apart from the observation that this approach requires optimization as well, and that rate granularity is partly limited, the idea of maintaining constant the codeword length can be inapplicable in many cases, where instead the constraint is on the information word length, due to the peculiarity of the application, that fixes the packet size or needs compatibility with existing standards. Based on these considerations, in our study we have assumed a constant k and a variable n, which is the choice done, for the same reasons, in all the papers reminded above.

VIII. Conclusion

When adopted in a system using high order QAM constellations to have large spectral efficiency, LDPC codes permit to obtain very good performance, comparable with and even better than that achieved by classic (and well-experienced) solutions, like TCM+RS systems. In applications permitting "ad hoc" designs, a significant additional coding gain can be achieved with low efforts. Often, however, there is the need to operate with variable rates and, in this case, it is desirable to have fast and easy reconfigurability, avoiding to switch among independent codes any time the rate changes.

The pseudo-puncturing strategy seems very promising for designing sets of codes with variable rates, based on a single (or very few, depending on the extent of the range of rates of interest) mother code(s). No substantial hardware or software changes are required when passing from one rate to another, except for an easy puncturing action at the transmitter and, in the sense explained, a "switch-off" of some routing paths at the receiver. This way, the pseudo-puncturing strategy is only slightly more complex than classic puncturing but, as our simulations have shown, ensures better performance.

The fundamental problem of designing the parity check matrix of the mother code, by selecting its own number of edges and symbols '1' density, has been faced by using some heuristic guidelines, whose effectiveness has been proved in a number of cases. We recognize this requires some skillfulness for a rapid convergence to an acceptable result. A possible development of the research could consist in searching for analytical approaches to explain the rationale of the method, thus translating it into general rules. Unfortunately, however, this task, together with other issues still open in the analysis of LDPC codes (like explicit formulas for the estimation of their error rate performance, the minimum distance evaluation, the weight distribution estimate, and others) is very difficult to solve.

For a given number of edges and symbols '1' density, the problem of designing a good code remains. For this purpose, we have proposed a new algorithm, called Local Cycles Optimization, based on the progressive construction of the Tanner graph of the code on an edge-by-edge basis.

This method has revealed particularly useful in applications showing stringent requirements on the complexity.

ACKNOWLEDGMENT

Part of this work has been sponsored by Siemens Mobile Communications S.p.A. The authors wish to thank the whole Siemens staff for helpful technical discussion and, in particular, Dr. Sergio Bianchi for having provided some results on TCM+RS systems.

Prof. Giovanni Cancellieri and Dr. Andrea Carassai are also acknowledged.

Finally, we are grateful to the anonymous reviewers that, with their comments and suggestions, have contributed to improve the quality of the paper.

REFERENCES

[1] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 21–28, Jan. 1962.

[2] D. J. C. MacKay and R. M. Neal, "Good codes based on very sparse matrices," in *Cryptography and Coding. 5th IMA Conference*, ser. Lecture Notes in Computer Science, C. Boyd, Ed. Berlin: Springer, 1995, no. 1025, pp. 100–111.

[3] ETSI EN 302 307 V1.1.1, "Digital video broadcasting (DVB); second generation framing structure, channel coding and modulation systems for broadcasting, interactive services, news gathering and other broadband satellite applications," June 2004.

[4] JPL, "LDPC code selection for CCSDS," CCSDS Meeting, Noordwijk, Tech. Rep., Apr. 2003.

[5] S. Lin, "Structured low-density parity-check codes: Algebraic constructions," IEEE P802.3an Task Force Meeting, Portland, OR, Tech. Rep., July 2004.

[6] X.-Y. Hu and E. Eleftheriou, "Progressive edge-growth Tanner graphs," in *Proc. IEEE Global Telecommunications Conference (GLOBECOM'01)*, San Antonio, Texas, Nov. 2001, pp. 995–1001.

[7] T. Richardson and R. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 638–656, Feb. 2001.

[8] Y. Kaji, M. P. Fossorier, and S. Lin, "Encoding LDPC codes using the triangular factorization," in *Proc. International Symposium on Information Theory and Its Applications (ISITA2004)*, Parma, Italy, Oct. 2004, pp. 37–42.

[9] L. Ping, W. Leung, and N. Phamdo, "Low density parity check codes with semi-random parity check matrix," *Electronics Letters*, vol. 35, pp. 38–39, Jan. 1999.

[10] X.-Y. Hu, E. Eleftheriou, and D.-M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Trans. Inform. Theory*, vol. 51, pp. 386–398, Jan. 2005.

[11] D. J. C. MacKay. (2004) Encyclopedia of sparse graph codes. [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html

[12] J. Hou, P. Siegel, L. Milstein, and H. Pfister, "Design of low-density parity-check codes for bandwidth efficient modulation," in *Proc. Information Theory Workshop*, Cairns, Qld, Sept. 2001, pp. 24–26.

[13] G. Bosco, R. Garello, and F. Mininni, "On low density parity check codes over Z_8," in *Proc. Conference on Software, Telecommunications and Computer Networks (SoftCOM2005)*, Marina Frapa, Croatia, Sept. 2005, SS1-5040-1509.

[14] Y. Li and W. E. Ryan, "Design of LDPC-coded modulation schemes," in *Proc. 3rd International Symposium on Turbo Codes*, Brest, France, Sept. 2003, pp. 551–554.

[15] G. Caire, G. Taricco, and E. Biglieri, "Bit-interleaved coded modulation," *IEEE Trans. Inform. Theory*, vol. 44, pp. 927–946, May 1998.

[16] R. Wesel, X. Liu, J. Cioffi, and C. Komninakis, "Constellation labeling for linear encoders," *IEEE Trans. Inform. Theory*, vol. 47, pp. 2417–2431, Sept. 2001.

[17] F. Chiaraluce and R. Garello, "Extended Hamming product codes analytical performance evaluation for low error rate applications," *IEEE Trans. Wireless Commun.*, vol. 3, pp. 2353–2361, Nov. 2004.

[18] J. Ha, J. Kim, and S. McLaughlin, "Rate-compatible puncturing of low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 50, pp. 2824–2836, Nov. 2004.

[19] T. Tian, C. Jones, and J. Villasenor, "Rate-compatible low-density parity-check codes," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, June 2004, p. 152.

[20] T. Flo, P. Orten, and B. Risløv, "Evaluation of coding schemes for spectrally efficient low-delay radio systems," in *Proc. Third International Symposium on Turbo Codes*, Brest, France, Sept. 2003, pp. 551–554.

[21] J. Ha, J. Kim, and S. McLaughlin, "Puncturing for finite length low-density parity-check codes," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, June 2004, p. 152.

[22] H. Pishro-Nik and F. Fekri, "Results on punctured LDPC codes," in *Proc. IEEE Information Theory Workshop (ITW)*, San Antonio, Texas, Oct. 2004, pp. 215–219.

[23] H. Pishro-Nik, N. Rahnavard, and F. Fekri, "Nonuniform error correction using low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 51, no. 7, pp. 2702–2714, July 2005.

[24] J. Li and K. R. Narayanan, "Rate-compatible low density parity check codes for capacity-approaching ARQ schemes in packet data communications," in *Proc. International Conference on Communications, Internet and Information Technology (CIIT)*, US Virgin Islands, Nov. 2002, pp. 201–206.

[25] M. Yazdani and A. Banihashemi, "On construction of rate-compatible low-density parity-check codes," *IEEE Commun. Lett.*, vol. 8, no. 3, pp. 159–161, Mar. 2004.

[26] M. Luby, M. Mitzenmacher, M. Shokrollahi, and D. Spielman, "Efficient erasure correcting codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 569–584, Feb. 2001.

[27] J. W. Byers, M. Luby, and M. Mitzenmacher, "A digital fountain approach to asynchronous reliable multicast," *IEEE J. Select. Areas Commun.*, vol. 20, no. 8, pp. 1528–1540, Oct. 2002.

[28] A. Shokrollahi. (2003, June) Raptor codes. [Online]. Available: http://www.digitalfountain.com

[29] O. Etesami, M. Molkaraie, and A. Shokrollahi. (2004) Raptor codes on symmetric channels. [Online]. Available: http://www.cs.berkeley.edu/∼etesami/

[30] A. I. Vila Casado, W.-Y. Weng, and R. D. Wesel, "Multiple rate low-density parity-check codes with constant blocklength," in *Proc. Thirty-Eighth Asilomar Conference on Signals, Systems and Computers*, vol. 2, Nov. 2004, pp. 2010–2014.

Marco Baldi was born in Macerata, Italy, in 1979. He received the 'Laurea' degree in Electronic Engineering (summa cum laude) from the Università Politecnica delle Marche in 2003. At present, he is a Ph.D. student at the Università Politecnica delle Marche, Department of Electronics, Artificial Intelligence and Telecommunications. His research activity mainly regards channel coding, with particular interest on iteratively decoded codes. He is co-author of several scientific papers on various topics.

Franco Chiaraluce was born in Ancona, Italy, in 1960. He received the Laurea in Ingegneria Elettronica (summa cum laude) from the Università di Ancona in 1985. Since 1987 he joined the Dipartimento di Elettronica ed Automatica of the same university. At present, he is an Associate Professor at the Università Politecnica delle Marche. His main research interests involve various aspects of communication systems theory and design, with special emphasis on coding, cryptography and multiple access techniques. He is co-author of more than 150 scientific papers and two books. He is member of IEEE and IEICE.