Iterative Soft-Decision Decoding of Binary Cyclic Codes

Marco Baldi, Giovanni Cancellieri, and Franco Chiaraluce

Original scientific paper

Abstract: **Binary cyclic codes achieve good error correction performance and allow the implementation of very simple encoder and decoder circuits. Among them, BCH codes represent a very important class of *t*-error correcting codes, with known structural properties and error correction capability. Decoding of binary cyclic codes is often accomplished through hard-decision decoders, although it is recognized that soft-decision decoding algorithms can produce significant coding gain with respect to hard-decision techniques. Several approaches have been proposed to implement iterative soft-decision decoding of binary cyclic codes. We study the technique based on "extended parity-check matrices", and show that such method is not suitable for high rates or long codes. We propose a new approach, based on "reduced parity-check matrices" and "spread parity-check matrices", that can achieve better correction performance in many practical cases, without increasing the complexity.**

Index terms**: error correction, binary cyclic codes, BCH codes, iterative soft decoding, belief propagation.**

I. INTRODUCTION

Binary cyclic codes represent a class of very important linear block codes, often included in telecommunication standards and applications. Their encoding and decoding can be accomplished through very simple circuits based on linear feedback shift registers (LFSR), that implement multiplication and division operations over the polynomial ring $GF_2[x] \bmod (x^n + 1)$, where n is the code length.

However, classic decoding techniques are applied onto a binary-output channel, thus realizing hard-decision decoders and allowing the correction of up to $\lfloor (d-1)/2 \rfloor$ errors, where d is the code minimum distance and $\lfloor x \rfloor$ the greatest integer smaller than x. On the contrary, the use of channel measurements in soft-decision decoders can significantly improve the error correction capability, thus approaching the theoretical limit of correcting $d - 1$ errors [1]. An important role in the current scenario of soft-decision decoders is played by low-density parity-check (LDPC) codes decoders, based on belief propagation (BP) algorithms [2]. They can approach the performance of the maximum likelihood (ML) decoder, while maintaining low decoding complexity.

Manuscript received February, 2008 and revised June, 2008. This paper was presented in part at the International Conference on Software, Telecommunications and Computer Networks (SoftCOM) 2007.

Authors are with Università Politecnica delle Marche, Ancona, Italy (e-mail: {m.baldi, g.cancellieri, f.chiaraluce}@univpm.it)

In order to achieve good performance, BP decoding needs a parity-check matrix with the following characteristics: i) sparsity, ii) absence of short cycles in the associated Tanner graph and iii) optimized (regular or irregular) row and column weight distributions. Such properties are rarely ensured by parity check-matrices of binary cyclic codes. For example, it can be shown that ($n = 2^m - 1$, k, d)-BCH codes, where k is the number of information bits, with rate $R \geq 1/2$ and $3 \leq m \leq 8$, cannot have 4-cycle-free Tanner graphs [3].

For these reasons, many alternative solutions have been proposed in the literature for effectively applying BP decoders to generic linear block codes, binary cyclic codes, or specific classes of cyclic codes [4]-[10]. All these techniques aim at finding, through different approaches, a graph representation for the code that is well-suited for BP decoding.

In [4], the author proposes to use the so called extended parity-check matrix (EPCM) in order to obtain a regular Tanner graph associated with the code. In [5] and [6], instead, the generalized parity-check matrix (GPCM) is adopted to reduce the number of short cycles. Such approach has been further investigated in [7], where an algorithm is presented that achieves a 4-cycle-free representation. All techniques based on GPCMs, however, require the introduction of auxiliary bits for which there is no evidence from the channel and this fact may cause performance degradation. In [8], it is demonstrated that Vardy's technique can be used to find sparse parity-check matrices for Reed-Solomon codes. Such technique, however, applies only for special cases, and considers binary images of RS codes, that are not necessarily cyclic. Clever techniques for applying belief propagation decoding to RS and more general codes have been proposed in [9] and [10]. The rationale of these methods lies in adapting the parity-check matrix at each iteration, according to the bit reliabilities, such that the unreliable bits correspond to a sparse submatrix, suitable for the BP algorithm. Such approach is able to provide good error correction performance, but graph modifications at each iteration could represent a problem in practical implementations.

In this paper, we start from the approach based on EPCMs, that ensures low decoding complexity and straightforward implementation. However, we show that such technique is convenient only for short length and low rate codes. We instead propose a new approach that is able to overcome such limitation, and to produce a significant improvement in the decoder performance.

We start reducing the density of the EPCM through linear operations (i.e., combination of rows), thus obtaining a

"reduced" parity-check matrix (RPCM) that is more suitable for BP decoding. Then, the RPCM is used as input for a "spreading" algorithm, that produces a "spread" parity-check matrix (SPCM). This way, the BP decoding algorithm works on a larger matrix that corresponds to a more favorable Tanner graph.

The paper is organized as follows: Section II is devoted to the parity-check matrix of a binary cyclic code and its modifications. Section III describes the standard decoding algorithm and its alternative version that works on the spread code. In Section IV the new technique is assessed through numerical simulations and, finally, Section V concludes the paper.

II. Parity-Check Matrices for Binary Cyclic Codes

We consider the class of binary cyclic codes $C(n, k)$, with length n, dimension k and redundancy $r = n - k$. Each codeword $\mathbf{c}$ of a binary cyclic code can be associated to a polynomial $c(x)$ over $GF_2[x] \bmod (x^n + 1)$. All the shifted versions of $c(x)$, $x^i c(x)$, are valid codewords as well, due to the cyclic property of the code. Within the set of code polynomials in C there is a unique monic polynomial $g(x)$, with minimal degree $r < n$, called the generator polynomial of C. Every codeword polynomial $c(x) \in C$ can be expressed uniquely as $c(x) = m(x)g(x) \bmod (x^n + 1)$, where $m(x) \in GF_2[x]$ is a polynomial of degree $< k$. The generator polynomial $g(x)$ of C is a factor of $(x^n + 1)$, and there exists a parity polynomial with degree k, $h(x)$, such that $g(x)h(x) = x^n + 1$. Moreover, since $g(x)$ divides $c(x)$, it is:

$$c(x)h(x) \equiv 0 \bmod \left(x^n + 1\right), \quad \forall c(x) \in C . \tag{1}$$

A. Standard Parity-Check Matrix

The standard form of the parity-check matrix (PCM) of a binary cyclic code is as follows [11]:

$$\mathbf{H} = \begin{bmatrix} h_k & \cdots & h_1 & h_0 & 0 & \cdots & 0 \\ 0 & h_k & \cdots & h_1 & h_0 & 0 & \cdots \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & \cdots & 0 & h_k & \cdots & h_1 & h_0 \end{bmatrix} \tag{2}$$

where h_i, $i = 0 \ldots k$, are the binary coefficients of $h(x)$.

The form (2) of the parity-check matrix is not suitable for BP decoding: it contains a high number of length-4 cycles and it has irregular and non-optimized column weights.

B. Extended Parity-Check Matrix

The parity-check matrix in the form (2) is a (non singular) submatrix of the EPCM of a cyclic code, that has the following form [4]:

$$\mathbf{H}^E = \begin{bmatrix} h_k & \cdots & h_1 & h_0 & 0 & \cdots & 0 \\ 0 & h_k & \cdots & h_1 & h_0 & 0 & \cdots \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & \cdots & 0 & h_k & \cdots & h_1 & h_0 \\ h_0 & 0 & \cdots & 0 & h_k & \cdots & h_1 \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ \cdots & h_1 & h_0 & 0 & \cdots & 0 & h_k \end{bmatrix} \tag{3}$$

$\mathbf{H}^E$ is a binary circulant matrix, in which each row is obtained through a cyclic shift of the previous row. Therefore, $\mathbf{H}^E$ can be represented through a set, $B^E \subset \mathbb{Z}_n$, containing the positions of the 1 symbols in its first row.

The form (3) of the parity-check matrix corresponds to a regular Tanner graph, free of low-weight nodes; therefore, at least in principle, it is more suitable for BP decoding. This is the underlying idea of the work [4]. However, the form (3) of the parity-check matrix contains a number of short cycles higher than that in matrix (2). When the number of non-null coefficients of $h(x)$ increases (as, for example, when long or high rate codes are considered), $\mathbf{H}^E$ has an extremely high number of short cycles, that deteriorate performance.

C. Reduced Parity-Check Matrix

A first improvement in the performance of the BP decoder can be obtained when a sparser representation for the code parity-check matrix is found.

For this purpose, we propose a very simple iterative algorithm that, by means of linear combinations between couples of rows, aims at deriving, from the EPCM, a "reduced parity-check matrix" (RPCM), $\mathbf{H}^R$, whose density is lower than that of $\mathbf{H}^E$.

The algorithm relies on the fact that, for a circulant matrix, the number of overlapping 1's between its first row and each other row can be easily calculated in terms of the periodic autocorrelation of the first row.

As an example, Fig. 1 shows the periodic autocorrelation of the first row of $\mathbf{H}^E$ (denoted as $\mathbf{h}_1$ in the following) for the BCH(127, 71) code.

We observe that, for a null shift, the periodic autocorrelation takes the value 48, that coincides with the Hamming weight of $\mathbf{h}_1$, denoted as w_1 in the following. We also notice that, for a shift value equal to 4, the periodic autocorrelation assumes its maximum value (except for the value with a null shift), that is equal to 32. It follows that, by summing the fifth row of $\mathbf{H}^E$ to its first row, we obtain a new vector, $\mathbf{h}_2$, with Hamming weight $w_2 = 2(48 - 32) = 32$.

The new vector $\mathbf{h}_2$ provides a valid parity-check equation for the original code, since it is obtained as a linear combination of parity-check vectors. Due to the cyclic nature of the code, any cyclically shifted version of $\mathbf{h}_2$ is a parity-check vector as well. Therefore, $\mathbf{h}_2$ can be used to obtain a new parity-check matrix in circulant form, with reduced density with respect to $\mathbf{H}^E$.

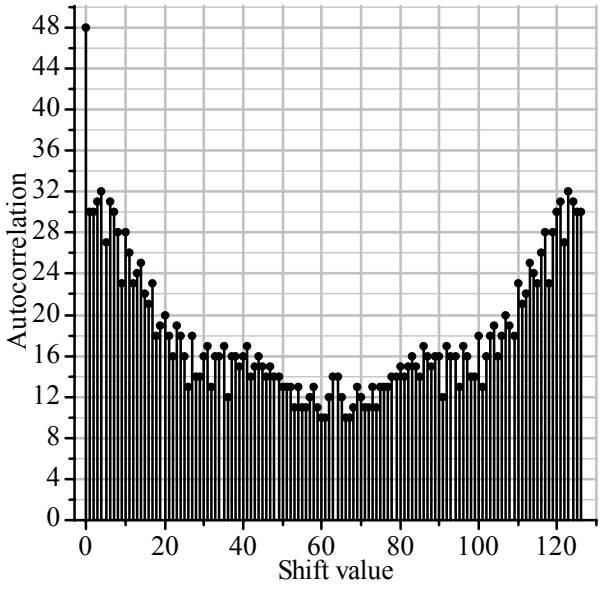

Fig. 1. Periodic autocorrelation of the first row of $\mathbf{H}^E$ for the BCH(127, 71) code

In general, given the vector $\mathbf{h}_i$, it is possible to reduce its density through this procedure if its periodic autocorrelation has a maximum value (out of the null shift) greater than half of its Hamming weight, $w_i/2$. So, we can apply an iterative density reduction algorithm as follows:

1. Set $i = 1$; initialize $\mathbf{h}_1$ as the first row of $\mathbf{H}^E$ and w_1 as its Hamming weight.
2. Calculate the periodic autocorrelation of $\mathbf{h}_i$ and its maximum value a, corresponding to the shift value $v \geq 1$. If $a > w_i/2$, go to step 3, otherwise stop and output $\mathbf{h}_i$.
3. Calculate $\mathbf{h}_{i+1} = \mathbf{h}_i + \mathbf{h}_i^v$ (where $\mathbf{h}_i^v$ represents the cyclically shifted version of $\mathbf{h}_i$ by v positions), and its Hamming weight $w_{i+1} = 2(w_i - a)$. Increment i and go back to step 2.

When the algorithm stops, it gives as output a binary vector $\mathbf{h}_i$ with density less than or equal to that of $\mathbf{h}_1$. $\mathbf{h}_i$ is then used to obtain the reduced parity-check matrix, in the form of a circulant matrix having $\mathbf{h}_i$ as its first row.

We say that the algorithm is successful when the RPCM has a reduced density with respect to the EPCM, that is, the algorithm executes step 3 at least once. This does not occur for BCH codes with small error-correction capability, as, for example, the BCH (31, 16), (63, 45) and (63, 57) codes, the first two being able to correct $t = 3$ errors, and the third one (that is a Hamming code) being able to correct $t = 1$ error. On the contrary, for the BCH(63, 39) code, that is able to correct $t = 4$ errors, it is possible to obtain an RPCM with half the density of the corresponding EPCM. For the BCH(127, 71) code, instead, the algorithm stops after one iteration, thus obtaining a moderate density reduction (from $w_1 = 48$ to $w_2 = 32$), but this is enough to produce a considerable performance improvement, as we will show in Section IV.

D. Spread Parity-Check Matrix

The target of having a parity-check matrix more suitable for BP decoding can be achieved when the decoder works on a "spread" code, simply built by repeating s times each codeword of the original code. In order to derive a valid parity-check matrix for the spread code, we identify a set of s binary circulant matrices, $\mathbf{H}_i^S$, $i = 1 \ldots s$, that sum into $\mathbf{H}^R$ (we remind that $\mathbf{H}^R$ may be either coincident with $\mathbf{H}^E$ or a sparser version of it). In formula:

$$\mathbf{H}^R = \sum_{i=1}^{s} \mathbf{H}_i^S \,. \tag{4}$$

If $\mathbf{c}$ is an n-bit codeword of the original code, it must be:

$$\mathbf{H}^R \mathbf{c}^T = \left(\sum_{i=1}^{s} \mathbf{H}_i^S \right) \mathbf{c}^T = \mathbf{0} \tag{5}$$

where superscript ^{T} denotes vector transposition, and $\mathbf{0}$ represents the $n \times 1$ null vector. Let us consider the following "spread" parity-check matrix:

$$\mathbf{H}^S = \left[\mathbf{H}_1^S \,|\, \mathbf{H}_2^S \,|\, \ldots \,|\, \mathbf{H}_s^S \right] \tag{6}$$

and the following "spread" codeword, obtained by repeating s times the generic codeword $\mathbf{c}$:

$$\mathbf{c}^S = \left[\mathbf{c} \,|\, \mathbf{c} \,|\, \ldots \,|\, \mathbf{c} \right] . \tag{7}$$

It follows from their definitions that:

$$\begin{aligned} \mathbf{H}^S \left(\mathbf{c}^S \right)^T &= \left[\mathbf{H}_1^S \,|\, \mathbf{H}_2^S \,|\, \ldots \,|\, \mathbf{H}_s^S \right] \left[\mathbf{c} \,|\, \mathbf{c} \,|\, \ldots \,|\, \mathbf{c} \right]^T = \\ &= \left[\mathbf{H}_1^S \mathbf{c}^T + \mathbf{H}_2^S \mathbf{c}^T + \ldots + \mathbf{H}_s^S \mathbf{c}^T \right] = \\ &= \mathbf{H}^R \mathbf{c}^T = \mathbf{0} \,. \end{aligned} \tag{8}$$

Therefore, $\mathbf{H}^S$ is a valid parity-check matrix for the spread code, and it will be used by the modified decoding algorithm to work on a more efficient graph. It is important to notice that the original code and its transmission rate are not altered: the spread code is used only inside the decoder, with the aim of decoding better the original code.

There exist several techniques and algorithms to design parity-check matrices in the form of a row of circulants free of length-4 cycles [12]. In the present case, however, we have a number of constraints on the elements of $\mathbf{H}^S$ deriving from the structure of $\mathbf{H}^R$. If we represent $\mathbf{H}^S$ through the multiset $B^S = \{B_1^S, B_2^S, \ldots, B_s^S\}$, that contains the positions of the 1 symbols in the first row of each block $\mathbf{H}_i^S$, we must solve the problem of optimally mapping B^R into B^S.

For this purpose, we adopt a heuristic approach, based on exhaustive enumerations or random searches, to partition the elements of B^R into the blocks of B^S, minimizing the number of length-4 cycles in $\mathbf{H}^S$.

Another relevant aspect is the choice of the spreading factor s, that is strictly related to the column weight of $\mathbf{H}^S$. Our numerical simulations have shown that, for the considered cases, the choice of a column weight close to 7 for $\mathbf{H}^S$ usually

yields the best performance; therefore, s has been fixed accordingly.

III. The Decoding Algorithm

We consider the sum-product algorithm with log-likelihood ratios (LLR-SPA) [13], that is very common for decoding LDPC codes. This algorithm works on Tanner graphs, that are bipartite graphs with variable and check nodes corresponding to code and control bits, respectively. An edge connecting the variable node v_i with the check node z_j exists if and only if the parity-check matrix associated with the Tanner graph has a 1 at position (j, i).

Decoding is based on the exchange of messages between variable and check nodes: information on the reliability of the i-th received bit c_i is sent as a message from the variable node v_i to the check node z_j, $\Gamma_{i\to j}(c_i)$, then elaborated, and sent back as a message from the check node z_j to the variable node v_i, $\Lambda_{j\to i}(c_i)$.

The algorithm starts by initializing both sets of messages, that is, $\forall\, i, j$ for which an edge exists between nodes v_i and z_j, we set:

$$\begin{cases} \Gamma_{i\to j}(c_i) = L(c_i) = \ln\left[\dfrac{P(c_i=0\mid y_i)}{P(c_i=1\mid y_i)}\right], & i=1\ldots n \\ \Lambda_{j\to i}(c_i) = 0 \end{cases} \tag{9}$$

where $L(c_i)$ is the initial reliability value based on the channel measurement information, and $P(c_i = x \mid y_i)$, $x \in \{0, 1\}$, is the probability that the codeword bit c_i at position i is equal to x, given a received signal y_i at the channel output.

After initialization, the LLR-SPA algorithm starts iterating. At each iteration, messages sent from the check nodes to the variable nodes are calculated by means of the following formula:

$$\Lambda_{j\to i}(c_i) = 2\tanh^{-1}\left\{\prod_{l\in A(j)\setminus i}\tanh\left[\frac{1}{2}\Gamma_{l\to j}(c_l)\right]\right\} \tag{10}$$

where $A(j)\backslash i$ represents the set of variable nodes connected to the check node z_j, with the exclusion of node v_i.

Messages sent from the variable nodes to the check nodes are then calculated as follows:

$$\Gamma_{i\to j}(c_i) = L(c_i) + \sum_{l\in B(i)\setminus j}\Lambda_{l\to i}(c_i) \tag{11}$$

where $B(i)\backslash j$ represents the set of check nodes connected to the variable node v_i, with the exclusion of node c_j. Moreover, the following quantity is evaluated:

$$\Gamma_i(c_i) = L(c_i) + \sum_{l\in B(i)}\Lambda_{l\to i}(c_i) \tag{12}$$

that is used to obtain an estimate ($\hat{\mathbf{c}}$) of the received codeword ($\mathbf{c}$) as follows:

$$\hat{c}_i = \begin{cases} 0 & \text{if} \quad \Gamma_i(c_i) \ge 0 \\ 1 & \text{if} \quad \Gamma_i(c_i) < 0. \end{cases} \tag{13}$$

The estimated codeword $\hat{\mathbf{c}}$ is then multiplied by the parity-check matrix associated with the Tanner graph. If the parity-check is successful, the decoding process stops and gives the estimated codeword as its result. Otherwise, the algorithm reiterates, using updated messages. In this case, a further verification is made on the number of decoding iterations: when a maximum number of iterations is reached, the decoder stops the estimation efforts and outputs the estimated codeword as its result. In this case, however, decoding is unsuccessful and the error detected.

A. Adaptation to the spread code

In order to take advantage of spread parity-check matrices, we adopt a modified version of the standard BP decoding algorithm.

The demodulator and demapper block produces, for each received bit, an initial reliability value, expressed as the log-likelihood ratio of *a priori* probabilities, $L(c_i)$, that is used to initialize the decoding algorithm [see Eq. (9)]. Once having obtained $L(c_i)$ for any i, the vector containing the $L(c_i)$ values is repeated s times to form the new vector of $L(c_i^S)$ values, valid for the spread code. This is used to initialize the LLR-SPA algorithm that works on the spread parity-check matrix, which then starts iterating and, at each iteration, produces updated versions of the extrinsic $[\Gamma_{i\to j}(c_i^S)]$ and *a posteriori* $[\Gamma_i(c_i^S)]$ messages. While the former are used as input for the subsequent iteration (if needed), the latter represent the decoder output, and serve to obtain an estimated codeword that is subject to the parity-check test. In addition, this version of the algorithm produces a posteriori messages also for the original codeword, as follows:

$$\Gamma_i(c_i) = \sum_{l=0}^{s-1}\Gamma_{i+l\cdot n}\left(c_{i+l\cdot n}^S\right), \qquad i=1\ldots n. \tag{14}$$

Two estimated codewords, $\hat{\mathbf{c}}^S$ and $\hat{\mathbf{c}}$, are derived on the basis of the sign of $\Gamma_i(c_i^S)$ and $\Gamma_i(c_i)$, respectively, and their corresponding parity-check tests are executed (based on $\mathbf{H}^S$ and $\mathbf{H}^R$, respectively). If both tests are successful, the decoder stops iterating and outputs $\hat{\mathbf{c}}$ as the estimated codeword; otherwise, decoding continues until a maximum number of iterations is reached. This double parity-check test permits to reduce significantly the number of undetected errors (decoder failures), as we have verified through numerical simulations.

IV. Numerical Simulations

In order to assess the benefits of the proposed approach, we have simulated transmission over the additive white Gaussian noise channel in conjunction with binary phase shift keying modulation, for different codes. We have considered several binary BCH codes with different length and rate. In this section, we report the results of numerical simulations carried out on BCH codes with (n, k) = (31, 16), (63, 45), (63, 57) and (127, 71).

For the first 3 codes, the density reduction algorithm is not successful, so we apply the spreading technique directly to the

extended parity-check matrix (that coincides with the reduced parity-check matrix).

TABLE I
POLYNOMIALS AND PARITY-CHECK MATRICES FOR THE (31, 16), (63, 45) AND (63, 57) BCH CODES

BCH code	$g(x)$	$h(x)$	B^S	# 4-cycles in PCM	# 4-cycles in EPCM	# 4-cycles in SPCM
(31, 16)	1, 5, 6, 7, 8, 9, 11, 13, 14, 15, 16	1, 5, 6, 7, 8, 13, 16, 17	[5 7 13 16 17] [1 6 8]	172	558	31
(63, 45)	1, 2, 3, 4, 10, 12, 13, 16, 17, 18, 19	1, 2, 5, 6, 9, 15, 16, 19, 22, 23, 24, 25, 26, 29, 30, 32, 35, 37, 39, 40, 41, 42, 45, 46	[5 15 23 26 40 42 46] [1 2 16 25 32 37 45] [7 20 23 30 31 36 40] [10 25 42]	7251	72954	2961
(63, 57)	1, 6, 7	1, 6, 7, 11, 13, 16, 17, 18, 19, 21, 25, 26, 27, 30, 33, 35, 36, 38, 39, 40, 42, 43, 46, 47, 49, 51, 53, 54, 55, 56, 57, 58	[25 33 38 47 54 57 58] [1 16 18 26 30 36 53] [6 7 19 27 40 49 56] [11 17 39 43 46 51] [13 21 35 42 55]	1800	234360	7749

Their characteristics are summarized in Table I, that reports, in particular, the indexes of the non null coefficients in $g(x)$ and $h(x)$. Enumeration is reversed, that is: 1 stays for $g_r = 1$ or $h_k = 1$ (that is always true), 2 for $g_{r-1} = 1$ or $h_{k-1} = 1$ and so on. Table I also reports the partition chosen to form the blocks of $\mathbf{H}^S$ (each set B_i^S is written in square brackets) and the number of length-4 cycles in each form of the parity-check matrix: standard, extended and spread.

We notice that the spread parity-check matrix has always a number of length-4 cycles greatly reduced with respect to the extended parity-check matrix and, in the first two cases, also lower than that of the classic parity-check matrix. Only the (63, 57)-BCH code, that is characterized by a very small r, has a parity-check matrix in the form (2) with the smallest number of length-4 cycles. Figs. 2-4 show the bit error rate (BER) and frame error rate (FER) curves, as a function of the signal-to-noise ratio E_b/N_0. These curves have been obtained, through numerical simulations of the considered codes, when decoding with the classic parity-check matrix (PCM), the extended parity-check matrix (EPCM) and the spread parity-check matrix (SPCM). The figures report also curves for the union bound (UB), that can be used as a reference for the error rate under ideal (maximum likelihood) decoding [14].

Fig. 2 shows that, for the (31, 16) BCH code, the EPCM technique achieves the best performance, with a loss of about 0.5 dB with respect to the union bound. The SPCM technique has almost the same performance as the classic PCM, although its curves seem to show a slightly more favorable slope. For the (63, 45) BCH code, instead, the EPCM and the SPCM techniques show the same BER performance, that is about 2 dB away from the union bound, as shown in Fig. 3. The PCM produces worse curves, with a further loss of about 0.5 dB. In addition, for the same BER, the EPCM technique is able to ensure better FER, even if the SPCM curves seem to show a more favorable slope also in this case.

The new SPCM-based technique becomes advantageous for longer codes and higher rates.

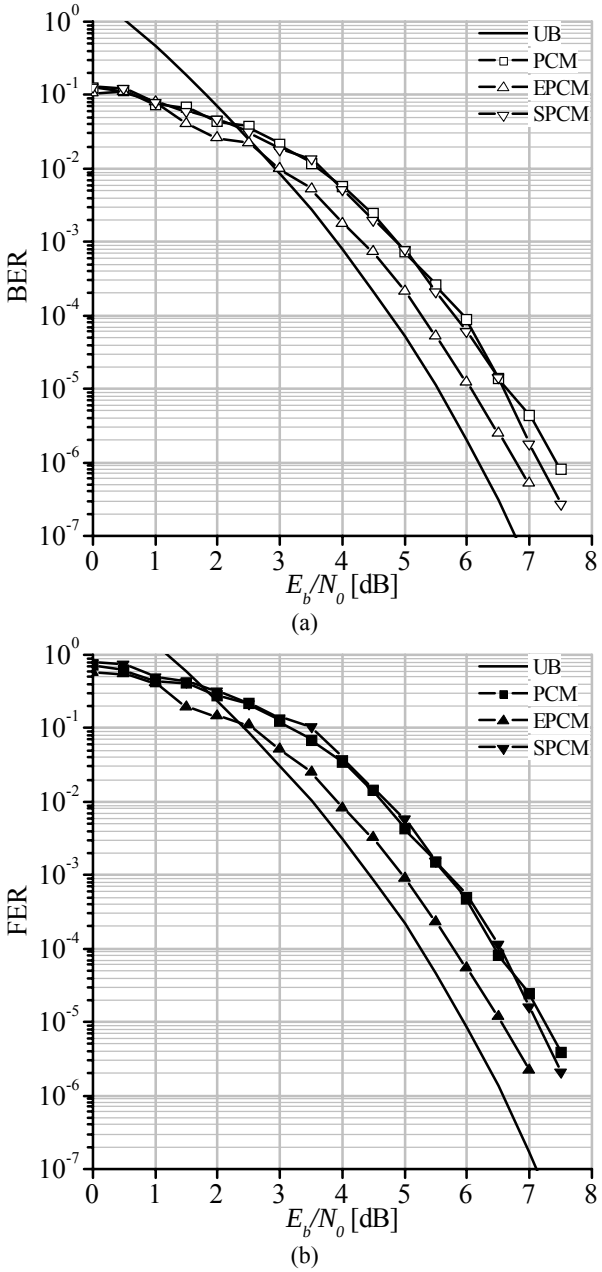

Fig. 2. Simulated BER (a) and FER (b) for the (31, 16) BCH code

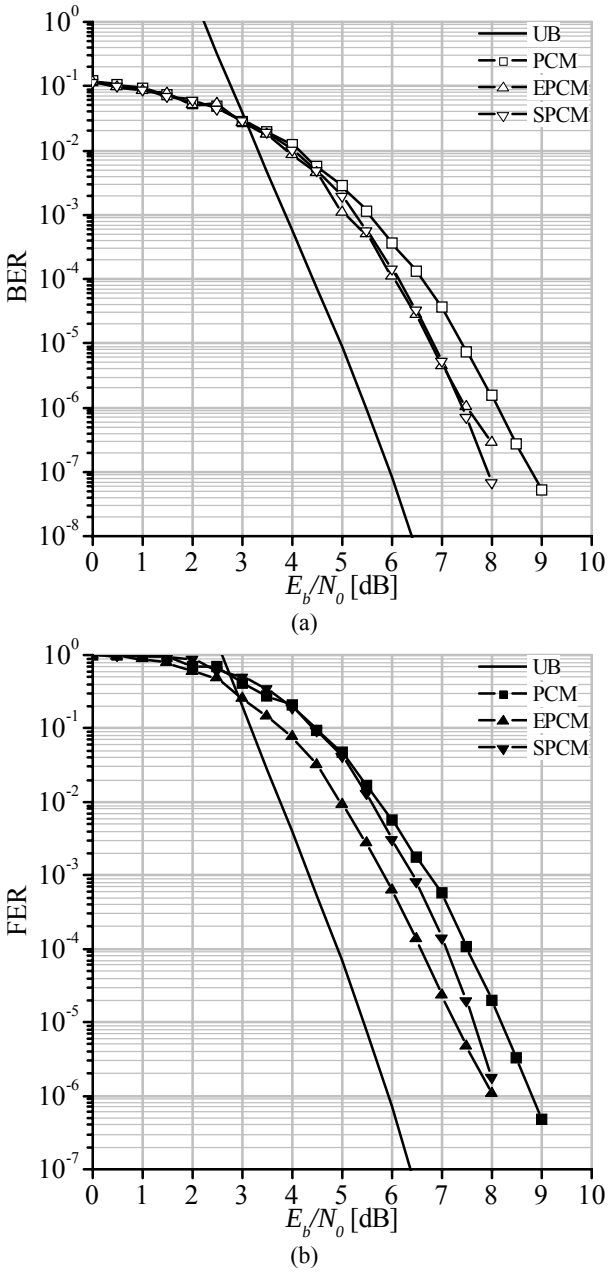

Fig. 3. Simulated BER (a) and FER (b) for the (63, 45) BCH code

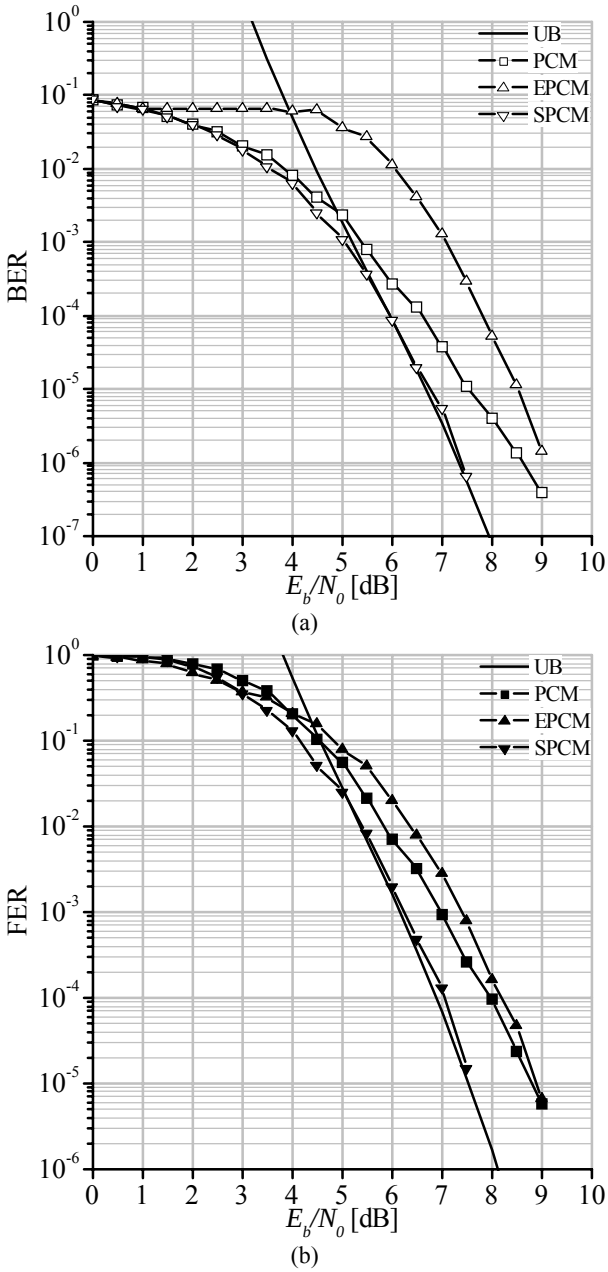

Fig. 4. Simulated BER (a) and FER (b) for the (63, 57) BCH code

Fig. 4 shows the simulated performance of the (63, 57)-BCH code: we notice that the new technique outperforms those based on the classic PCM and on the EPCM, with a gain of more than 1 dB over the PCM and more than 1.5 dB over the EPCM.

Furthermore, the curves obtained through the SPCM approach are almost overlaid to those of the union bound: this means that the SPCM decoder achieves the performance of the ideal decoder, at least in the explored region of E_b/N_0 values.

Finally, we have considered the (127, 71) BCH code. In this case, the density reduction algorithm is successful and, starting from $\mathbf{h}_1$ with Hamming weight 48, a vector $\mathbf{h}_2$ is obtained with Hamming weight 32, thus reducing by 1/3 the parity-check matrix density.

Hence, spreading has been applied to the RPCM. Table II reports the parameters of the code's characteristic matrices, and Fig. 5 its simulated performance.

Also in this case, the new technique outperforms the others, with about 2 dB of gain over the PCM-based algorithm and 3 dB over the EPCM approach. On the other hand, the SPCM-based algorithm, though working on a graph with a smaller number of length-4 cycles (see Table II), does not provide any improvement, at least for BER $\geq 10^{-6}$ and FER $\geq 10^{-5}$.

This seems to suggest that, when successful, the density reduction algorithm produces a representation of the code that does not need any further processing in order to achieve very good performance.

TABLE II
CHARACTERISTICS OF THE (127, 71) BCH CODE

1's positions in $\mathbf{h}_1$	1's positions in $\mathbf{h}_2$	B^S	# 4-cycles in PCM	# 4-cycles in EPCM	# 4-cycles in RPCM	# 4-cycles in SPCM
1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 17, 18, 20, 21, 23, 26, 27, 30, 33, 34, 37, 41, 43, 45, 47, 48, 49, 50, 51, 53, 55, 56, 57, 58, 60, 61, 64, 65, 66, 67, 68, 70, 71, 72	1, 2, 3, 4, 5, 9, 11, 15, 19, 22, 23, 24, 25, 26, 31, 33, 38, 43, 48, 50, 52, 54, 56, 58, 59, 62, 66, 67, 69, 74, 75, 76	[11 5 62 24 33 48 31] [3 67 56 15 9 75 59] [50 38 25 1 4 2 43] [76 22 66 19 26 54 52] [69 58 74 23]	378314	1356614	240284	4699

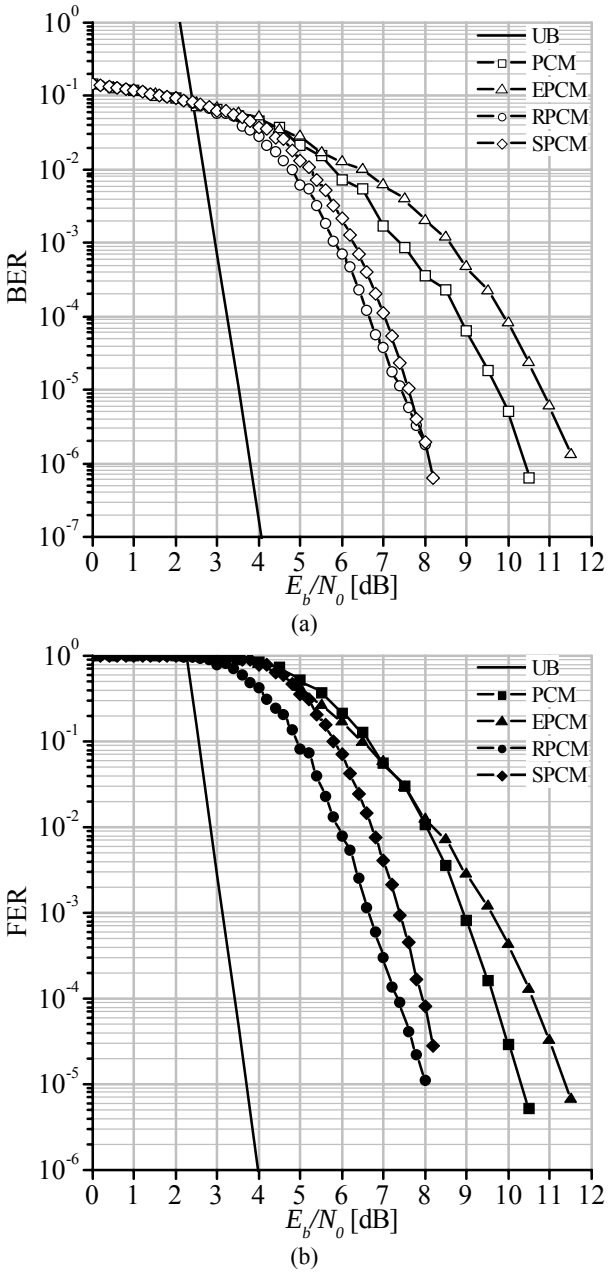

Fig. 5. Simulated BER (a) and FER (b) for the (127, 71) BCH code

Moreover, for the (127, 71) BCH code, the FER curve is quite distant from the union bound, showing that further coding gain could be achieved, in principle. Actually, techniques based on the adaptation of the parity-check matrix during decoding show better performance, for the same code parameters [10], though requiring higher complexity.

VI. CONCLUSIONS

We have described a new approach for iterative soft-decision decoding of binary cyclic codes, based on the concepts of reduced and spread parity-check matrices (RPCMs and SPCMs).

The essence of the method is in the possibility to overcome the drawbacks of the parity check matrix of these codes, namely the high density of 1 symbols and the presence of short cycles in the Tanner graph, that prevent effective application of the BP decoding algorithm.

The proposed technique can ensure performance comparable with, or better than, that of the classic approach and the extended parity-check matrix (EPCM) approach, without increasing complexity of the decoding stage.

REFERENCES

[1] D. Chase: *A Class of Algorithms for Decoding Block Codes with Channel Measurement Information*, IEEE Trans. Inf. Theory, Vol. 18, pp. 170–182, Jan. 1972.

[2] T. J. Richardson, R. L. Urbanke: *The Capacity of Low-Density Parity-Check Codes Under Message-Passing Decoding*, IEEE Trans. Inf. Theory, Vol. 47, pp. 599–618, Feb. 2001.

[3] T. R. Halford, A. J. Grant, K. M. Chugg: *Which Codes Have 4-Cycle-Free Tanner Graphs?*, IEEE Trans. Inf. Theory, Vol. 52, pp. 4219–4223, Sep. 2006.

[4] R. H. Morelos-Zaragoza: "Architectural Issues of Soft-Decision Iterative Decoders for Binary Cyclic Codes", Tech. Rep., Sony ATL, Aug. 2000.

[5] J. S. Yedidia, J. Chen, M. Fossorier: "Generating Code Representations Suitable for Belief Propagation Decoding," Tech. Rep. TR-2002-40, Mitsubishi Electric Research Laboratories, Sep. 2002.

[6] J. S. Yedidia, J. Chen, M. Fossorier: *Representing Codes for Belief Propagation Decoding*, in Proc. IEEE ISIT 2003, Yokohama, Japan, Jul. 2003, p. 176.

[7] S. Sankaranarayanan, B. Vasic: *Iterative Decoding of Linear Block Codes: A Parity-Check Orthogonalization Approach*, IEEE Trans. Inf. Theory, Vol. 51, pp. 3347–3353, Sep. 2005.

[8] B. Kamali, A. H. Aghvami: *Belief Propagation Decoding of Reed-Solomon Codes; a Bit-Level Soft Decision Decoding Algorithm*, IEEE Trans. Broadcasting, Vol. 51, pp. 106–113, Mar. 2005.

[9] J. Jiang, K. R. Narayanan: *Iterative Soft-Input Soft-Output Decoding of Reed–Solomon Codes by Adapting the Parity-Check Matrix*, IEEE Trans. Inf.. Theory, Vol. 52, pp. 3746–3756, Aug. 2006.

[10] A. Kothiyal, O. Y. Takeshita; *A Comparison of Adaptive Belief Propagation and the Best Graph Algorithm for the Decoding of Linear Block Codes*, in Proc. IEEE ISIT 2005, Adelaide, Australia, Sep. 2005, pp. 724–728.
[11] S. B. Wicker: "Error Control Systems for Digital Communication and Storage," Prentice-Hall, Jul. 1994.
[12] M. Baldi, F. Chiaraluce, R. Garello, F. Mininni: *Quasi-Cyclic Low-Density Parity-Check Codes in the McEliece Cryptosystem*, Proc. IEEE ICC 2007, Glasgow, Scotland, Jun. 2007, pp. 951-956.
[13] J. Hagenauer, E. Offer, L. Papke: *Iterative Decoding of Binary Block and Convolutional Codes*, IEEE Trans. Inf.. Theory, Vol. 42, pp. 429–445, Mar. 1996.
[14] R. H. Morelos-Zaragoza: "The Art of Error Correcting Coding," Wiley, 2002.

Marco Baldi was born in Macerata, Italy, in 1979. He received the "Laurea" degree (summa cum laude) in Electronics Engineering in 2003, and the Doctoral degree in Electronics, Informatics and Telecommunications Engineering in 2006 from the Polytechnic University of Marche, Ancona, Italy. At present, he is a post-doctoral researcher and contract Professor at the same university. His main research activity is in channel coding, with particular interest in linear block codes for symmetric and asymmetric channels, low-density parity-check (LDPC) codes and their application in cryptography.

Giovanni Cancellieri was born in Florence, Italy, in 1952. He received the degrees in Electronic Engineering and in Physics from the University of Bologna. Since 1986 he is Full Professor of Telecommunications at the Polytechnic University of Marche. His main research activities are focused on optical fibres, radio communications and wireless systems, with special emphasis on channel coding and modulation systems. He is co-author of about one hundred fifty papers, five books of scientific contents, and two international patents. Since 2003 he is president of CReSM (Centro Radioelettrico Sperimentale G. Marconi).

Franco Chiaraluce was born in Ancona, Italy, in 1960. He received the "Laurea in Ingegneria Elettronica" (summa cum laude) from the University of Ancona in 1985. Since 1987 he joined the Department of Electronics and Automatics of the same university. At present, he is an Associate Professor at the Polytechnic University of Marche. His main research interests involve various aspects of communication systems theory and design, with special emphasis on error correcting codes, sensor networks, cryptography and multiple access techniques. He is co-author of more than 180 papers and two books. He is member of IEEE and IEICE.