Packet-LDPC Codes for Tape Drives

Yang Han and William E. Ryan, *Senior Member, IEEE*

Electrical and Computer Engineering Department, University of Arizona, Tucson, AZ 85712 USA

In this paper, we introduce packet low-density parity-check (packet-LDPC) codes for high-density tape storage systems. We report on the performance of two error control code (ECC) architectures based on the packet-LDPC codes. The architectures are designed to be (approximately) compatible with the widely used ECMA-319 ECC standard based on two interleaved concatenated 8-bit Reed–Solomon (RS) codes. One architecture employs an inner RS code; the other employs an inner turbo product code with single parity-check constituent codes (TPC-SPC). Both employ a packet-LDPC code as the outer code. As for the ECMA-319 system, both schemes are required to correct noise bursts due to media defects and synchronization loss, as well as the loss of one of eight tracks (due to a head clog, for example). We show that the first packet-LDPC code architecture substantially outperforms the ECMA-319 scheme and is only a few tenths of a decibel inferior to a long, highly complex 12-bit RS scheme. The second architecture outperforms both the ECMA-319 and the long RS code scheme.

Index Terms—Error control code (ECC), low-density parity-check (LDPC), packet-LDPC, tape recording.

I. INTRODUCTION

BECAUSE of their superior performance capabilities on many channels, low-density parity-check (LDPC) codes have received quite a bit of attention in the magnetic disk drive industry in recent years [1]–[7]. LDPC codes are also considered for high-density magnetic tape recording channel in [8]. However, to our knowledge, no published work exists on the application of concatenated schemes involving LDPC codes to tape storage channels. In this paper, we report on our very promising results in this regard. In particular, by selecting the ECMA-319 Reed–Solomon (RS) code-based standard as a baseline in a case study, we show that the tape storage channel presents a unique challenge to the code system designer. Specifically, while tape drives permit longer codes over larger frames (sectors) than is possible for disk drives, the code must be capable of recovering one entire track out of eight (for ECMA-319), lost as a result of a head clog, for example. Such a requirement has led us to consider the application of LDPC codes in a nonstandard way, resulting in what we call packet-LDPC codes. The idea for packet-LDPC codes originated in the work of Luby *et al.* [9], who applied them to file multicast over the Internet.

In this paper, we show that two error control code (ECC) architectures based on packet-LDPC codes are very promising for tape storage devices. The first architecture employs an inner RS code and the second employs an inner turbo product code with single parity-check constituent codes (TPC-SPC). Both employ a packet-LDPC code as the outer code. Substantial gains relative to the state-of-the-art RS-based system are shown to be possible, even in the event of a track loss. Further, the first architecture has complexity comparable to that of ECMA-319.

This paper is organized as follows. Section II presents the ECMA-319 coding standard. Section III discusses LDPC codes in general and packet-LDPC and TPC-SPC codes in particular. Section IV presents the performance of an ECMA-319-compatible coding architecture with an inner RS code and an outer packet-LDPC code. The performance of this scheme is compared with the 8-bit RS code ECMA-319 scheme as well as an

Fig. 1. ECMA-319 standard data format.

ECMA-319-compatible scheme using an outer 12-bit RS code (whose complexity is orders of magnitude greater than that of a packet-LDPC code). An extensive simulation study of the floor of the selected LDPC code is also presented in that section. Section V compares the decoder complexity of the various schemes and Section VI draws some concluding remarks.

II. THE ECMA-319 STANDARD

The ECMA-319 standard employs two RS codes concatenated with two levels of interleaving (intra-track and inter-track). As depicted in Fig. 1, user data, after being compressed, is grouped into 404,352-byte chunks which are the ECC-protected entities known as *data sets*. Each data set is then broken into 16 two-dimensional (2-D) arrays called *sub data sets*, each having 25,272 bytes, which will be protected by two different RS encodings, C_1 and C_2 , in turn. The structure of each sub data set is shown in Fig. 1. The data in each row is protected by an inner code, C_1 , an even–odd interleave of two RS codewords. Each row codeword is from a RS(240, 234) code over $\text{GF}(2^8)$, designed to allow correction of a relatively

high background error rate. The data in each column is protected by an outer code C_2 , RS(64, 54) code over GF(2⁸), designed to allow correction of long error bursts (with the aid of the row-column interleaver). Thus, the row code C_1 together with the column code C_2 gives a cross-product ECC design capable of simultaneously correcting both a high background error rate and long error bursts. After encoding, the size of one encoded sub data set is 30, 720 bytes, and the size of one encoded data set is 491, 520 bytes, and the overall code rate is $R_{\text{total}} = R_1 \times R_2 = (234/240) \times (54/64) = 0.82$.

In addition to the row-column interleaver within a sub data set which facilitates the correction of moderate-size error bursts, there is also an interleaver within a data set, i.e., among the 16 sub data sets which facilitates the correction of very long error bursts. This second interleaver distributes the data set uniformly over eight tracks so that, upon playback, the deinterleaver will uniformly distribute the eight tracks among the 16 sub data sets. In the event of a lost track, e.g., due to a head clog or media-defect-induced synchronization loss), the erroneous bytes will be flagged by C_1 as erasures, and will occupy eight rows in each of the 16 sub data set interleaver blocks (Fig. 1). Lastly, since C_2 is capable of correcting ten erasures, it will correct all eight erased rows within each sub data set and hence, an entire lost track.

III. PACKET-LDPC AND TPC-SPC CODES

A. LDPC Codes

There are a large number of classes of LDPC codes. In this paper, the LDPC codes we use are *extended irregular repeat-accumulate* (eIRA) codes ([10], [11]). The eIRA codes are a class of efficiently encodable irregular LDPC codes which have lower error-rate floors than other irregular LDPC code design approaches. It has been demonstrated in [10] that eIRA codes of moderate length and high rate possess no error-rate floor down to probability of bit error $P_b = 10^{-9}$. This class of LDPC codes is an extension of irregular repeat-accumulate codes and has the special parity-check matrix form

$$\mathbf{H} = [\mathbf{H}_1 \quad \mathbf{H}_2] \quad (1)$$

where $\mathbf{H}_1$ is a sparse $m \times k$ matrix randomly generated by computer containing no weight-2 columns and $\mathbf{H}_2$ is a full-rank $m \times m$ matrix whose columns are weight-2 except the last one:

$$\mathbf{H}_2 = \begin{bmatrix} 1 & & & & \\ 1 & 1 & & & \\ & & \dots & & \\ & & & 1 & 1 \\ & & & 1 & 1 \end{bmatrix}. \quad (2)$$

In the above, n is the codeword length, k is the information word length, $m = n - k$ is the parity word length, and H is m -by- n . Details of how H may be designed can be found in [10].

Note that encoding is easily performed directly from H by solving for the parity bits recursively. Alternatively, encoding may be performed via the systematic generator matrix G which is given by [10]

$$\mathbf{G} = [\mathbf{I} \quad \mathbf{P}] = [\mathbf{I} \quad \mathbf{H}_1^T \mathbf{H}_2^{-T}] \quad (3)$$

where

$$\mathbf{H}_2^{-T} = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 & 1 \\ & 1 & \dots & \dots & \dots & 1 \\ & & 1 & \dots & \dots & 1 \\ & & & \dots & \dots & \\ & & & & 1 & 1 \\ & & & & & 1 \end{bmatrix}. \quad (4)$$

Note that $\mathbf{H}_2^{-T}$ is precisely the transformation matrix corresponding to a differential encoder with transfer function $1/(1 \oplus D)$.

B. Packet-LDPC Codes

Most work on LDPC codes has focused on the bit level where a stream of bits is transmitted over a channel that introduces bit-level errors. In this case, encoding and decoding procedures perform operations on bits. For example, in the Tanner graph representation for bit-wise LDPC codes, each variable node corresponds to a bit: the encoder maps each block of k bits into an n -bit codeword. The message-passing decoder iteratively computes probability (or likelihood) for each bit to represent the current belief that the transmitted bit was a zero or one. However, inspired by the idea of packet-based transmission over the Internet, LDPC codes operating on packets instead of bits have been proposed [9], [13]. This coding scheme performs computational operations at the packet level instead of the bit level which makes high speeds possible on many systems. Because the code symbols are packets, decoding complexity for most channels becomes unwieldy. It is a fortunate consequence, however, that decoding is simple on erasure channels and that a tape channel with a reliable erasure-flagging inner code may be modeled as such. In the rest of the paper, we call these packet-wise LDPC codes *packet-LDPC* codes.

The matrix and graphical representations and the encoding and decoding procedures for bit-wise LDPC codes can all be extended to packet-LDPC codes. An (n, k) packet-LDPC code can be specified by its parity check matrix, $\mathbf{H}$, and its packet length L . The $\mathbf{H}$ matrix can be one derived for a bit-wise LDPC code for a binary erasure channel (BEC) with elements H_{ij} from GF(2). Note that a packet-LDPC code is different from a so-called Q -ary LDPC code whose $\mathbf{H}$ matrix has elements from GF(2 ^{L}) [14]. The size of a packet can be hundreds of bits or several tens of megabytes, depending on the application. Each codeword has the form $\mathbf{c} = [\bar{c}_1 \bar{c}_2 \dots \bar{c}_n]$, where $\mathbf{c}$ is an $L \times n$ matrix and $\bar{c}_i$ is a binary L -tuple $[c_i^1 c_i^2 \dots c_i^L]^T$ representing a packet. An (n, k) packet-LDPC code has k user packets and $m = n - k$ parity packets (m is the rank of $\mathbf{H}$), with code rate $R = k/n$. In the Tanner graph specified by $\mathbf{H}$, each variable node corresponds to a packet (symbol) in the codeword, and each check node corresponds to a constraint the code places on adjacent variable packets. The constraints are such that the sum (bit-wise XOR) of the packets associated with the variable nodes adjacent to each check node is $\bar{0}$ (an L -tuple of zeros). Thus, if a check node is connected to w variable nodes $v_1, v_2, \dots, v_w$, the packets $\bar{c}_1, \bar{c}_2, \dots, \bar{c}_w$ associated with these variable nodes satisfy

$$\bar{c}_1 \oplus \bar{c}_2 \oplus \dots \oplus \bar{c}_w = \bar{0}. \quad (5)$$

Encoding is straightforward for a packet-LDPC based on an eIRA-type LDPC code. Since the eIRA packet-LDPC code has an $\mathbf{H}$ matrix of the form in (2), the encoding may be performed directly by solving for the parity packets recursively: 1) the first check equation (first row of $\mathbf{H}$ matrix) involves only the first parity packet; 2) the second check equation involves only the first two parity packets; 3) the third check equation involves only the next two parity packets; and so on. The generator matrix of eIRA bit-wise LDPC derived in (3) can also be used as the generator matrix for eIRA packet-LDPC.

One important property of packet-LDPC codes for our work is that they can be used in concatenated codes. An inner code that works at the bit or byte level can be used on an individual packet or several packets. An outer packet-LDPC code designed to deal with erasures at the packet level is then used to correct failures from the inner code which have been flagged as erasures by the inner decoder. In this paper, we consider concatenations of a packet-LDPC code (outer code) with some other inner codes, where the packet-LDPC sees only correct packets or packets "erased" by the inner code (we say a packet is erased if it is flagged as suspect by the inner decoder). Thus, the decoding algorithm considered here for packet-LDPC codes is for the *packet erasure channel* (PEC).

Iterative decoding algorithm: The packet-LDPC code iterative decoding algorithm over the PEC is essentially identical to that of the bit-wise LDPC code over the binary erasure channel (BEC). The decoder, which relies on the code's Tanner graph, is presented with either correct packets or erased packets, i.e., there are no errors). Consider now decoding iteration j . If a check node is connected to only one packet node with an erasure, with all other connections to correct packets, then the erased packet can easily be resolved via the constraint (5) using packet-wise XOR operations. If a check node is connected to no erased packet nodes, then, of course, no operations are performed. If a check node is connected to multiple erased packet nodes, again no operations are performed since now (5) involves multiple unknowns. In iteration $j + 1$, since some of the erasures had been resolved in the previous iteration (and can now be called "correct"), there are generally some check nodes which previously had multiple erasures and now have a single erasure; these erasures may now be resolved. Eventually, at some iteration, either all erasures will have been resolved, or no erasures will have been resolved. In either case, decoding may stop (clearly in the latter case an additional iteration will be identical to the previous one). A maximum number of decoding iterations may also be set; we chose a maximum of 20 decoding iterations.

C. TPC-SPC Codes

Single parity check turbo product codes (TPC-SPC) are a very simple class of turbo product codes with many good properties [15]. They may be considered to be a type of LDPC code. They have high rate, linear encoding/decoding complexity, and a parallelizable encoding/decoding process. A 2-D turbo product code is formed from two component codes C_1 and C_2 , with parameters $(N_1, N_2, K_1, K_2, G_1, G_2)$, where N, K, G denote the codeword length, information length, and generator matrix, respectively, and the code rate is $R = K_1 K_2 / N_1 N_2$. The component codes used in this paper are single parity check (SPC) codes, so that $N_1 = K_1 + 1$, $N_2 = K_2 + 1$. Since

the encoding operation only adds a single parity bit in each row and column, it is very simple to implement. As shown in [15], although TPC-SPC codes have a very small minimum distance, if several codewords are combined and used with an interleaver and a precoded partial response channel, the distance spectrum improves significantly. Further, the system enjoys interleaving gain. In addition to simple encoding, the advantages of TPC-SPC codes are that, down to a bit-error rate of about 10^{-5} , they perform as well as LDPC codes, but possess better burst error statistics (an error event generally contains a small number of errors), which facilitates decoding for an outer code [15].

IV. PACKET-LDPC CODE-BASED ECMA-319-LIKE CODE ARCHITECTURES

Before we present results for packet-LDPC codes, we briefly mention results for other LDPC code architectures which were studied in [16]. First, we considered replacing the ECMA-319 C_1 (RS) code with a very long (e.g., 30, 720 bits), high rate (e.g., 0.94) LDPC code. We found that, although the soft-decoded LDPC is ostensibly advantageous, it is difficult to find such an LDPC code with an error-rate floor that is sufficiently low for the overall LDPC/RS system to outperform the ECMA-319 RS/RS system at very low error rates.

We considered two alternatives to the LDPC/RS scheme to circumvent the floor issue [16]. In the first, we replaced the LDPC/RS concatenation with a single long (e.g., 65, 535 bits), lower rate (0.82) LDPC code. This approach was very promising, offering a 4 dB gain over the ECMA-319 scheme at $P_{cw} \simeq 10^{-4}$ on an additive white Gaussian noise (AWGN) channel. The second technique we considered to circumvent the floor problem replaced the ECMA-319 C_1 (RS) code with a TPC-SPC code. This scheme provided a gain of about 2.7 dB over the ECMA-319 scheme at $P_{cw} \simeq 10^{-9}$ on EPR4 channel. It is also likely to have a very low floor due to the favorable error statistics of the TPC-SPC code: when it fails to decode, there are generally very few errors remaining for the outer (C_2) RS code to correct. We do not report details of these schemes in this paper due to space limitations and instead favor the more novel packet-LDPC architectures.

A. RS/Package-LDPC Concatenation

In this section, we will compare the performance of outer packet-LDPC codes with that of an outer RS code. In both cases, the inner code C_1 , is a (240, 234) RS code arranged as in the ECMA-319 standard. The outer RS code is (64, 54) and we will consider two packet-LDPC codes to be described below.

It is necessary to choose a packet-LDPC code which is capable of correcting long erasure bursts. To measure a code's burst noise correcting efficacy, we introduce a parameter, the *maximum resolvable erasure-burst length*, $L_{\max}$ [17]. $L_{\max}$ is defined to be the longest string of erasures that the code's decoder is guaranteed to fill in with the correct bit (packet) values, no matter where the burst starts in the codeword. The assumption in this definition is that, outside of the burst, the packets are correct. For bit-wise LDPC codes, the $L_{\max}$ is specified in terms of bits; for packet-LDPC codes, $L_{\max}$ is in terms of packets. Clearly, if we want a code be capable of correcting the erasure of one of eight tracks, we require $L_{\max} > (n/8)$.

TABLE I
RS/PACKET-LDPC CONFIGURATION PARAMETER SUMMARY

C_2	n	k	R_{LDPC}	R_{total}
8K	8192	6717	0.82	0.80
64K	65536	53749	0.82	0.80
C_2	P_{RS}	L	L_{max}	1 track
8K	4	480	1067	1024
64K	32	60	8778	8192

Fig. 2. RS/8K packet-LDPC code structure (one sub data set).

An (n, k) RS code employing s -bit symbols without interleaving is capable of filling in an erasure burst of length $L_{max} = s(n - k)$ bits. With an interleaver of interleaving depth D , an interleaved burst of length $L_{max} = Ds(n - k)$ is correctable. As for LDPC codes, a method to determine L_{max} is given in [17]: for an arbitrary codeword and impose on it an erasure burst of length l (bits) in every location; the largest l for which the decoder successfully decodes at all possible burst locations is L_{max} . This method is clearly time consuming, so an improved algorithm for computing L_{max} is also derived in [17] using the Tanner graph representation of the code and the associated message-passing decoding algorithm. The parameter L_{max} can be maximized by properly permuting the columns of an irregular LDPC code's H matrix. A technique for doing this, called *parity insertion*, is described in [18] for eIRA codes.

The parameters of the two eIRA packet-LDPC codes we consider are listed in Table I. Both cases have the same data set size (3840K bits, where $K = 1024$) and sub data set size (240K bits) as that of the ECMA-319 standard. There are still 16 sub data sets in one data set; the only difference is that the C_2 RS code is replaced by the packet-LDPC codes.

1) *8K Packet-LDPC*: Fig. 2 shows the structure of one sub data set for the RS/8K packet-LDPC configuration. Each RS codeword contains four 480-bit packets. Thus, there are $64 \times 4 \times 2 = 512$ packets in one sub data set and $512 \times 16 = 8192$ packets in one data set. When the C_1 decoder fails to decode a RS codeword, it erases the four packets within this codeword. Then, the C_2 decoder follows the erasure decoding algorithm described in [19]. Note that L_{max} for this 8K LDPC code is larger than the length of one track (Table I), so it can resolve one erased track of eight.

The performance of this scheme in terms of probability of packet error (P_{pk}) and probability of data set error (P_{ds}) is presented in Fig. 3 together with the P_{ds} of the ECMA-319

Fig. 3. Performance of RS/8K packet-LDPC array versus ECMA-319 over AWGN channel.

Fig. 4. RS/64K packet-LDPC code structure (one sub data set).

scheme.¹ Both schemes were simulated on an AWGN channel. For the AWGN-only case, this configuration has a 0.7 dB gain in terms of P_{ds} at 10^{-5} relative to the RS/RS concatenation. For the AWGN-plus-one-erased-track case, there is a 0.8 dB gain at 10^{-4} .

2) *64K Packet-LDPC*: In the 64K packet-LDPC code scheme, there are fewer bits in each packet and more packets in each RS codeword, resulting in a longer packet-LDPC code length (Fig. 4). The L_{max} of this LDPC code is larger than the length of one track, so it can resolve one erased track of eight.

The performance for this code is presented in Fig. 5 and compared with the P_{ds} of the ECMA-319. The simulations were conducted on an AWGN channel. Note that this configuration has a 0.7 dB gain in terms of P_{ds} at 10^{-5} over the RS/RS code array for the AWGN-only case, and a 0.6 dB gain at 10^{-4} for the AWGN-plus-one-erased-track case. Upon comparing the performance of the two packet-LDPC codes, we can see that they perform comparably, although the 8K code is slightly better in the one-erased-track case. Hence, we will favor the 8K code hereafter for its shorter length, hence lower complexity.

3) *On the Packet-LDPC Code Floor*: At the beginning of this section, we have mentioned that configurations using LDPC codes may suffer from an error-rate floor. In Fig. 6, we present

¹ $P_{ds,ECMA-319} = 1 - (1 - P_{cw}^{II})^N \approx NP_{cw}^{II}$, where P_{cw}^{II} is the probability of codeword error for C_2 and N is the number of C_2 codewords in each data set ($N = 480 \times 16 = 7680$).

Fig. 5. Performance of RS/64K packet-LDPC array versus ECMA-319 over AWGN channel.

Fig. 6. 8K packet-LDPC code on PEC.

the performance of the 8K packet-LDPC code on the PEC for various numbers of iterations (for the PEC, a packet is either received correctly or it is erased). The simulation results show that the 8K packet-LDPC code has no floor down to about $P_{\text{packet}} = 10^{-11}$ (since the decoder is very simple, it is possible to run the simulation down to very low error rate). Fig. 7 presents the performance on the PEC-plus-one-erased-track channel, and there is no floor down to about $P_{\text{packet}} = 10^{-9}$. These results demonstrate that this packet-LDPC code has good error-rate floor characteristics, and it is as effective in correcting erasure bursts as the RS- C_2 code in the ECMA-319 architecture. Also, as a practical concern, ten decoding iterations appears to be sufficient.

We remark that the capacity of a PEC with erasure probability ρ is $C = 1 - \rho$. We note from Fig. 6 with $\rho = 0.125$, the (8192, 6717) packet-LDPC code achieves essentially error-free communication (given a sufficient number of decoding iterations). Thus, this code achieves an efficiency relative to capacity of $R/C = 0.82/0.875 = 93.7\%$. Another measure of the code's efficiency, regarding the correction of correcting burst erasures, is $L_{\text{max}}/(n - k) = 1067/1475 = 72.2\%$.

Fig. 7. 8K packet-LDPC code on PEC plus one erased track of eight.

Fig. 8. One data set of 8-bit/12-bit RS/RS concatenation.

B. Longer RS Codes

We have seen the improvement attained by using longer LDPC (packet-LDPC) codes instead of short ones. To be fair to RS codes, we consider a longer (larger block length and larger alphabet size) outer RS code here. The RS codes in the ECMA-319 standard and the above section are all shortened RS codes over $\text{GF}(2^8)$, i.e., each symbol has $s = 8$ bits, and the codeword length is given by $n \leq 2^8 - 1 = 255$. In this section, we consider RS codes over $\text{GF}(2^{12})$, with $n \leq 2^{12} - 1 = 4095$. (RS codes with a larger value of s will likely be at the limits of acceptable complexity.) We want the overall code rate and the data set size be the same as those of ECMA-319. We also want to maintain the capability of correcting one erased track out of eight.

Although we have not exhaustively examined the performance of all possible ECMA-319-compatible architectures involving a 12-bit RS code, we believe we have found a near-optimal configuration. Fig. 8 presents the organization of this RS/RS data set which uses a 12-bit RS code for C_2 . In this scheme, C_1 is a RS(120, 114) code over $\text{GF}(2^8)$ along each row, and C_2 is an extended [20] RS(4096, 3546) over $\text{GF}(2^{12})$ along columns. Note that there are 120×8 bits in each row (C_1), i.e., 80 symbols with 12 bits in each symbol, so there are 80 columns in each data set. Recall that in ECMA-319, the encoded data set size is $64 \times 480 \times 16 \times 8$ bits, which is equal to the encoded data set size of this longer RS/RS scheme

Fig. 9. Performance of RS/RS array with 12-bit symbol RS codes versus RS/8K packet-LDPC array.

($4096 \times 80 \times 12$ bits). The overall code rate for ECMA-319 is $(54/64) \times (234/240) = 0.82266$, which is also approximately equal to $(3546/4096) \times (114/120) = 0.82244$. Also the outer RS code C_2 has the erasure-correcting capability $\lambda = n - k = 4096 - 3546 = 550$ symbols $> n/8 = 512$ symbols, which means it is capable of correcting one erased track.

Fig. 9 presents the performance (in terms of P_{ds}) of the 8-bit/12-bit RS/RS code array in comparison with the RS/8K-packet LDPC code array. For the AWGN-only case, the RS/RS longer code concatenation has about a 0.1 dB gain over the RS/8K-packet LDPC concatenation at around $P_{ds} = 10^{-7}$. For the AWGN-plus-one-erased-track case, the RS/RS concatenation has about a 0.4 dB gain over the RS/packet-LDPC scheme at around $P_{ds} = 10^{-4}$. We note that the 12-bit RS code possesses superior erasure-burst correcting capability, but its encoding and decoding complexity is quite large which discourages its use. The packet-LDPC codes achieve similar performance, but have much lower complexity than the 12-bit long RS code as will be shown in Section V.

C. TPC-SPC/Package-LDPC Concatenation

At the beginning of this section we mentioned that the TPC-SPC/RS concatenation (replacing the C_1 -RS code with a TPC-SPC code) provides a performance improvement over the ECMA-319 RS/RS concatenation. Above, we showed that replacing the C_2 -RS code with packet-LDPC code also results in improved performance in the tape recording system considered. We now examine a coding scheme which replaces the C_1 -RS and C_2 -RS codes with TPC-SPC and packet-LDPC codes, respectively.

As stated in Section IV-A, the packet-LDPC code we favor is the 8K code. C_1 is the TPC-SPC code, but to reduce the data set size and to make it nearly compatible with that of the ECMA-319 standard, we need to adjust the size of each TPC array and the number of arrays combined. Also, the number of bits in each row (length of C_1) needs to be a multiple of 8. The TPC-SPC code we choose here has parameters $2 \times (32, 31)^2 = (2048, 1922)$. Fig. 10 shows the organization of one data set. We see that this configuration has a data set size of 4096K bits as compared to 3840K bits for the ECMA-319.

Fig. 10. TPC-SPC/8K packet-LDPC code structure (one sub data set).

TABLE II
TPC-SPC/PACKET-LDPC CODE ARRAY PARAMETERS

$R_{\text{TPC-SPC}}$	R_{LDPC}	R_{total}	$\frac{PK}{TPC}$	L (bits)	DataSet(bits)
0.9385	0.8200	0.7696	4	512	4096K

Fig. 11. Probability of undetected error for TPC-SPC code on EPR4 channel.

Table II presents the parameters of this coding array. Each TPC-SPC codeword has four packets, with 512 bits in each packet. As we mentioned in Section III-C, TPC-SPC codes have a relatively small $d_{\min}$, and hence a high probability of undetected error (P_{ud}), i.e., the probability that the decoder outputs a codeword different from the transmitted codeword without recognizing its mistake. Fig. 11 presents the P_{ud} of the TPC-SPC code on EPR4 channel from which we see the error detection capability of this code is not reliable. Hence, a *cyclic redundancy check* (CRC) code is used on each packet to give the outer packet-LDPC decoder reliable (and localized) information. As an example, one may use a rate-0.968 CRC (496, 480) code and suffer negligible rate loss. Note that with the help of the CRC code, we can know exactly which packets are erroneous (hence, erased), thus we do not need to erase all four packets if C_1 fails.

The simulation results for this configuration is compared with that of the ECMA-319 system in Fig. 12, assuming the EPR4 channel in each case. In this figure, we also compare the performance of the TPC-SPC/packet-LDPC concatenation with a longer RS/RS concatenation we introduced in Section IV-B. This longer RS/RS code array has C_1 : RS(126, 113) $s = 8$ -bit symbol codes along rows and C_2 : RS(4096, 3546) $s = 12$ -bit symbol codes along columns. Note that, each row has $126 \times$

Fig. 12. Performance of TPC-SPC/packet-LDPC array versus ECMA-319 over EPR4 channel.

8 bits, i.e., $(126 \times 8)/12 = 84$ 12-bit symbols, so there are 84 C_2 codewords in each data se; thus, we have

$$P_{ds, \text{ long RS/RS}} = 1 - \left(1 - P_{cw, \text{ long RS/RS}}^{II}\right)^{84}. \quad (6)$$

Note also that the overall code rate of this longer RS/RS concatenation is $(113/126) \times (3546/4096) = 0.7764 \simeq 0.7696$, and the encoded data set size is $4096 \times 84 \times 12 = 4032K$ bits $\simeq 4096K$ bits. All parameters are compatible with the TPC-SPC/packet-LDPC concatenation (Table II). These results demonstrate that this TPC-SPC/packet-LDPC coding architecture is very promising as it provides a substantial improvement over the ECMA-319 code array on the EPR4 channel, and also beats the longer RS/RS code array. Further, as demonstrated in Figs. 6 and 7, the error-rate floor does not appear to be an issue.

V. COMPLEXITY COMPARISON

It has been shown in many papers that the use of iterative detection and soft information can provide significant performance improvement. However, the application of this technology has been limited by its complexity. Lynch *et al.* have shown in [21] that the implementation of a soft iterative decoder for structured LDPC or TPC-SPC codes are feasible in CMOS technology, but it is at the expense of an N -fold increase in complexity relative to the state of art ($N \simeq 6$). As we will show, packet-LDPC codes may be a possible solution for the complexity problem (while trading off performance). In this section, we will compare the complexity of RS code and packet-LDPC code.

As we know, RS codes are optimal with respect to recovering erasures, as they are a class of MDS codes, so we cannot hope for better erasure recovery than that given by RS codes. However, a code with a little bit lower erasure correction capability, but much lower encoding and decoding complexity is desirable for practical use. We have demonstrated that packet-LDPC codes have excellent erasure correction capability. We will compare the complexity of packet-LDPC codes and RS codes in this section.

TABLE III
COMPLEXITY COMPARISON

	Packet-LDPC	RS
Decoding inefficiency	$1 + \epsilon$	1
Encoding (#operation/codeword)	$n \ln(1/\epsilon) L$	$k(n-k)s$
Decoding (#operation/codeword)	$n \ln(1/\epsilon) L$	$k(n-k)s$
Basic operation	XOR	$GF(2^s)$ field operation

For an (n, k) RS code over $GF(2^s)$, the encoding procedure is to calculate $n - k$ parity symbols from the k information symbols. Thus, obviously, the encoding time is proportional to $k(n - k)s$. For decoding RS erasure codes, it is known that a codeword can be recovered from any k or more symbols that are received correctly. To analyze the complexity of an erasure decoder, we consider the special case of exactly k symbols being received correctly. In this case, the decoder needs to recover the $n - k$ symbols that are erased, i.e., to solve a system with $n - k$ unknowns from the k known symbols, which has decoding time proportional to $k(n - k)s$.² The basic encoding and decoding operations for RS codes are multiplications and additions over $GF(2^s)$. From the analysis, we can see the large processing time for RS codes arises from the dense system of linear equations used, and RS codes can only be applied in practice when k and $n - k$ are relatively small.

For an (n, k) packet-LDPC code with packet length L , to process encoding or decoding of a codeword on PEC is equivalent to processing L bit-wise LDPC codewords on BEC with the same erasure positions. The encoding and decoding procedures on the PEC are almost symmetrical. Both are very simple, which compute exactly L XORs for each edge in the Tanner graph. Hence, the number of operations are proportional to the number of edges in graph. And because a packet-LDPC code has a sparse graph, the algorithm runs in time linear in the block length times the packet length of the code. Furthermore, for packet-LDPC codes, k packets no longer suffice to recover the information packets, instead, slightly more than k packets are needed. As shown in [19] and [9] for Tornado codes, if a packet-LDPC code can be recovered with high probability from $(1 + \epsilon)k$ or more packets, where ϵ is a real number and $(1 + \epsilon)$ is defined as *decoding inefficiency*, the encoding and decoding algorithms run in time proportional to $n \ln(1/\epsilon)L$. Here, the basic operations are binary XORs. For RS codes, the decoding inefficiency is 1 with $\epsilon = 0$. For packet-LDPC codes, the decoding inefficiency is not a fixed quantity, but depends on the structure of the graph and the packet erasure pattern (the positions of erasures). However, the variance of ϵ is generally very small. We will show in the next paragraph that packet-LDPC codes have small ϵ (around 0.07 for the $8K$ packet-LDPC code). The encoding and decoding complexity of both RS codes and packet-LDPC codes, measured in *operations per codeword*, are shown in Table III [9].

We now substitute into the equations in Table III the values for: 1) the $8K$ packet-LDPC code in Section IV-A; 2) the

²We remark that both encoding and decoding complexity are proportional to $k(n - k)s$ since it is an erasure decoder. An errors-only or an errors-and-erasures decoder would be much complex.

TABLE IV
COMPLEXITY COMPARISON

	Packet-LDPC	RS $s = 8$	RS $s = 12$
$1 + \epsilon$	$1 + 0.0732$	1	1
# of codewords data set	1	480×16	80
Encoding (#operation data set)	1.03×10^7	3.32×10^7	1.87×10^9
Decoding (#operation data set)	1.03×10^7	3.32×10^7	1.87×10^9
Basicoperation	XOR	$GF(2^8)$ field	$GF(2^{12})$ field

RS(64, 54) $s = 8$ code used as C_2 in the ECMA-319 standard; and 3) the RS(4096, 3546) $s = 12$ code discussed in Section IV-B. From Fig. 6 we see, for the 8K packet-LDPC code with ten decoding iterations, when the packet loss rate $\rho = 0.12$, the P_{ds} (also the P_{cw}) is around 10^{-12} , so we can say the code can be recovered with high probability. At this point, the average number of erased packets is ρn , and the average number of correctly received packets is $(1 - \rho)n = (1 + \epsilon)k$, from which we get $\epsilon = (1 - \rho)(n/k) - 1 = 0.0732$.

Notice that the complexity of a data set equals to the number of codewords per data set times the complexity of a codeword. The complexity of encoding and decoding a data set using the above three codes are listed in Table IV. The table shows that, even if we temporarily ignore the difference between binary XOR and field operation (field operation is obviously much more complex than XOR), the number of operations required for encoding and decoding the RS(64, 54) $s = 8$ code is about three times of that of the 8K packet-LDPC code; and the RS(4096, 3546) $s = 12$ code requires about 200 times more operations than the 8K packet-LDPC code.

At this point, we can conclude that, as we claimed, packet-LDPC codes on the PEC have near channel capacity erasure correction capability and very low encoding and decoding complexity compared to RS codes, especially long RS codes.

VI. CONCLUSION

In this paper, we have introduced two packet-LDPC code architectures for tape drives. We compared the performance of these schemes with that of the RS/RS concatenation in the ECMA-319 standard and the RS/RS concatenation with a longer outer RS code. These two architectures, especially the second one, provide a substantial performance improvement over the ECMA-319 architecture. Simulations show that the error-rate floor usually seen for LDPC codes does not appear to be a problem for our packet-LDPC codes. We have demonstrated that a packet-LDPC code allows for a very long outer LDPC code in the concatenated scheme (hence, low error-rate floor), and solves the problem that bit-wise LDPC codes are not as effective at correcting burst errors as RS codes. Moreover, packet-LDPC codes have a very low decoding complexity, as we demonstrated in Section V.

ACKNOWLEDGMENT

This work was supported in part by the INSIC Tape program. The authors would like to thank G. Jaquette and B. Hutchins of IBM for helpful discussions.

REFERENCES

- [1] J. Fan, E. Kurtas, A. Friedmann, and S. McLaughlin, "Low density parity check codes for magnetic recording," in *Proc. 37th Allerton Conf.*, Sep. 1999, pp. 1314–1323.
- [2] B. Vasic, "Structured iteratively decodable codes based on Steiner systems and their application in magnetic recording," in *Proc. IEEE Globecom*, vol. 5, Nov. 2001, pp. 2954–2960.
- [3] H. Song, R. M. Todd, and J. R. Cruz, "Applications of low-density parity check codes to magnetic recording channels," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 5, pp. 918–923, May 2001.
- [4] N. Varnica and A. Kavcic, "Optimized LDPC codes for partial response channels," in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, Jun. 30–Jul. 5 2002, p. 197.
- [5] J. Park and J. Moon, "A new soft-output detection method for the magnetic recording channel," in *Proc. IEEE Globecom*, vol. 5, Nov. 2001, pp. 3002–3006.
- [6] B. M. Kurkoski, P. H. Siegel, and J. K. Wolf, "Joint message passing decoding of LDPC coded partial-response channels," *IEEE Trans. Inf. Theory*, vol. 48, no. 6, pp. 1410–1422, Jun. 2002.
- [7] H. Song and B. V. K. V. Kumar, "Low-density parity check codes for partial response channels," *IEEE Signal Process. Mag.*, vol. 21, no. 1, pp. 56–66, Jan. 2004.
- [8] S. Sahu, H. Song, and B. V. K. V. Kumar, "Performance of low density parity check (LDPC) codes on high-density magnetic tape recording signals," in *INTERMAG Conf.*, Mar. 28–Apr. 3 2003, p. DT-10.
- [9] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach reliable distribution of bulk data," in *Proc. ACM SIGCOMM*, vol. 28:4, Sep. 1998, pp. 56–67.
- [10] M. Yang, W. E. Ryan, and Y. Li, "Design of efficiently encodable moderate-length high-rate irregular LDPC codes," *IEEE Trans. Commun.*, vol. 52, no. 4, pp. 564–571, Apr. 2004.
- [11] M. Yang and W. E. Ryan, "Lowering the error-rate floors of moderate-length high-rate irregular LDPC codes," in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, Jun. 29–Jul. 4 2003, p. 237.
- [12] W. E. Ryan, "An introduction to LDPC codes," in *CRC Handbook for Coding and Signal Processing for Recording Systems*, B. Vasic, Ed. Boca Raton, FL: CRC, 2004.
- [13] M. Luby and M. Mitzenmacher, "Verification codes: simple low-density parity-check codes for large alphabets," in *Proc. 40th Annu. Allerton Conf. Communication, Control, and Computing*, 2002, pp. 38–47.
- [14] M. C. Davey and D. J. C. MacKay, "Low density parity check codes over $GF(q)$," *IEEE Commun. Lett.*, vol. 2, pp. 165–167, Jun. 1998.
- [15] J. Li, K. R. Narayanan, and C. N. Georgiades, "On the performance of high-rate TPC/SPC codes and LDPC codes over partial response channels," *IEEE Trans. Commun.*, vol. 50, no. 1, pp. 723–734, May 2002.
- [16] Y. Han, "Design and analysis of LDPC code architectures for tape drives," M.S. thesis, Dept. ECE, Univ. Arizona, Tucson, Jun. 2004.
- [17] M. Yang and W. E. Ryan, "Performance of efficiently encodable LDPC codes in noise bursts on the EPR4 channel," *IEEE Trans. Magn.*, vol. 40, no. 2, pp. 507–512, Mar. 2002.
- [18] M. Yang, "Construction of LDPC codes for data storage and transmission," Ph.D. dissertation, Dept. ECE, Univ. Arizona, May 2004.
- [19] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Efficient erasure correcting codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 569–584, Feb. 2001.
- [20] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Englewood Cliffs, NJ: Prentice-Hall, Jan. 15, 1995.
- [21] R. Lynch, E. M. Kurtas, A. Kuznetsov, E. Yeo, and B. Nikolic, "The search for a practical iterative detector for magnetic recording," *IEEE Trans. Magn.*, vol. 40, no. 1, pp. 213–218, Jan. 2004.

Manuscript received June 27, 2004; revised January 12, 2005.