Bounds on the Minimum Distance of Punctured Quasi-Cyclic LDPC Codes

Brian K. Butler, *Senior Member, IEEE*, and Paul H. Siegel, *Fellow, IEEE*

***Abstract*—Recent work by Divsalar *et al.* has shown that properly designed protograph-based low-density parity-check codes typically have minimum (Hamming) distance linearly increasing with block length. This fact rests on ensemble arguments over all possible expansions of the base protograph. However, when implementation complexity is considered, the expansions are frequently selected from a smaller class of structured expansions. For example, protograph expansion by cyclically shifting connections generates a quasi-cyclic (QC) code. Other recent work by Smarandache and Vontobel has provided upper bounds on the minimum distance of QC codes. In this paper, we generalize these bounds to punctured QC codes and then show how to tighten these for certain classes of codes. We then evaluate these upper bounds for the family of protograph codes known as AR4JA codes that have been recommended for use in deep space communications in a standard established by the Consultative Committee for Space Data Systems. At block lengths larger than 4400 bits, these upper bounds fall well below the ensemble lower bounds.**

***Index Terms*—Binary codes, block codes, error correction codes, linear codes, sparse matrices.**

I. Introduction

Low-DENSITY parity-check (LDPC) codes originated in the seminal work by Gallager [1] over 50 years ago. The study of these codes remained largely dormant for decades, with the important exception of Tanner's work on graph-based code constructions [2]. At low SNR, properly designed LDPC codes exhibit good performance with practical, iterative message-passing decoders. However, at higher SNRs, they may suffer from an abrupt change in the slope of the error-rate curve, a phenomenon known as an error floor. The floor can be attributed, in part, to the existence of certain properties of the Tanner graph that is associated with a chosen parity-check matrix and upon which the decoder operates. Techniques that reduce the occurrence of short cycles in the Tanner graph, for example [3], [4], have been shown to mitigate the error floor phenomenon. Specifically, the ACE algorithm [5] for placing edges in a graph-based code brings down the error floor substantially by preventing short cycles from clustering around low-degree variable nodes.

Another property of a code that limits its error performance at high SNR is the minimum (Hamming) distance between codewords. The minimum distance is also important in understanding the likelihood of undetected errors, a critical concern in many applications. Yet, relatively little attention has been paid to analyzing the minimum distance of LDPC codes and to developing LDPC code design methodologies that ensure large minimum distance. MacKay and Davey introduced upper bounds on the minimum distance for a class of codes that included quasi-cyclic (QC) LDPC codes in [6]. Notable later work appears in [7], [8]. Of particular relevance to this study are the upper bounds of Smarandache and Vontobel [9] which allow for more variation in the underlying protograph used to represent the code.

Another line of research has shown that most codes in the ensemble of protograph-based codes characterized by a limited number of degree-2 variable nodes have minimum distance that increases linearly with block length [10]–[12]. The family of LDPC codes known as AR4JA codes, recommended for deep-space communications by the Consultative Committee for Space Data Systems (CCSDS) [13], are obtained by puncturing QC-LPDC codes designed from protographs in this ensemble. The selected protographs are expanded to the actual codes (in two stages) using the ACE algorithm to place the edges with QC constraints.

In this paper, we extend the bounds of [9] to the general class of punctured QC-LDPC codes and show that these bounds can be tightened in cases where the protomatrices associated with the underlying protograph contain many zero entries. Much of our methodology parallels [9], with our extensions motivated by an interest in bounding the actual minimum distance of the AR4JA codes specified in the CCSDS standard. Somewhat surprisingly, the application of our methodology to the protomatrices underlying the AR4JA constructions for code rates $1/2, 2/3$, and $4/5$ yields upper bounds of 66, 58, and 56, respectively, independent of the code block length. For large block lengths, these bounds fall well short of the linearly growing ensemble lower bound mentioned above. Finally, using slight modifications of previously proposed search techniques, we identify specific codewords for each of these code rates at two of the standardized code lengths. The weights of these codewords validate the upper bounds and suggest that the upper bounds may be fairly tight.

The remainder of this paper is organized as follows. Section II provides a review of protograph-based LDPC code design, as

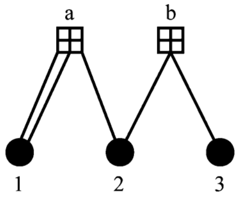
Manuscript received January 11, 2012; revised February 20, 2013; accepted February 25, 2013. Date of publication March 19, 2013; date of current version June 12, 2013. This work was supported in part by the Center for Magnetic Recording Research at UCSD and by the National Science Foundation under Grants CCF-0829865 and CF-1116739. The material in this paper was presented in part at the IEEE International Symposium on Information Theory, Austin, TX, USA, June 2010.

The authors are with the Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093 USA (e-mail: butler@ieee.org; psiegel@ucsd.edu).

Communicated by P. O. Vontobel, Associate Editor for Coding Techniques.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIT.2013.2253152



$$\mathbf{A} = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

Fig. 1. Simple protograph G and corresponding protomatrix $\mathbf{A}$.

well as the specific family of AR4JA codes. Section III provides the necessary mathematical background on the polynomial representation and properties of QC-LDPC codes obtained by the expansion of protographs in which the expansion is based on circulant matrices. In Section IV, we review the upper bounds on the minimum-distance of QC-LDPC codes in [9], and then develop the necessary algebraic results to generalize these bounds to punctured QC-LPDC codes. Section V describes techniques that can produce tighter upper bounds for protographs with specific properties, including some of the AR4JA protographs. In Section VI, we apply our methods to calculate upper bounds on the minimum distance of the codes in the CCSDS standard, which are obtained by a two-step QC expansion of the AR4JA protographs. In Section VII, we use computer search to find low-weight codewords for several AR4JA codes, compare them to our length-independent bounds, and reconcile them with the ensemble minimum-distance lower bounds [10]–[12]. We also examine the girth of AR4JA codes. Section VIII concludes this paper.

II. Protographs and the AR4JA Family of Codes

Protographs were introduced as a way to impart structure to the interconnectivity of graph-based codes [14]. Protographs themselves are a subset of multiedge type graphs [15, ch.7].

A *protograph* is essentially a Tanner graph with a relatively small number of nodes. More specifically, a protograph, $G = (V, C, E)$, consists of a set of variable nodes V, a set of check nodes C, and a collection of edges E. Each edge, $e \in E$, connects a variable node, $v_e \in V$, to a check node, $c_e \in C$. Protographs have the additional property that parallel edges are permitted. Moreover, variable nodes in V can be designated as punctured; *i.e.*, the corresponding bits are not included in the transmitted codeword.

A simple protograph G with three variable nodes, two check nodes, and five edges is shown in Fig. 1. The accompanying *protomatrix* $\mathbf{A}$ fully describes the protograph structure. The entry in the jth row and ith column of the protomatrix $\mathbf{A}$ indicates the number of edges connecting the jth check node to the ith variable node within the corresponding protograph.

A *derived graph* is constructed by replicating the protograph a specified number of times and interconnecting the copies of the variable and check nodes in a manner consistent with the topology of G. In this example, all copies of check node a are called “type a” check nodes. Similarly, all copies of variable node 1 are called “type 1” variable nodes. The replicas of an edge connecting check node a and variable node 1 form a so-called edge set, and their connected check nodes may be permuted within the set of “type a” check nodes. (The term “type,” when used in Section VI to classify matrices as in [9], is unrelated.) The corresponding linear code is referred to as a *protograph code*.

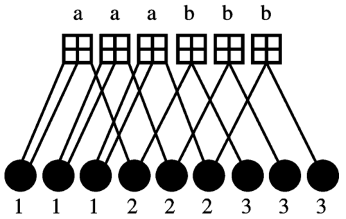


Fig. 2. Protograph G replicated $N = 3$ times.

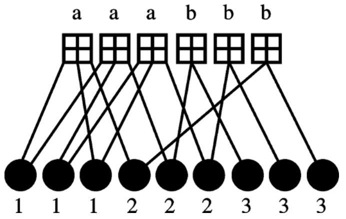


Fig. 3. Derived graph obtained by edge set permutations that preserve degree and interconnectivity among node types.

The application of this interconnection procedure to all replicas ensures that node degrees and connectivity by node types of the original protograph are maintained in the resulting derived graph. The corresponding iterative decoder implementation may, in fact, be less complex than that of “random” LDPC codes because of the structure imposed on the node interconnections.

Figs. 2 and 3 illustrate the process of making $N = 3$ copies of the protograph of Fig. 1 and interconnecting them to generate the derived graph. The parity-check matrix corresponding to the derived graph of Fig. 3 is shown below, divided into submatrices so that the relationship to the protomatrix $\mathbf{A}$ of Fig. 1 is evident:

$$\mathbf{H} = \left[\begin{array}{ccc:ccc:ccc} 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hdashline 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{array}\right].$$

Protograph-based code design allows for the introduction of degree-one variable nodes and punctured variable nodes in a structured way. With regard to degree-one nodes, recall that the optimization of irregular LDPC codes by density evolution typically avoids degree-one variable nodes. This is because they induce an error rate floor in randomly constructed codes, even as the block length grows toward infinity [15, p. 161]. However, density evolution often produces a significant fraction of degree-2 variable nodes. This suggests that the incorporation of degree-one variable nodes may offer a potential benefit in code performance, as noted in [15, p. 382].

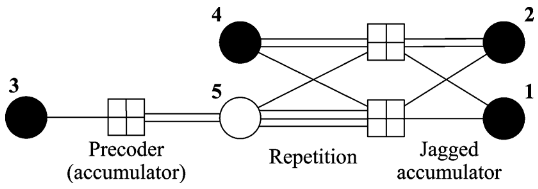


Fig. 4. AR4JA protograph, rate-$1/2$.

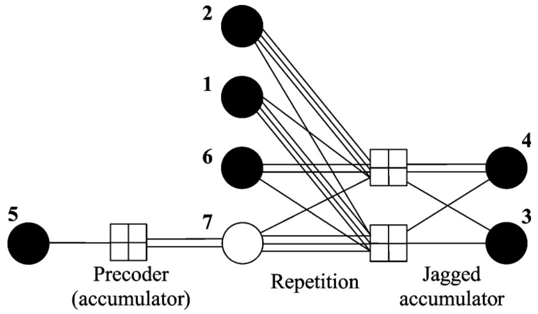


Fig. 5. AR4JA protograph, rate-$2/3$.

The protograph for the rate $r = 1/2$ AR4JA code [10] is shown in Fig. 4. We follow the convention of representing the transmitted variable nodes as solid circles and the punctured variable nodes as unfilled circles. The protograph of the rate-$1/2$ code is extended to rate-$2/3$ by adding two degree-4 variable nodes, as shown in Fig. 5. The corresponding protomatrices are

$$\mathbf{A}_{r=1/2} = \begin{bmatrix} 0 & 0 & 1 & 0 & 2 \\ 1 & 1 & 0 & 1 & 3 \\ 1 & 2 & 0 & 2 & 1 \end{bmatrix} \tag{1}$$

and

$$\mathbf{A}_{r=2/3} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 2 \\ 3 & 1 & 1 & 1 & 0 & 1 & 3 \\ 1 & 3 & 1 & 2 & 0 & 2 & 1 \end{bmatrix} \tag{2}$$

respectively. The numerical labels of the variable nodes in the figures correspond to the columns in the protomatrices, enumerated from left to right, and the check nodes correspond to the rows of the protomatrix.

The AR4JA family of protographs further extends the code rate options by adding an additional four degree-4 variable nodes. The rate-$4/5$ protograph has 11 variable nodes altogether, as can be seen from its protomatrix,

$$\mathbf{A}_{r=4/5} = \begin{bmatrix} 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 2 \\ 3\ 1\ 3\ 1\ 3\ 1\ 1\ 1\ 0\ 1\ 3 \\ 1\ 3\ 1\ 3\ 1\ 3\ 1\ 2\ 0\ 2\ 1 \end{bmatrix}. \tag{3}$$

For all of these rate options, the variable nodes in the derived graph that correspond to copies of the degree-6 variable node in the protograph (equivalently, the right-most column of the protomatrix) are punctured.

The name *AR4JA* is derived from the operations reflected in the protograph structure and indicated in Figs. 4 and 5. As can be seen, the protograph embodies several features similar to those of an Accumulate–Repeat–Accumulate code. For the AR4JA code construction, a partial precoding by accumulation (A) is followed by "repetition four times" (R4), culminating in a "jagged" accumulation (JA). The jagged accumulation differs from a standard accumulation, which contains degree-2 variable nodes only, by the additional edge at the upper right of the protograph. In fact, the switch from a standard accumulation stage to the jagged accumulation stage, which reduces the number of degree-2 variable nodes, allows the AR4JA protographs to meet the criterion of the ensemble of protographs with linearly increasing minimum distance. More specifically, the techniques of Divsalar and other researchers [10]–[12], [16] can be used to calculate the asymptotic ensemble weight enumerators for protograph-based codes, from which the *typical relative minimum distance* $\delta_{\min}$ can be found. They prove that the minimum distance $d_{\min}$ of most of the codes in the ensemble derived from the protograph exceeds $\delta_{\min} n$, where n is the block length of the code. For the rate-$1/2$ AR4JA protomatrix in (1), $\delta_{\min} = 0.015$ [10].

III. QC Expansion and Polynomial Representation

The codewords of a block code may be divided into nonoverlapping *subblocks* of N consecutive symbols. A *quasi-cyclic* (QC) code is a linear block code having the property that applying identical circular shifts to every subblock of a codeword yields a codeword. QC codes are a generalization of conventional cyclic block codes and are simple to encode [17, Sec. 8.14].

A binary QC-LDPC code of length $n = LN$ can be described by an $m \times n$ sparse parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{m \times n}$, with $m = JN$, which is composed of $N \times N$ circulant submatrices. A right *circulant matrix* is a square matrix with each successive row right-shifted circularly one position relative to the row above. Therefore, circulant matrices can be completely described by a single row or column. As in [9], we use the description corresponding to the left-most column.

A binary QC-LDPC code can also be described in polynomial form, since there exists an isomorphism between the commutative ring of $N \times N$ circulant binary matrices and the commutative ring of binary polynomials modulo $x^N - 1$, *i.e.*, $\mathbb{F}_2[x]/\langle x^N - 1\rangle$. Addition and multiplication in the latter ring correspond to, respectively, addition and multiplication of polynomials in $\mathbb{F}_2[x]$, modulo $x^N - 1$.

The isomorphism between $N \times N$ binary circulant matrices and polynomial residues in the quotient ring maps a matrix to the polynomial in which the coefficients in order of increasing degree correspond to the entries in the left-most matrix column taken from top to bottom. Under this isomorphism, the $N \times N$ identity matrix maps to the multiplicative identity in the polynomial quotient ring, namely 1. A few examples of the mapping (indicated by $\mapsto$) for $N = 3$ are shown as follows:

$$\begin{bmatrix} 1\,0\,0 \\ 0\,1\,0 \\ 0\,0\,1 \end{bmatrix} \mapsto 1 \qquad \begin{bmatrix} 0\,0\,1 \\ 1\,0\,0 \\ 0\,1\,0 \end{bmatrix} \mapsto x \qquad \begin{bmatrix} 1\,1\,0 \\ 0\,1\,1 \\ 1\,0\,1 \end{bmatrix} \mapsto 1 + x^2.$$

This isomorphism requires that care be taken when representing multiplication of a circulant matrix $\mathbf{M}$ by a binary vector $\mathbf{v} = (v_0, v_1, \ldots, v_{N-1})$. If we associate the polynomial $M(x)$ with the matrix $\mathbf{M}$ under the isomorphism just described, and let $v(x) = v_0 + v_1 x + \cdots + v_{N-1}x^{N-1}$ represent the vector $\mathbf{v}$, then the product $\left(\mathbf{M}\mathbf{v}^T\right)^T = \mathbf{v}\mathbf{M}^T$ maps to the polynomial

$M(x)v(x)$ modulo $x^N - 1$, and the product $\mathbf{vM}$ maps to the polynomial $x^N M(x^{-1})v(x)$ modulo $x^N - 1$. Note that from this paragraph onwards, all indexing begins at zero and all vectors are row vectors.

Given a polynomial residue $a(x) \in \mathbb{F}_2[x]/\langle x^N - 1\rangle$, we define its weight $\mathrm{wt}(a(x)) \in \mathbb{Z}$ to be the number of nonzero coefficients. Thus, the weight $\mathrm{wt}(a(x))$ of the polynomial $a(x)$ is equal to the Hamming weight $w_{\mathrm{H}}(\mathbf{a})$ of the corresponding binary vector of coefficients $\mathbf{a}$. For a length-L vector of elements in the ring, $\mathbf{a}(x) = (a_0(x), a_1(x), \ldots, a_{L-1}(x))$, we define its Hamming weight to be the sum of the weights of its components, *i.e.*, $w_{\mathrm{H}}(\mathbf{a}(x)) = \sum_{i=0}^{L-1} \mathrm{wt}(a_i(x))$. Throughout this work, computations implicitly shift to integer arithmetic upon taking the weight.

In using a ring, there are a few important points to bear in mind. The elements in a ring R need not have a multiplicative inverse; the ones that do are called *units*. The ring may include zero divisors, where a *zero divisor* (or *factor of zero*) is a nonzero element $a \in R$, such that $ab = 0$ for some $b \in R, b \neq 0$. For example, the elements $a = 2$ and $b = 3$ in $R = \mathbb{Z}/6\mathbb{Z}$, the ring of integers modulo 6, are zero divisors. Note that units cannot be zero divisors. Moreover, in finite rings, such as $\mathbb{Z}/6\mathbb{Z}$ or $\mathbb{F}_2[x]/\langle x^N - 1\rangle$, every nonzero element of the ring must be either a unit or a zero divisor [18, p. 205].

The elements of weight one in the polynomial quotient ring $\mathbb{F}_2[x]/\langle x^N - 1\rangle$ are the monomials, all of which are units in the ring. Specifically, the inverse of the monomial $a(x) = x^i, 0 \leq i < N$, is the monomial $(a(x))^{-1} = x^{(N-i) \bmod N}$. Under the isomorphism defined above, the monomials in the ring correspond to *cyclic permutation matrices*, which are binary circulant matrices with a single one in each row and each column.

For any $N > 0$, the nonzero elements in the ring $\mathbb{F}_2[x]/\langle x^N - 1\rangle$ that have even weight are zero divisors. For instance, the product of the polynomial $x^{N-1} + x^{N-2} + \cdots + x + 1$ and any even weight polynomial is zero. Odd weight polynomials, on the other hand, may sometimes be zero divisors, such as $x^3 + x + 1$ in the ring $\mathbb{F}_2[x]/\langle x^7 - 1\rangle$.

As we are interested in the connection between protographs and QC-LDPC codes, we focus on parity-check matrices $\mathbf{H}$ that are in $J \times L$ block matrix form, *i.e.*,

$$\mathbf{H} \triangleq \begin{bmatrix} \mathbf{H}_{0,0} & \cdots & \mathbf{H}_{0,L-1} \\ \vdots & \ddots & \vdots \\ \mathbf{H}_{J-1,0} & \cdots & \mathbf{H}_{J-1,L-1} \end{bmatrix}$$

where each submatrix $\mathbf{H}_{j,i}$ is an $N \times N$ binary circulant matrix. Let $h_{j,i,s} \in \mathbb{F}_2$ be the left-most entry in the sth row of the submatrix $\mathbf{H}_{j,i}$. We can then write $\mathbf{H}_{j,i} = \sum_{s=0}^{N-1} h_{j,i,s}\mathbf{I}_s$, where $\mathbf{I}_s$ is the $N \times N$ identity matrix circularly left-shifted by s positions. Now, using the same convention as above for identifying matrices with polynomial residues, we can associate with $\mathbf{H}$ the *polynomial parity-check matrix* $\mathbf{H}(x)$, where $\mathbf{H}(x) \in \left(\mathbb{F}_2[x]/\langle x^N - 1\rangle\right)^{J \times L}$,

$$\mathbf{H}(x) \triangleq \begin{bmatrix} h_{0,0}(x) & \cdots & h_{0,L-1}(x) \\ \vdots & \ddots & \vdots \\ h_{J-1,0}(x) & \cdots & h_{J-1,L-1}(x) \end{bmatrix},$$

and $h_{j,i}(x) \triangleq \sum_{s=0}^{N-1} h_{j,i,s}x^s$.

We will be interested in the weight of each polynomial entry of $\mathbf{H}(x)$, or, equivalently, the row or column sum of each submatrix of $\mathbf{H}$. The *weight matrix* of $\mathbf{H}(x)$, which is a $J \times L$ matrix of nonnegative integers, is defined as

$$\mathrm{wt}\,(\mathbf{H}(x)) \triangleq \begin{bmatrix} \mathrm{wt}\,(h_{0,0}(x)) & \cdots & \mathrm{wt}\,(h_{0,L-1}(x)) \\ \vdots & \ddots & \vdots \\ \mathrm{wt}\,(h_{J-1,0}(x)) & \cdots & \mathrm{wt}\,(h_{J-1,L-1}(x)) \end{bmatrix}.$$

Note that for a protograph-based QC-LDPC code, the weight matrix of the associated polynomial parity-check matrix $\mathrm{wt}\,(\mathbf{H}(x))$ is precisely the corresponding protomatrix, $\mathbf{A} = \mathrm{wt}\,(\mathbf{H}(x))$. It is also convenient to represent codewords of QC codes in polynomial form. In particular, the set of codewords, which is the set of vectors $\mathbf{c}$ such that $\mathbf{Hc}^T = \mathbf{0}^T$ over $\mathbb{F}_2$, maps to the set of polynomial vectors $\mathbf{c}(x) \in \left(\mathbb{F}_2[x]/\langle x^N - 1\rangle\right)^L$ satisfying $\mathbf{H}(x)\mathbf{c}(x)^T = \mathbf{0}^T$. Under this identification, the entries $c_i(x)$ of the polynomial vector $\mathbf{c}(x) = (c_0(x), c_1(x), \ldots, c_{L-1}(x))$ correspond to the length-N subblocks of the codeword $\mathbf{c}$ that were defined at the start of this section.

IV. Minimum Distance Bounds for QC Codes

In this section, we review the upper bounds on the minimum (Hamming) distance of QC-LDPC codes that were established in [9] and then extend them to punctured QC-LDPC codes.

A. Upper Bounds for QC Codes

We will use the shorthand notation $[L]$ to indicate the set of L consecutive integers, $\{0, 1, 2, \ldots, L-1\}$. We also let $\mathcal{S} \setminus i$ denote all the elements of $\mathcal{S}$, excluding the element i. We denote by $\mathbf{A}_{\mathcal{S}}$ the submatrix of $\mathbf{A}$ containing the columns indicated by the index set $\mathcal{S}$. Similarly, $\mathbf{a}_{\mathcal{S}}$ denotes the subvector containing the elements of the vector $\mathbf{a}$ indicated by the index set $\mathcal{S}$.

The permanent of a $J \times J$ matrix $\mathbf{B} = [b_{j,i}]$ over a ring is defined to be

$$\mathrm{perm}(\mathbf{B}) \triangleq \sum_{\sigma} \prod_{j \in [J]} b_{j,\sigma(j)}$$

where the summation is over all $J!$ permutations σ of the set $[J]$, and $\sigma(j)$ is the jth entry of the permuted set $\sigma([J])$. The definition of the permanent resembles that of the determinant of a square matrix;

$$\det(\mathbf{B}) \triangleq \sum_{\sigma} \mathrm{sign}(\sigma) \prod_{j \in [J]} b_{j,\sigma(j)}$$

where $\mathrm{sign}(\sigma)$ equals $+1$ if σ is an even permutation and -1 if σ is an odd permutation. (Recall that an even permutation is obtained by applying an even number of transpositions of pairs of elements to the sequence $\{0, 1, 2, \ldots, J-1\}$.) When the elements of $\mathbf{B}$ belong to a ring of characteristic two, where addition and subtraction are interchangeable, $\mathrm{perm}(\mathbf{B}) = \det(\mathbf{B})$. Like the determinant, the permanent may be computed recursively by taking the cofactor expansion along any row or column. That is, for any $j \in [J]$, the cofactor expansion of the permanent of matrix $\mathbf{B}$ along the jth row is

$$\mathrm{perm}(\mathbf{B}) = \sum_{i \in [J]} b_{j,i} \cdot \mathrm{perm}(\mathbf{B}'_{[J] \setminus i}) \tag{4}$$

with $\mathbf{B}'$ denoting the submatrix of $\mathbf{B}$ with the jth row removed. (Note that the subscript $[J] \setminus i$ on $\mathbf{B}'$ in (4) removes the ith column.)

In the derivation of upper bounds on the minimum distance, we will make use of the notion of dependence of vectors over a commutative ring R. This will require an appeal to some special properties of the ring of matrices over a finite commutative ring with unity. We remark that, in [9], an alternative approach that exploits the connection between QC block codes and convolutional codes was used in the derivation of the upper bounds on the minimum distance.

Let $\mathcal{U}$ be a set of vectors over R; that is, $\mathcal{U} = \{\mathbf{u}_0, \ldots, \mathbf{u}_{n-1}\}$, where $\mathbf{u}_i \in R^N$. The set $\mathcal{U}$ is said to be *dependent*[1] if there exist elements $r_0, \ldots, r_{n-1} \in R$, not all zero, such that the linear combination

$$r_0 \mathbf{u}_0 + \cdots + r_{n-1} \mathbf{u}_{n-1} = (0, \ldots, 0). \tag{5}$$

If no such set of elements exists, the set $\mathcal{U}$ is said to be *independent*. (See, for example, [19, p. 454].) We note that this independence test may be applied to the set of row vectors of a matrix with elements in R. A *dependent row* is any row in (5) which is multiplied by a nonzero scalar.

Lemma 1: Let $\mathbf{B}$ be a square matrix over a finite commutative ring with unity. Then, $\det(\mathbf{B})$ equals zero or is a zero divisor if and only if the set of row vectors of $\mathbf{B}$ is dependent.

Proof: The proof of Lemma 1 can be found in the Appendix, along with some illustrative examples. ■

We now review a technique from [9] for explicitly constructing codewords of a QC code specified by a polynomial parity-check matrix.

Lemma 2 (Lemma 6 [9]): Let $\mathcal{C}$ be a QC code with polynomial parity-check matrix $\mathbf{H}(x) \in \left(\mathbb{F}_2[x]/\langle x^N - 1\rangle\right)^{J \times L}$. Let $\mathcal{S}$ be an arbitrary size-$(J+1)$ subset of $[L]$ and let $\mathbf{c}(x) \in \left(\mathbb{F}_2[x]/\langle x^N - 1\rangle\right)^{L}$ be a length-L vector whose elements are given by

$$c_i(x) \triangleq \begin{cases} \mathrm{perm}\left(\mathbf{H}_{\mathcal{S}\setminus i}(x)\right), & \text{if } i \in \mathcal{S} \\ 0, & \text{otherwise.} \end{cases}$$

Then, $\mathbf{c}(x)$ is a codeword in $\mathcal{C}$.

Proof: For any $j \in [J]$, let the jth row of $\mathbf{H}(x)$ be $\mathbf{h}_j(x)$. Then,

$$\begin{aligned} \mathbf{h}_j(x)\mathbf{c}(x)^T &= \sum_{i \in [L]} h_{j,i}(x) \cdot c_i(x) \\ &= \sum_{i \in \mathcal{S}} h_{j,i}(x)\mathrm{perm}\left(\mathbf{H}_{\mathcal{S}\setminus i}(x)\right) && (6) \\ &= \mathrm{perm}\begin{bmatrix} \mathbf{h}_{j,\mathcal{S}}(x) \\ \mathbf{H}_{\mathcal{S}}(x) \end{bmatrix} && (7) \\ &= \det\begin{bmatrix} \mathbf{h}_{j,\mathcal{S}}(x) \\ \mathbf{H}_{\mathcal{S}}(x) \end{bmatrix} = 0 && (8) \end{aligned}$$

where computations are in the ring $\mathbb{F}_2[x]/\langle x^N - 1\rangle$. The cofactor expansion of (7) is (6). As the elements belong to a ring of characteristic two, the permanent in (7) equals the determinant in (8). The determinant shown must be zero as it contains a repeated row. Since every row of $\mathbf{H}(x)$ has zero inner product with $\mathbf{c}(x)$, we conclude that $\mathbf{H}(x)\mathbf{c}(x)^T = \mathbf{0}^T$. Therefore, $\mathbf{c}(x)$ is a codeword in $\mathcal{C}$. ■

While the $\min$ function applied to a collection of nonnegative real numbers returns the minimum, we will require a variant of this function, denoted $\min^*$, defined as follows. For a finite collection of nonnegative real numbers $\mathcal{R}$, let $\mathcal{R}^+ \subseteq \mathcal{R}$ be the subset of positive elements of $\mathcal{R}$. We define

$$\min{}^* \mathcal{R} \triangleq \begin{cases} \min \mathcal{R}^+, & \text{if } \mathcal{R}^+ \neq \emptyset \\ +\infty, & \text{if } \mathcal{R}^+ = \emptyset. \end{cases}$$

The minimum distance of the QC code $\mathcal{C}$ can then be written as

$$d_{\min}(\mathcal{C}) = \min_{\mathbf{c}(x) \in \mathcal{C}}{}^* w_{\mathrm{H}}\left(\mathbf{c}(x)\right) \tag{9}$$

where we have used $\min^*$ to exclude the all-zero codeword.

We now develop two upper bounds on the minimum distance. The first, based upon [9], uses Lemma 2 to produce low-weight codewords from the polynomial parity-check matrix of the code. We will generate as many codewords as possible and apply (9) to achieve an upper bound on the minimum distance.

Theorem 3 (Theorem 7 [9]): Let $\mathcal{C}$ be a QC code with the polynomial parity-check matrix $\mathbf{H}(x) \in \left(\mathbb{F}_2[x]/\langle x^N - 1\rangle\right)^{J \times L}$. Then, the minimum distance of $\mathcal{C}$ satisfies the upper bound

$$d_{\min}(\mathcal{C}) \le \min_{\substack{\mathcal{S} \subseteq [L] \\ |\mathcal{S}| = J+1}}{}^* \sum_{i \in \mathcal{S}} \mathrm{wt}\left(\mathrm{perm}\left(\mathbf{H}_{\mathcal{S}\setminus i}(x)\right)\right). \tag{10}$$

Proof: Let $\mathcal{S}$ be a subset of $[L]$ of size-$(J+1)$ and apply Lemma 2 to construct a codeword $\mathbf{c}(x)$ in $\mathcal{C}$. The weight of $\mathbf{c}(x)$ is

$$\begin{aligned} w_{\mathrm{H}}\left(\mathbf{c}(x)\right) &= \sum_{i \in [L]} \mathrm{wt}\left(c_i(x)\right) \\ &= \sum_{i \in \mathcal{S}} \mathrm{wt}\left(\mathrm{perm}\left(\mathbf{H}_{\mathcal{S}\setminus i}(x)\right)\right). && (11) \end{aligned}$$

The upper bound (10) follows immediately by combining (9) and (11) and noting that, in general, only a strict subset of the codewords can be generated by Lemma 2. We must use the $\min^*$ function because, for some choices of the set $\mathcal{S}$, the construction in Lemma 2 will yield the all-zero codeword, and we have to exclude those sets from the calculation of the upper bound. ■

The second upper bound on the minimum distance makes use of an upper bound on the weight of the permanent of a matrix over the ring $\mathbb{F}_2[x]/\langle x^N - 1\rangle$, as described in the following lemma.

Lemma 4: Let $\mathbf{B}(x)$ be a $J \times J$ matrix with elements $b_{j,i}(x)$ in the ring $\mathbb{F}_2[x]/\langle x^N - 1\rangle$. Then, the weight of the permanent of $\mathbf{B}$ satisfies the upper bound

$$\mathrm{wt}\left(\mathrm{perm}(\mathbf{B}(x))\right) \le \mathrm{perm}\left(\mathrm{wt}(\mathbf{B}(x))\right).$$

[1] Some authors use the term "linearly dependent" in this context.

Proof: Let the polynomials $a(x)$ and $b(x)$ be in the ring $\mathbb{F}_2[x]/\langle x^N - 1\rangle$. We know that $\mathrm{wt}[a(x)+b(x)] \leq \mathrm{wt}(a(x)) + \mathrm{wt}(b(x))$, as the maximum number of nonzero coefficients in the sum $a(x)+b(x)$ is $\mathrm{wt}(a(x)) + \mathrm{wt}(b(x))$.

Similarly, we know that $\mathrm{wt}[a(x)\cdot b(x)] \leq \mathrm{wt}(a(x))\cdot \mathrm{wt}(b(x))$, as the maximum number of nonzero coefficients in the product $a(x)\cdot b(x)$ is $\mathrm{wt}(a(x))\cdot\mathrm{wt}(b(x))$. Therefore,

$$\begin{aligned}\mathrm{wt}\left(\mathrm{perm}(\mathbf{B}(x))\right) &= \mathrm{wt}\left[\sum_{\sigma}\prod_{j\in[J]} b_{j,\sigma(j)}(x)\right]\\ &\leq \sum_{\sigma}\mathrm{wt}\left[\prod_{j\in[J]} b_{j,\sigma(j)}(x)\right]\\ &\leq \sum_{\sigma}\prod_{j\in[J]}\mathrm{wt}\left(b_{j,\sigma(j)}(x)\right)\\ &= \mathrm{perm}\left(\mathrm{wt}(\mathbf{B}(x))\right).\end{aligned}$$

■

The second upper bound on the minimum distance, described in the next theorem, is expressed in terms of the weight matrix of the code.

Theorem 5 (Theorem 8 [9]): Let $\mathcal{C}$ be a QC code with polynomial parity-check matrix $\mathbf{H}(x) \in \left(\mathbb{F}_2[x]/\langle x^N-1\rangle\right)^{J\times L}$ and let $\mathbf{A} \triangleq \mathrm{wt}(\mathbf{H}(x))$. Then, the minimum distance of $\mathcal{C}$ satisfies the upper bound

$$d_{\min}(\mathcal{C}) \leq \min_{\substack{\mathcal{S}\subseteq[L]\\ |\mathcal{S}|=J+1}}{}^{*} \sum_{i\in\mathcal{S}} \mathrm{perm}\left(\mathbf{A}_{\mathcal{S}\setminus i}\right) \quad (\text{in } \mathbb{Z}). \tag{12}$$

Proof: The proof follows from Theorem 3 and Lemma 4. The only subtlety arises from the fact that the sets $\mathcal{S}$ excluded from (10) by the $\min^*$ function may not be excluded from (12) by the application of $\min^*$, *i.e.*, there might be sets $\mathcal{S}$ such that $\sum_{i\in\mathcal{S}}\mathrm{wt}\left(\mathrm{perm}\left(\mathbf{H}_{\mathcal{S}\setminus i}(x)\right)\right) = 0$ but $\sum_{i\in\mathcal{S}}\mathrm{perm}\left(\mathbf{A}_{\mathcal{S}\setminus i}\right) > 0$. The resolution of this potential complication can be achieved by reference to Theorem 8 in [9]. ■

Remark 1: Alternatively, we may prove the correctness of this upper bound by following the arguments presented in the proof of Theorem 9, below. Since Theorem 9 includes puncturing, consider the set $\mathcal{P}$ to be empty for this case.

B. Upper Bounds for Punctured QC Codes

We now extend the preceding upper bounds on the minimum distance to the class of punctured QC codes. The puncturing strategy of the AR4JA codes is prompted by the addition of the precoder to the underlying protographs, a modification that generally improves the decoding threshold [10], [11]. Note that puncturing whole subblocks of the polynomial codeword $\mathbf{c}(x)$ preserves quasi-cyclicity. The set $\mathcal{P} \subset [L]$ indexes the subblocks of the polynomial codeword $\mathbf{c}(x)$ which are not transmitted. Indices of $\mathcal{P}$ may also be associated with columns of the $J\times L$ polynomial parity-check matrix $\mathbf{H}(x)$.

We begin with a QC code $\mathcal{C}$ based upon $\mathbf{H}(x)$. Next, we define a new QC code $\mathcal{C}'$ by puncturing the components of $\mathbf{c}(x)$ that are indexed by $\mathcal{P}$. We mark the subblocks to be punctured with the symbol "φ" as reindexing would introduce unnecessary notational complexity, and we define $\mathrm{wt}(\varphi) = 0$, since the punctured symbols are not transmitted.

Lemma 6: Let $\mathcal{C}'$ be a punctured QC code constructed by puncturing subblocks of the QC code $\mathcal{C}$, defined by the polynomial parity-check matrix $\mathbf{H}(x) \in \left(\mathbb{F}_2[x]/\langle x^N-1\rangle\right)^{J\times L}$. Let the subblocks of $\mathcal{C}$ indexed by the set $\mathcal{P}, \mathcal{P}\subset[L]$, be punctured. Let $\mathcal{S}$ be an arbitrary size-$(J+1)$ subset of $[L]$. Let the length-L vector $\mathbf{c}'(x) = \left(c'_0(x), c'_1(x), \ldots, c'_{L-1}(x)\right)$, with $c'(x) \in \mathbb{F}_2[x]/\langle x^N-1\rangle \cup \{\varphi\}$, be defined by

$$c'_i(x) \triangleq \begin{cases}\mathrm{perm}\left(\mathbf{H}_{\mathcal{S}\setminus i}(x)\right), & \text{if } i\in\mathcal{S}\setminus\mathcal{P}\\ \varphi, & \text{if } i\in\mathcal{P}\\ 0, & \text{otherwise.}\end{cases}$$

Then, $\mathbf{c}'(x)$ is a codeword of the punctured code $\mathcal{C}'$.

Proof: This follows by noting that $\mathbf{c}'(x)$ is obtained by puncturing the subblocks indexed by $\mathcal{P}$ from the codeword $\mathbf{c}(x)$ of Lemma 2. ■

Theorem 7: Let $\mathcal{C}'$ be a punctured QC code constructed by puncturing subblocks of the QC code $\mathcal{C}$ with polynomial parity-check matrix $\mathbf{H}(x) \in \left(\mathbb{F}_2[x]/\langle x^N-1\rangle\right)^{J\times L}$. Let the subblocks of $\mathcal{C}$ indexed by the set $\mathcal{P}, \mathcal{P}\subset[L]$, be punctured. Then

$$d_{\min}(\mathcal{C}') \leq \min_{\substack{\mathcal{S}\subseteq[L]\\ |\mathcal{S}|=J+1}}{}^{*} \sum_{i\in\mathcal{S}\setminus\mathcal{P}} \mathrm{wt}\left(\mathrm{perm}\left(\mathbf{H}_{\mathcal{S}\setminus i}(x)\right)\right). \tag{13}$$

Proof: Let $\mathcal{S}$ be a subset of $[L]$ of size-$(J+1)$, and apply Lemma 6 to construct a codeword $\mathbf{c}'(x)$ in $\mathcal{C}'$. The weight of this codeword is

$$\begin{aligned}w_{\mathrm{H}}\left(\mathbf{c}'(x)\right) &= \sum_{i\in[L]}\mathrm{wt}\left(c'_i(x)\right)\\ &= \sum_{i\in\mathcal{S}\setminus\mathcal{P}}\mathrm{wt}\left(\mathrm{perm}\left(\mathbf{H}_{\mathcal{S}\setminus i}(x)\right)\right)\end{aligned} \tag{14}$$

where we use the fact that $\mathrm{wt}(\varphi) = 0$. Combining (14) with (9), we obtain the upper bound (13), again noting that we obtain an upper bound because in general only a strict subset of the codewords can be generated by Lemma 6. ■

Care must be taken to ensure that the puncturing operation does not reduce the dimensionality of the code, *i.e.*, the base-2 logarithm of the number of distinct codewords. This is a requirement in the results that follow. Clearly, if puncturing a nonzero codeword of $\mathcal{C}$ produces the all-zero codeword of $\mathcal{C}'$, dimensionality will be lost with respect to the original code.

Lemma 8: Let $\mathcal{C}'$ be a punctured QC code constructed by puncturing subblocks of the QC code $\mathcal{C}$, while maintaining the dimensionality of $\mathcal{C}$. Let the length-L vector $\mathbf{c}(x)$ be a codeword of $\mathcal{C}$ and $\mathbf{c}'(x)$ be a codeword of $\mathcal{C}'$ obtained by puncturing $\mathbf{c}(x)$. Then, $c'_i(x) \in \{0, \varphi\}\ \forall\, i\in[L]$ if and only if $\mathbf{c}(x) = \mathbf{0}$.

Proof: The necessity of the condition $\mathbf{c}(x) = \mathbf{0}$ follows from the requirement that the dimensionality of the original code be maintained. The sufficiency of the condition follows directly from the fact that puncturing the all-zero codeword of $\mathcal{C}$ produces the all-zero codeword of $\mathcal{C}'$. ■

Since the contribution of each subblock to the weight of the codeword is nonnegative, the weight of any particular punctured codeword must be less than or equal to its weight before puncturing, as can be seen by comparison of (14) to (11). Moreover, Lemma 8 implies that the upper bound of Theorem 7 will always be less than or equal to the upper bound of Theorem 3, where no puncturing is used.

Theorem 9: Let $\mathcal{C}'$ be a punctured QC code constructed by puncturing subblocks of the QC code $\mathcal{C}$, defined by the polynomial parity-check matrix $\mathbf{H}(x) \in \left(\mathbb{F}_2[x]/\langle x^N - 1\rangle\right)^{J \times L}$ and let $\mathbf{A} \triangleq \mathrm{wt}\left(\mathbf{H}(x)\right)$. Let the subblocks of $\mathcal{C}$ indexed by the set $\mathcal{P}$, $\mathcal{P} \subset [L]$, be punctured, while maintaining the dimensionality of the code. Then

$$d_{\min}(\mathcal{C}') \leq \min_{\substack{\mathcal{S} \subseteq [L] \\ |\mathcal{S}| = J+1}}{}^{*} \sum_{i \in \mathcal{S} \setminus \mathcal{P}} \mathrm{perm}\left(\mathbf{A}_{\mathcal{S} \setminus i}\right) \qquad (\text{in } \mathbb{Z}). \qquad (15)$$

Proof: The proof techniques we use are similar to those used in the proof of Theorem 8 in [9], while avoiding the use of intermediate bounds based on convolutional codes. Let $\mathcal{S}$ be a subset of $[L]$ of size-$(J+1)$, and apply Lemma 6 to construct a codeword, $\mathbf{c}'(x)$, in code $\mathcal{C}'$. From (14), we obtain

$$\begin{aligned} w_{\mathrm{H}}\left(\mathbf{c}'(x)\right) &= \sum_{i \in \mathcal{S} \setminus \mathcal{P}} \mathrm{wt}\left(\mathrm{perm}\left(\mathbf{H}_{\mathcal{S} \setminus i}(x)\right)\right) \\ &\leq \sum_{i \in \mathcal{S} \setminus \mathcal{P}} \mathrm{perm}\left(\mathrm{wt}\left(\mathbf{H}_{\mathcal{S} \setminus i}(x)\right)\right) \\ &= \sum_{i \in \mathcal{S} \setminus \mathcal{P}} \mathrm{perm}\left(\mathbf{A}_{\mathcal{S} \setminus i}\right) \end{aligned}$$

where we invoked Lemma 4 to obtain the inequality in the second step.

We now show the validity of using the $\min^*$ function in the upper bound. The potential complication arises from the fact that for specific choices of the set $\mathcal{S}$, namely those which yield the all-zero codeword in Lemma 6, the $\min^*$ function may not exclude their contribution to (15), even though it does so in the bound (13). So, assume that a specific choice of $\mathcal{S}$ yields $w_{\mathrm{H}}(\mathbf{c}'(x)) = 0$ according to (14), but produces a nonzero value for $\sum_{i \in \mathcal{S} \setminus \mathcal{P}} \mathrm{perm}\left(\mathbf{A}_{\mathcal{S} \setminus i}\right)$. We must show that there exists a nonzero codeword $\mathbf{c}^{*'}(x)$ for which $w_{\mathrm{H}}(\mathbf{c}^{*'}(x)) \leq \sum_{i \in \mathcal{S} \setminus \mathcal{P}} \mathrm{perm}\left(\mathbf{A}_{\mathcal{S} \setminus i}\right)$. Thus, for such specific choices of $\mathcal{S} \subseteq [L]$, we assume in the remainder of the proof that $c'_i(x) \in \{0, \varphi\}$ $\forall i \in [L]$. By Lemma 8, we know that $\mathbf{c}(x) = \mathbf{0}$ for this $\mathcal{S}$. By Lemma 2, we know that every $J \times J$ submatrix of $\mathbf{H}_{\mathcal{S}}(x)$ must have a zero permanent and determinant. We now consider two cases.

Case 1: If $\sum_{i \in \mathcal{S} \setminus \mathcal{P}} \mathrm{perm}\left(\mathbf{A}_{\mathcal{S} \setminus i}\right) = 0$, then this specific $\mathcal{S}$ has no effect on the bound (15), as the zero result will be discarded by the $\min^*$ function.

Case 2: Alternatively, if $\sum_{i \in \mathcal{S} \setminus \mathcal{P}} \mathrm{perm}\left(\mathbf{A}_{\mathcal{S} \setminus i}\right) > 0$, there are no all-zero rows in $\mathbf{A}_{\mathcal{S}}$ and, therefore, none in $\mathbf{H}_{\mathcal{S}}(x)$. However, we know that every $J \times J$ submatrix of $\mathbf{H}_{\mathcal{S}}(x)$ has a zero determinant for the specific $\mathcal{S}$ that generates $\mathbf{c}(x) = \mathbf{0}$. By Lemma 1, this implies that the set of rows of $\mathbf{H}_{\mathcal{S}}(x)$ is dependent. We analyze this case further by setting aside the tth row of $\mathbf{H}_{\mathcal{S}}(x), \mathbf{h}_{t,\mathcal{S}}(x)$, preferring a dependent row to be the tth row.[2] We form a new matrix, $\mathbf{H}'(x)$, with the remaining $J-1$ rows of $\mathbf{H}(x)$ and a matrix, $\mathbf{A}'$, with the corresponding $J-1$ rows of $\mathbf{A}$. Because of the assumption that $\sum_{i \in \mathcal{S} \setminus \mathcal{P}} \mathrm{perm}\left(\mathbf{A}_{\mathcal{S} \setminus i}\right) > 0$, there must be at least one index $i \in \mathcal{S} \setminus \mathcal{P}$, such that $\mathrm{perm}\left(\mathbf{A}_{\mathcal{S} \setminus i}\right) > 0$. The cofactor expansion of this term along row t contains a term

$$a_{t,i^*} \cdot \mathrm{perm}\left(\mathbf{A}'_{(\mathcal{S} \setminus i) \setminus i^*}\right) > 0 \qquad (16)$$

for some $i^* \in \mathcal{S} \setminus i$, where the positive integer a_{t,i^*} is the entry in the tth row and i^*th column of $\mathbf{A}$. Let $\mathcal{S}^* \triangleq \mathcal{S} \setminus i^*$.

Proceeding, we now assume that $\mathbf{H}'_{\mathcal{S}^*}(x)$ contains at least one $(J-1) \times (J-1)$ submatrix with nonzero permanent. (If this is not true, we repeat the row removal process above, which may need to be repeated several times. In the extreme case, these reductions could continue until we get a 1×2 matrix $\mathbf{H}'_{\mathcal{S}^*}(x)$, having at least one nonzero entry.) Then, applying Lemma 2, with $\mathbf{H}(x) = \mathbf{H}'(x)$ and $\mathcal{S} = \mathcal{S}^*$, we generate a nonzero vector, $\mathbf{c}^*(x)$, with components

$$c_i^*(x) = \begin{cases} \mathrm{perm}\left(\mathbf{H}'_{\mathcal{S}^* \setminus i}(x)\right), & \text{if } i \in \mathcal{S}^* \\ 0, & \text{otherwise.} \end{cases}$$

The proof of Lemma 2 implies that $\mathbf{H}'(x)\mathbf{c}^*(x)^T = \mathbf{0}^T$. Multiplying the removed row of the parity-check matrix by the vector $\mathbf{c}^*(x)$ yields

$$\begin{aligned} \mathbf{h}_t(x)\mathbf{c}^*(x)^T &= \sum_{i \in [L]} h_{t,i}(x) \cdot c_i^*(x) \\ &= \sum_{i \in \mathcal{S}^*} h_{t,i}(x) \mathrm{perm}\left(\mathbf{H}'_{\mathcal{S}^* \setminus i}(x)\right) \\ &= \mathrm{perm}\left(\mathbf{H}_{\mathcal{S}^*}(x)\right) = 0 \end{aligned}$$

since all $J \times J$ submatrices of $\mathbf{H}_{\mathcal{S}}(x)$ were assumed to have a zero permanent. Therefore, the nonzero vector $\mathbf{c}^*(x)$ is a codeword in $\mathcal{C}$.

By puncturing $\mathbf{c}^*(x)$, we generate another nonzero vector, $\mathbf{c}^{*'}(x)$, which is a codeword in $\mathcal{C}'$. The Hamming weight of this codeword satisfies the upper bound

$$\begin{aligned} w_{\mathrm{H}}(\mathbf{c}^{*'}(x)) &= \sum_{i \in \mathcal{S}^* \setminus \mathcal{P}} \mathrm{wt}\left(\mathrm{perm}\left(\mathbf{H}'_{\mathcal{S}^* \setminus i}(x)\right)\right) \\ &\leq \sum_{i \in \mathcal{S}^* \setminus P} \mathrm{perm}\left(\mathbf{A}'_{\mathcal{S}^* \setminus i}\right) && (17) \\ &\leq a_{t,i^*} \cdot \sum_{i \in \mathcal{S}^* \setminus P} \mathrm{perm}\left(\mathbf{A}'_{\mathcal{S}^* \setminus i}\right) && (18) \\ &\leq \sum_{i \in \mathcal{S}^* \setminus \mathcal{P}} \sum_{j \in \mathcal{S} \setminus i} a_{t,j} \cdot \mathrm{perm}\left(\mathbf{A}'_{(\mathcal{S} \setminus i) \setminus j}\right) && (19) \\ &= \sum_{i \in \mathcal{S}^* \setminus \mathcal{P}} \mathrm{perm}\left(\mathbf{A}_{\mathcal{S} \setminus i}\right) && (20) \\ &\leq \sum_{i \in \mathcal{S} \setminus \mathcal{P}} \mathrm{perm}\left(\mathbf{A}_{\mathcal{S} \setminus i}\right). \end{aligned}$$

Applying Lemma 4 produces (17). Using the fact that $a_{t,i^*} \geq 1$, a consequence of (16), we upper bound (17) by (18). Next, we further upper bound (18) by (19) by adding additional nonnegative terms to (19). We recognize that (19) contains the sum of the cofactor expansions of each addend of (20).

[2] The proof holds no matter which row is chosen for removal; however, the row removal process terminates more rapidly if a dependent row is chosen.

We can now conclude that, even if a set $\mathcal{S}$ generates the all-zero codeword in Lemma 6, it still yields a valid upper bound on the minimum distance of the punctured code, *i.e.,*

$$d_{\min}(\mathcal{C}') \leq \sum_{i \in \mathcal{S}\setminus\mathcal{P}} \operatorname{perm}\left(\mathbf{A}_{\mathcal{S}\setminus i}\right)$$

provided that $\operatorname{perm}(\mathbf{A}_{\mathcal{S}\setminus i})$ is positive for at least one $i \in \mathcal{S}\setminus\mathcal{P}$. The validity of the upper bound in (15) follows. ■

V. Tighter Bounds on the Minimum Distance

Examining the AR4JA protomatrices for rate-2/3 in (2) and rate-4/5 in (3), we see cases where the selection of $J+1=4$ columns of the weight matrix $\mathbf{A}$ will produce a submatrix $\mathbf{A}_{\mathcal{S}}$ containing an all-zero top row. This particular selection of $\mathcal{S}$ produces the all-zero codeword by the codeword construction of Lemmas 2 and 6, and, thus, will have no effect on the upper bounds of Theorems 3, 5, 7, and 9. We can improve those bounds by finding nonzero codewords after row elimination, as in the proof of Theorem 9.

In the interest of brevity, we will state the following theorems in a way that applies to both unpunctured and punctured codes. In the unpunctured case, it is understood that the set $\mathcal{P}$ is empty.

Lemma 10: Let $\mathcal{C}'$ be a QC code constructed by optionally puncturing subblocks of the QC code $\mathcal{C}$, defined by the polynomial parity-check matrix $\mathbf{H}(x) \in \left(\mathbb{F}_2[x]/\langle x^N - 1\rangle\right)^{J\times L}$. Let the subblocks of $\mathcal{C}$ indexed by the set $\mathcal{P}, \mathcal{P} \subset [L]$, be punctured. Let $\mathbf{H}'(x)$ be a submatrix of $\mathbf{H}(x)$ with rows $\mathbf{h}_t(x), t \in \mathcal{T} \subset [J]$, removed. Let $\mathcal{S}$ be a subset of $[L]$ of size $J+1-|\mathcal{T}|$, such that

$$\operatorname{perm}\begin{bmatrix}\mathbf{h}_{t,\mathcal{S}}(x)\\ \mathbf{H}'_{\mathcal{S}}(x)\end{bmatrix} = 0 \quad \forall t \in \mathcal{T}. \tag{21}$$

Let the components of the length-L vector $\mathbf{c}'(x) = \left(c'_0(x), c'_1(x), \ldots, c'_{L-1}(x)\right)$, with $c'_i(x) \in \mathbb{F}_2[x]/\langle x^N - 1\rangle \cup \{\varphi\}$, be defined as

$$c'_i(x) = \begin{cases} \operatorname{perm}\left(\mathbf{H}'_{\mathcal{S}\setminus i}(x)\right), & \text{if } i \in \mathcal{S}\setminus\mathcal{P}\\ \varphi, & \text{if } i \in \mathcal{P}\\ 0, & \text{otherwise.}\end{cases}$$

Then, $\mathbf{c}'(x)$ is a codeword in $\mathcal{C}'$.

Proof: We consider two cases.

Case 1: If $\mathcal{P} = \emptyset$ (the code is unpunctured), then $\mathcal{C} = \mathcal{C}'$. We first examine every retained row of $\mathbf{H}(x)$, denoted by $\mathbf{h}_j(x)$, where $j \in [J]$ and $j \notin \mathcal{T}$. The inner product of $\mathbf{h}_j(x)$ with the vector $\mathbf{c}'(x)$ is

$$\begin{aligned}\mathbf{h}_j(x)\mathbf{c}'(x)^T &= \sum_{i\in\mathcal{S}} h_{j,i}(x)\operatorname{perm}\left(\mathbf{H}'_{\mathcal{S}\setminus i}(x)\right)\\ &= \operatorname{perm}\begin{bmatrix}\mathbf{h}_{j,\mathcal{S}}(x)\\ \mathbf{H}'_{\mathcal{S}}(x)\end{bmatrix}\\ &= \det\begin{bmatrix}\mathbf{h}_{j,\mathcal{S}}(x)\\ \mathbf{H}'_{\mathcal{S}}(x)\end{bmatrix} = 0\end{aligned}$$

since the determinant expression contains a repeated row. Next, for every row $\mathbf{h}_t(x)$ removed from $\mathbf{H}(x)$, *i.e.*, every row $\mathbf{h}_t(x)$ in which $t \in \mathcal{T}$, we have

$$\mathbf{h}_t(x)\mathbf{c}'(x)^T = \operatorname{perm}\begin{bmatrix}\mathbf{h}_{t,\mathcal{S}}(x)\\ \mathbf{H}'_{\mathcal{S}}(x)\end{bmatrix} = 0$$

because the permanent was assumed to be zero in (21). Since all rows of the original polynomial parity-check matrix $\mathbf{H}(x)$ have been accounted for, $\mathbf{H}(x)\mathbf{c}'(x)^T = \mathbf{0}^T$ and $\mathbf{c}'(x)$ is a codeword in $\mathcal{C} = \mathcal{C}'$.

Case 2: If the code $\mathcal{C}'$ is punctured, let the components of the length-L vector $\mathbf{c}(x) = (c_0(x), c_1(x), \ldots, c_{L-1}(x))$, with $c_i(x) \in \mathbb{F}_2[x]/\langle x^N - 1\rangle$, be defined as

$$c_i(x) = \begin{cases}\operatorname{perm}\left(\mathbf{H}'_{\mathcal{S}\setminus i}(x)\right), & \text{if } i \in \mathcal{S}\\ 0, & \text{otherwise.}\end{cases}$$

The proof follows by noting that $\mathbf{c}'(x)$ is obtained by puncturing subblocks indexed by $\mathcal{P}$ from the unpunctured codeword $\mathbf{c}(x)$, above. Since Case 1 establishes that $\mathbf{c}(x) \in \mathcal{C}$, we conclude that $\mathbf{c}'(x) \in \mathcal{C}'$. ■

Not only does Lemma 10 remove all-zero rows from $\mathbf{H}_{\mathcal{S}}(x)$, it also helps produce lower weight codewords in more general conditions, as the following example shows.

Example 1: Consider the polynomial parity-check matrix

$$\mathbf{H}(x) = \begin{bmatrix} 0 & 0 & 0 & f_1(x)\\ x^a & x^b & x^c & f_2(x)\\ x^a & x^b & x^d & f_3(x)\end{bmatrix}$$

where $f_i(x), i = 1, 2, 3$, are arbitrarily chosen polynomials. Since $\operatorname{perm}(\mathbf{H}_{\mathcal{S}}(x)) = 0$ with the column set $\mathcal{S} = \{0, 1, 2\}$, as required by (21), we proceed with single row removal on $\mathbf{H}(x)$. Application of Lemma 10 for all possible choices of $\mathcal{T}$ with $|\mathcal{T}| = 1$ yields the codewords $\mathbf{c}(x) = \mathbf{0}$ and

$$\mathbf{c}(x) = \left(x^{b+d} + x^{b+c}, x^{a+d} + x^{a+c}, 0, 0\right) \bmod x^N - 1.$$

However, with careful consideration, we see that Lemma 10 will let us delete two subrows when the column set is $\mathcal{S} = \{0, 1\}$. This produces the obvious codeword $\mathbf{c}(x) = \left(x^b, x^a, 0, 0\right)$, when $\mathcal{T} = \{0, 1\}$ or $\mathcal{T} = \{0, 2\}$.

Theorem 11: Let $\mathcal{C}'$ be a QC code constructed by optionally puncturing subblocks of the QC code $\mathcal{C}$, defined by the polynomial parity-check matrix $\mathbf{H}(x) \in \left(\mathbb{F}_2[x]/\langle x^N - 1\rangle\right)^{J\times L}$. Let the subblocks of $\mathcal{C}$ indexed by the set $\mathcal{P}, \mathcal{P} \subset [L]$, be punctured. Let $\mathbf{H}'(x)$ be a submatrix of $\mathbf{H}(x)$ with rows $\mathbf{h}_t(x), t \in \mathcal{T} \subset [J]$, removed. Let $\mathcal{S}$ be a subset of $[L]$ of size $J+1-|\mathcal{T}|$, such that (21) holds. Then

$$d_{\min}(\mathcal{C}') \leq \min_{\mathcal{S},\mathcal{T}}{}^{*} \sum_{i\in\mathcal{S}\setminus\mathcal{P}} \operatorname{wt}\left(\operatorname{perm}\left(\mathbf{H}'_{\mathcal{S}\setminus i}(x)\right)\right). \tag{22}$$

Proof: The proof mirrors the proof of Theorem 7, with the weight of the resulting codeword now given by

$$w_{\mathrm{H}}\left(\mathbf{c}'(x)\right) = \sum_{i\in\mathcal{S}\setminus\mathcal{P}} \operatorname{wt}\left(\operatorname{perm}\left(\mathbf{H}'_{\mathcal{S}\setminus i}(x)\right)\right).$$

■

Note that the minimization in (22) requires the removal of every possible set of rows $\mathcal{T}$ and every set of retained columns $\mathcal{S}$ for which $|\mathcal{S}| + |\mathcal{T}| = J + 1$ and (21) holds. In the case of single row removal ($|\mathcal{T}| = 1$), any row may be removed, as our requirement (21) degenerates to the condition $\mathrm{perm}\,(\mathbf{H}_{\mathcal{S}}(x)) = 0$, which is independent of the row selected for removal. For multiple row removal, the conditions are more complex to evaluate as each row in the set to be removed must be tested individually to verify that (21) holds.

For a specified expansion factor N, Lemma 10 and Theorem 11 impose certain conditions on the set $\mathcal{T}$ that allow for the removal of rows from the polynomial parity-check matrix. However, these conditions cannot be directly translated into a form applicable to the nonnegative weight matrix $\mathbf{A}$, which is independent of N. Therefore, the following theorem uses the stricter condition that the subrow $\mathbf{a}_{t,\mathcal{S}}$ is all-zero before removal.

Theorem 12: Let $\mathcal{C}'$ be a QC code constructed by optionally puncturing subblocks of the QC code $\mathcal{C}$, defined by the polynomial parity-check matrix $\mathbf{H}(x) \in \left(\mathbb{F}_2[x]/\langle x^N - 1\rangle\right)^{J\times L}$ and let $\mathbf{A} \triangleq \mathrm{wt}\,(\mathbf{H}(x))$. Let the subblocks of $\mathcal{C}$ indexed by the set $\mathcal{P}, \mathcal{P} \subset [L]$, be punctured, while maintaining the dimensionality of the code. Let $\mathbf{A}'$ be a submatrix of $\mathbf{A}$ with rows $\mathbf{a}_t, t \in \mathcal{T} \subset [J]$, removed. Let $\mathcal{S}$ be a subset of $[L]$ of size $J + 1 - |\mathcal{T}|$, such that the subrows $\mathbf{a}_{t,\mathcal{S}} = \mathbf{0}\ \forall\, t \in \mathcal{T}$. Then

$$d_{\min}(\mathcal{C}') \leq \min_{\mathcal{S},\mathcal{T}}{}^{*} \sum_{i\in\mathcal{S}\setminus\mathcal{P}} \mathrm{perm}\left(\mathbf{A}'_{\mathcal{S}\setminus i}\right) \qquad (\text{in } \mathbb{Z}).$$

Proof: Let $\mathbf{H}'(x)$ be the submatrix of $\mathbf{H}(x)$ with rows $\mathbf{h}_t(x), t \in \mathcal{T}$, removed. The $|\mathcal{T}|$ subrows of the weight matrix $\mathbf{A}$ to be removed are all-zero (*i.e.*, $\mathbf{a}_{t,\mathcal{S}} = \mathbf{0}\ \forall\, t \in \mathcal{T}$) if and only if the corresponding subrows of the polynomial parity-check matrix $\mathbf{H}(x)$ are all-zero (*i.e.*, $\mathbf{h}_{t,\mathcal{S}}(x) = \mathbf{0}\ \forall\, t \in \mathcal{T}$). The latter condition implies that (21) holds and we may apply Lemma 10 with this $\mathcal{S}$ and $\mathcal{T}$ and construct a codeword $\mathbf{c}'(x)$ in the code $\mathcal{C}'$. By Theorem 11, the weight of $\mathbf{c}'(x)$ is

$$\begin{aligned} w_{\mathrm{H}}\left(\mathbf{c}'(x)\right) &= \sum_{i\in\mathcal{S}\setminus\mathcal{P}} \mathrm{wt}\left(\mathrm{perm}\left(\mathbf{H}'_{\mathcal{S}\setminus i}(x)\right)\right) \\ &\leq \sum_{i\in\mathcal{S}\setminus\mathcal{P}} \mathrm{perm}\left(\mathbf{A}'_{\mathcal{S}\setminus i}\right) \end{aligned}$$

where Lemma 4 is applied to obtain the inequality. Once again, the use of the $\min^{*}$ function in the bound must be validated by consideration of the all-zero codewords discarded in Theorem 11. The reasoning largely parallels that used in the proof of Theorem 9, but with $|\mathcal{T}|$ subrows of $\mathbf{H}_{\mathcal{S}}(x)$ guaranteed to be all-zero. We omit the details. ■

Example 2: The benefits of Theorem 12 may be seen by considering the weight matrix given by

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 2 \\ 1 & 2 & 2 & 1 \end{bmatrix}.$$

Treating the code as unpunctured, Theorem 9 produces a minimum distance upper bound of $+\infty$, since all 3×3 submatrices of $\mathbf{A}$ have a zero permanent. Theorem 12 produces a much tighter bound of 3, when $\mathbf{A}'_{\mathcal{S}} = [1 \quad 2]$.

Example 3: We now consider the weight matrix

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 3 & 0 & 3 \\ 1 & 1 & 0 & 1 & 3 \\ 1 & 2 & 0 & 2 & 1 \end{bmatrix}$$

which resembles the rate-$1/2$ AR4JA protomatrix in (1). Treating the code as unpunctured, Theorem 9 produces a minimum distance upper bound of 30, while Theorem 12 produces a substantially tighter upper bound of 10. The reason for the difference is that the permanents of Theorem 9 produce large values with the top row of $\mathbf{A}$ present. Theorem 12 will remove the top row of $\mathbf{A}$ when the chosen column set is $\mathcal{S} = \{0, 1, 3\}$, yielding the tighter bound.

VI. QC Expansion of the AR4JA Family of Codes

A direct QC expansion of the AR4JA protographs shown in Figs. 4 and 5 will generate a QC-LDPC code. Applying Theorems 9 and 12 to the AR4JA protomatrices (1)–(3) yields an upper bound of 10 on the minimum distance for all code rates, independent of block length. A minimum distance of 10 or less is rather small for the large block lengths desired, motivating the consideration of a more involved expansion procedure.

In fact, the construction of the AR4JA codes defined in [13] makes use of a two-step expansion process. After a first QC expansion ("lifting") by a factor of 4, a larger weight matrix is obtained, as illustrated for rate-$1/2$ by the matrix

$$\mathbf{A} = \left[\begin{array}{cccccccccccccccccccc} 0&0&0&0 & 0&0&0&0 & 1&0&0&0 & 0&0&0&0 & 1&0&0&1 \\ 0&0&0&0 & 0&0&0&0 & 0&1&0&0 & 0&0&0&0 & 1&1&0&0 \\ 0&0&0&0 & 0&0&0&0 & 0&0&1&0 & 0&0&0&0 & 0&1&1&0 \\ 0&0&0&0 & 0&0&0&0 & 0&0&0&1 & 0&0&0&0 & 0&0&1&1 \\ 1&0&0&0 & 1&0&0&0 & 0&0&0&0 & 1&0&0&0 & 0&1&1&1 \\ 0&1&0&0 & 0&1&0&0 & 0&0&0&0 & 0&1&0&0 & 1&0&1&1 \\ 0&0&1&0 & 0&0&1&0 & 0&0&0&0 & 0&0&1&0 & 1&1&0&1 \\ 0&0&0&1 & 0&0&0&1 & 0&0&0&0 & 0&0&0&1 & 1&1&1&0 \\ 1&0&0&0 & 0&0&1&1 & 0&0&0&0 & 1&1&0&0 & 1&0&0&0 \\ 0&1&0&0 & 1&0&0&1 & 0&0&0&0 & 0&1&1&0 & 0&1&0&0 \\ 0&0&1&0 & 1&1&0&0 & 0&0&0&0 & 0&0&1&1 & 0&0&1&0 \\ 0&0&0&1 & 0&1&1&0 & 0&0&0&0 & 1&0&0&1 & 0&0&0&1 \end{array}\right]. \qquad (23)$$

This is a so-called *type-1* weight matrix—that is, it contains only ones and zeros—implying that the associated protograph does not have parallel edges [9].

According to the CCSDS standard, the weight matrices so obtained, such as (23), are considered to be protomatrices themselves, that are then expanded in a second QC expansion to create QC-LDPC codes with three block lengths, corresponding to $k = 1024$, 4096, and 16 384 information bits. For example, quasi-cyclically expanding (23) by a factor of $N = 128$ and puncturing the last four columns of (23) yields the $(n, k) = (2048, 1024)$ AR4JA code. In this final expansion, the binary parity-check matrix $\mathbf{H}$ is constructed by replacing each 1 entry in (23) by a cyclic permutation matrix selected using a variation on the ACE algorithm. These codes are QC with a subblock size equal to the second step expansion factor (*e.g.*, $N = 128$).

TABLE I
MINIMUM DISTANCE OF AR4JA PROTOMATRICES AFTER FIRST QC EXPANSION (INDEPENDENT OF BLOCK LENGTH)

Code Rate r	Upper Bounds by Theorems 9 and 12	Num. of sets $\mathcal{S}$ of size $J+1$ in $[L]$
$1/2$	66	7.8×10^4
$2/3$	58	3.7×10^7
$4/5$	56[a]	5.2×10^{10}

[a]Computations are not exhaustive in sets $\mathcal{S}$ due to complexity.

In other words, the two-step process is not equivalent to any single-step QC expansion.

To compute length-independent minimum distance bounds for the AR4JA codes specified in the CCSDS standard using the techniques we have developed in this paper, the protomatrices such as (23) should be used. The resulting upper bounds, shown in Table I, range from 56 to 66. Note that, prior to [9], the tightest known upper bound on the minimum distance of QC-LDPC codes was $(J+1)!$ in the case when the protomatrices are all-ones [6], [8]. (We find in [9] that $d_{\min} \leq (J+1)!$ also holds for the more general case of type-1 protomatrices.) For the protomatrix of (23), where $J = 12$, this would yield the extremely loose upper bound $d_{\min} \leq 6.2 \times 10^9$.

This example points to the potential advantage of a two-step QC expansion, as suggested by the increase from 10 to the range 56–66 in the minimum distance upper bound. It also illustrates the strength of the general class of hierarchical QC-LDPC codes that have recently been examined in [20].

In order to reduce the computation time required to produce the results in Table I, a number of techniques were used. For larger weight matrices, if we assume that calculations are dominated by the computation time t_J for the $J \times J$ permanent, then the total time to evaluate Theorem 9 is $t_J(J+1)\binom{L}{J+1}$. The final term $\binom{L}{J+1}$ is the number of sets $\mathcal{S}$ in the weight matrix $\mathbf{A}$ and is shown in the right column of Table I for AR4JA. For the computations of interest, with $J = 12$, we built a simple recursive routine with $t_{12} = 44\,\mu$s for computing sparse permanents (as measured on a 2.6 GHz CPU). Thus, the estimated time for computing the rate-$1/2$ results in Table I is 44 s, while we measured an actual run-time of 53 s, including bookkeeping and set manipulations. On the other hand, for the rate-$4/5$ construction, the estimated time to completely evaluate the upper bound of Theorem 9 is 344 days. Therefore, we selectively directed the computations, yielding the results shown in Table I. These selective computations were performed by choosing sets $\mathcal{S}$ which, based upon the findings from the rate-$2/3$ code, we thought would yield the smallest value permanents in evaluating (15). Additional efforts at each code rate to recompute the minimum distance bounds using the row elimination logic of Theorem 12 did not yield tighter results with the AR4JA weight matrices.

VII. Numerical Results Obtained by Search

In this section, we present numerical results on the minimum distance of AR4JA codes obtained by means of computer search for low-weight codewords. We also examine bounds on the girth of AR4JA codes.

TABLE II
DISTANCE OF CCSDS AR4JA PARITY-CHECK MATRIX

Code Rate	Minimum Distance U.B. by Searching		Stopping Distance U.B. by Searching	
r	$k = 1024$	$k = 4096$	$k = 1024$	$k = 4096$
$1/2$	52	63	50	63[a]
$2/3$	26	–	23	62[b]
$4/5$	13	27	11	25

[a]The smallest stopping set found was a codeword.

[b]Beyond the upper bound shown in Table I.

A. Distance Bounds From Codeword Search for AR4JA Codes

Several papers, including [21]–[23], have described search techniques to find the minimum distance and/or stopping distance of general LDPC codes. To validate our bounds on the minimum Hamming distance, we utilized the error impulse and decoding algorithm of [21] to conduct a nonexhaustive search for small stopping sets, and then examined the results to identify the codewords. We modified the algorithm to take advantage of QC symmetry by skipping impulse combinations that are identical after cyclically shifting every subblock. We also broadened the search space by increasing the value of the parameters in [21] corresponding to the number of iterations I and the maximum threshold T. In addition, before the erasure decoding step, we erased the punctured symbols in addition to the symbols already erased by the algorithm. The resulting upper bounds on the minimum distance and the stopping distance obtained by this search methodology are summarized in Table II.

It should be noted that, in [21], the search algorithm was applied to rate-$1/2$ codes up to a minimum distance of 19, and the estimated weight spectrum results were listed only up to a maximum codeword weight of 25. It may be the case, then, that the application of this method to some of the AR4JA codes under consideration here may be pushing the algorithm beyond its effective range. Thus, further searching may turn up lower weight codewords and stopping sets than we have found.

We note that, as a consequence of the quasi-cyclicity of the code, when a codeword of a QC-LDPC code is located by our search technique, it is an indication of a group of codewords with the same weight. For instance, the low-weight codewords of the rate-$4/5, k = 1024$ CCSDS AR4JA code generally occur as a set of $N = 32$ cyclically shifted versions of a base codeword. On occasion, when all subblocks are periodic with a common period, the cyclically shifted codeword returns to the base codeword after only a fraction of N shifts. Accounting for this, we tabulated all distinct low-weight codewords found using our search algorithm for three of the AR4JA parity-check matrices given in the CCSDS standard. The estimated weight spectra, based upon the codewords we identified during our search, are shown in Tables III–V.

Fig. 6 summarizes our minimum distance results for rate-$1/2$ AR4JA codes as a function of the block length n. First, the upper bound of 66 obtained from the weight matrix using Theorems 9 and 12 is plotted as a horizontal line. The points indicated by the symbol × represent the minimum distance upper bounds based on the QC polynomial parity-check matrices from Theorem 7. These bounds are 62, 63, and 65 for the block lengths 2048,

TABLE III
ESTIMATED WEIGHT SPECTRUM FOR CCSDS AR4JA RATE-$4/5$, $k = 1024$ USING SEARCH PARAMETERS: $I = 150$ AND $T = 10$

	Hamming Weight				**Search**
	13	14	15	16	**Time**
Num. of Codewords	32	256	128	400	1.5 hrs

TABLE IV
ESTIMATED WEIGHT SPECTRUM FOR CCSDS AR4JA RATE-$4/5$, $k = 4096$ USING SEARCH PARAMETERS: $I = 300$ AND $T = 19$

	Hamming Weight			**Search**
	27	28	29	**Time**
Num. of Codewords	128	0	0	82 hrs

TABLE V
ESTIMATED WEIGHT SPECTRUM FOR CCSDS AR4JA RATE-$2/3$, $k = 1024$ USING SEARCH PARAMETERS: $I = 300$ AND $T = 19$

	Hamming Weight				**Search**
	26	29	31	32	**Time**
Num. of Codewords	64	128	64	64	5 hrs

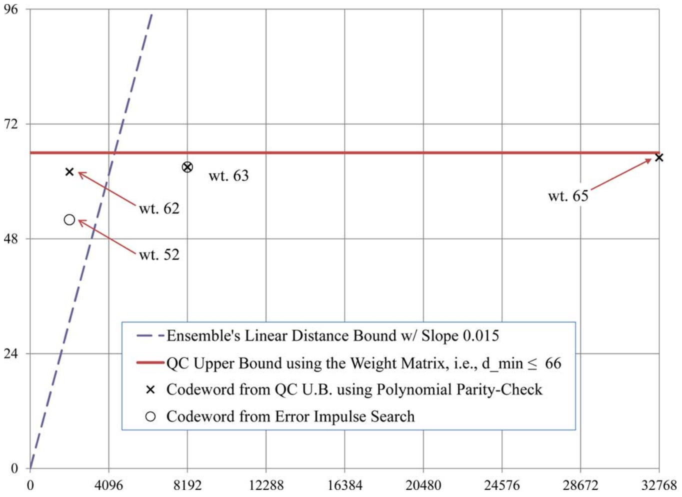


Fig. 6. Minimum distance bounds versus block length for rate-$1/2$ AR4JA.

8192, and 32 768, respectively. Finally, the codeword weights found by our search technique, as shown in Table II, are designated by the symbol $\circ$. Note that the smallest codeword weights found in our search are fairly close to the length-independent bound of 66 for all block lengths. We note that while our codeword searches generally used error impulse pairs, the codeword of weight 52 at the smallest block length was found using impulse triplets.

Divsalar *et al.* showed that most codes in the ensemble of certain protograph-based codes, including the AR4JA codes, have minimum distance linearly increasing with block length [10]–[12]. Specifically, by upper bounding the ensemble average weight enumerator, they were able to prove that $\Pr\{d_{\min} < \delta_{\min} n\} \rightarrow 0$ exponentially fast as $n \rightarrow \infty$, for some constant $\delta_{\min} > 0$. For the rate-$1/2$ AR4JA-based ensemble of codes, they computed the value $\delta_{\min} = 0.015$. The corresponding linear growth of the ensemble minimum-distance is plotted in Fig. 6 as a dashed line.

TABLE VI
GIRTH OF THE CCSDS AR4JA CODES

Information Bits k	**Measured Girth (Upper Bound)**		
	$r = 1/2$	$r = 2/3$	$r = 4/5$
1024	6 (12)	4 (10)	4 (8)
4096	8 (14)	6 (10)	4 (10)
16384	10 (16)	6 (12)	4 (10)
QC Limit of Protomatrix	12	12	12

(#) Denotes upper bound computed by tree method [1], [16]

It can be seen that for the QC-AR4JA codes that appear in the standard, our bounds are tighter than the linearly increasing ensemble bound for $n \geq 4400$ bits. By examining the probability that a random expansion is QC, we can shed some light on this possibly surprising situation. Consider the expansion of each 1 entry in a protomatrix such as (23) by a factor N. There are N cyclic permutation matrices to choose from and $N!$ general permutation matrices. Thus, a randomly chosen permutation matrix has only a probability of $1/(N-1)!$ of being cyclic. This probability goes to zero superexponentially fast. Since the QC class of expansions is such a small fraction of the ensemble of all possible expansions, one cannot claim with certainty that the probabilistic bound of Divsalar *et al.* applies to the resulting class of codes.

B. Bounds on the Girth of AR4JA Codes

In this section, we show that the two-step expansion approach was essential in order to achieve girths beyond 6 for larger block lengths of the rate-$1/2$ AR4JA codes. Recall that the *girth* of a code denotes the length of the shortest cycle in its Tanner graph. Table VI summarizes the results of our calculations of the girth of the AR4JA codes. The girths of the standardized codes for each block length and code rate are shown in the table. Also shown, in parentheses, are upper bounds obtained by the tree method of [1] and [16], which we briefly describe. In the protograph, the neighborhood of any node can be diagrammed as a tree. We can measure how tall this tree is at a given number of nodes corresponding to the specified block length. We do this for each node type in the protograph and select the smallest as an upper bound on girth that would apply to any possible expansion method. These are the upper bounds recorded in the table.

We also determined upper bounds on the girth that derive from properties of QC expansions. Since the AR4JA protomatrices of (1)–(3) all contain the element 3, the girth of the derived graphs obtained by QC expansion cannot exceed 6 (see, for example, [9]). However, since the codes in the CCSDS standard use a two-step expansion, we must examine the intermediate protomatrices such as (23) to evaluate the girth. We find that they contain the submatrix

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}^T$$

at every code rate. This limits the girth of the QC expansion to a maximum of 12, independent of block length [8], [24], [25], as shown in the final row of Table VI.

VIII. Conclusion

This work has extended the upper bounds on the minimum distance of QC-LDPC codes developed in [9] to the class of punctured QC-LDPC codes. We have also tightened those distance bounds in situations where the codes are derived from protographs whose protomatrices contain many zeros.

We evaluated the minimum distance upper bounds for the AR4JA codes specified in the CCSDS standard for deep space communication. Our results show that the use of a two-step expansion in the definition of these codes was critical to achieve reasonably high minimum distance. On the other hand, we have also shown that the minimum distance of the standardized QC AR4JA codes does not grow with block length, even though the asymptotic ensemble minimum distance of AR4JA codes grows linearly in the block length [10]–[12]. Nevertheless, the minimum distance of the CCSDS codes is likely high enough for practical purposes.

The bounds developed here and in [9] can be useful tools in evaluating future QC-LDPC code designs, both punctured and unpunctured.

Appendix
Proof of Lemma 1

In this appendix, we state several properties of matrices over a commutative ring with unity. Several terms defined in Sections III and IV will be used here, such as determinant, unit, zero divisor, and independence.

A. Matrices Over Commutative Rings

Let R be a commutative ring and let $\mathcal{M}_N(R)$ be the ring of $N \times N$ matrices over R. The determinant of a matrix $\mathbf{A} \in \mathcal{M}_N(R)$ is denoted by $\det(\mathbf{A})$. Given matrices $\mathbf{A}, \mathbf{B} \in \mathcal{M}_N(R)$, we have the identity [19, Sec. 12.2]

$$\det(\mathbf{AB}) = \det(\mathbf{A})\det(\mathbf{B}). \tag{24}$$

If R is a commutative ring with unit element 1, then a matrix $\mathbf{A}$ is a unit in $\mathcal{M}_N(R)$ if and only if $\det(\mathbf{A})$ is a unit in R. To see this, suppose that $\mathbf{A}$ has inverse $\mathbf{A}^{-1}$. From (24), we conclude that $\det(\mathbf{A}^{-1})\det(\mathbf{A}) = \det(\mathbf{I}) = 1$, where $\mathbf{I}$ is the $N \times N$ identity matrix. This implies that $\det(\mathbf{A})$ is a unit in R. Conversely, from Cramer's rule, we have $\det(\mathbf{A})\mathbf{I} = \mathbf{A}\mathrm{adj}(\mathbf{A})$, where $\mathrm{adj}(\mathbf{A})$ is the adjugate (or classical adjoint) of $\mathbf{A}$ [19]. If $\det(\mathbf{A})$ is a unit in R, then $\mathbf{A}$ is invertible, with inverse $\mathbf{A}^{-1} = (\det(\mathbf{A}))^{-1}\mathrm{adj}(\mathbf{A})$.

Consider the free module of rank N over R, that is, $R^N = \{\mathbf{r} = (r_0, \ldots, r_{N-1}) | r_i \in R, \forall i = 0, \ldots, N-1\}$. The set $\mathcal{V} = \{\mathbf{v}_0, \ldots, \mathbf{v}_{n-1}\}$, where $\mathbf{v}_i \in R^N, i = 0, \ldots, n-1$, *generates* R^N if every $\mathbf{r} \in R^N$ can be expressed as a linear combination of members of the set $\mathcal{V}$, *i.e.*, $\mathbf{r} = \sum_{i=0}^{n-1} a_i \mathbf{v}_i$, with $a_i \in R$, for all $i = 0, \ldots, n-1$. Furthermore, a generating set $\mathcal{V}$ is a *basis* for R^N if it is also independent. The *standard basis* $\{\mathbf{e}_0, \ldots, \mathbf{e}_{N-1}\}$ of R^N is simply the set of rows of the $N \times N$ identity matrix $\mathbf{I}$.

The following theorem is a reformulation of material from [26, Ch. 5].

Theorem 13: Let R be a commutative ring with unity, and let $\mathbf{A} \in \mathcal{M}_N(R)$ have rows $\mathbf{v}_0, \ldots, \mathbf{v}_{N-1} \in R^N$ and columns $\mathbf{w}_0, \ldots, \mathbf{w}_{N-1} \in R^N$. The following statements are equivalent.

1) $\mathbf{A}$ is invertible, *i.e.,* a unit in $\mathcal{M}_N(R)$.
2) $\det \mathbf{A}$ is a unit in R.
3) The set $\{\mathbf{v}_0, \ldots, \mathbf{v}_{N-1}\}$ generates R^N.
4) The set $\{\mathbf{w}_0, \ldots, \mathbf{w}_{N-1}\}$ generates R^N.

Proof: We have already shown in the discussion above the equivalence between 1) and 2). We now prove that 1) implies 3). Namely, if $\mathbf{A}$ is invertible, then for any $\mathbf{r} \in R^N$ we can write

$$\mathbf{r} = \mathbf{rI} = (\mathbf{rA}^{-1})\mathbf{A}.$$

Conversely, if the rows of $\mathbf{A}$ generate R^N, then they must generate the elements of the standard basis. Given vectors $\{\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}\} \in R^N$ satisfying $\mathbf{u}_i\mathbf{A} = \mathbf{e}_i$, for all $i = 0, \ldots, N-1$, the inverse matrix $\mathbf{A}^{-1}$ is the matrix whose rows are the vectors $\mathbf{u}_0, \ldots, \mathbf{u}_{N-1}$.

Finally, the equivalence of 3) and 4) follows from the fact that $\det(\mathbf{A}) = \det(\mathbf{A}^T)$, which implies that $\det(\mathbf{A}^T)$ is a unit in R if and only if $\det(\mathbf{A})$ is. ■

B. Finite Commutative Rings

In our application, we consider finite rings with unity. As already noted, every nonzero element of a finite ring must be either a unit or a zero divisor.

Theorem 14: Let R be a finite commutative ring with unity and let $\mathbf{A} \in \mathcal{M}_N(R)$. The rows $\mathbf{v}_0, \ldots, \mathbf{v}_{N-1}$ of $\mathbf{A}$ generate R^N if and only if they are independent.

Proof: If the rows of $\mathbf{A}$ generate R^N, then, by Theorem 13, $\mathbf{A}$ is invertible and not a zero divisor in $\mathcal{M}_N(R)$. Therefore, if a matrix $\mathbf{B} \in \mathcal{M}_N(R)$ satisfies $\mathbf{BA} = \mathbf{0}$, then $\mathbf{B} = \mathbf{0}$, proving that the rows of $\mathbf{A}$ are independent. Conversely, suppose the rows of $\mathbf{A}$ are independent. If they do not generate R^N, then, by the pigeonhole principle, there exist distinct vectors $\mathbf{r}, \mathbf{s} \in R^N$ such that $\mathbf{rA} = \mathbf{sA}$. This implies that $\mathbf{rA} - \mathbf{sA} = (\mathbf{r} - \mathbf{s})\mathbf{A} = (0, \ldots, 0)$. This contradicts the assumed independence of the rows of $\mathbf{A}$, so the rows of $\mathbf{A}$ must generate R^N. ■

Theorems 13 and 14 imply that if R is a finite commutative ring with unity and $\mathbf{A} \in \mathcal{M}_N(R)$, then the determinant $\det(\mathbf{A})$ is zero or a zero divisor in R if and only if the rows of $\mathbf{A}$ are dependent. Of course, the ring used throughout this work, $\mathbb{F}_2[x]/\langle x^N - 1\rangle$, is a finite commutative ring with unity. This completes the proof of Lemma 1.

Example 4 (Integer and quotient rings):

a) Consider the infinite ring of integers with $R = \mathbb{Z}$. Define the matrix

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}.$$

Note that $\det(\mathbf{A}) = 2$, and that 2 is not a unit in $\mathbb{Z}$. By Theorem 13, the matrix $\mathbf{A}$ is not invertible and, therefore, its rows cannot generate all of $\mathbb{Z}^2$. In particular, no linear combination of the rows of $\mathbf{A}$ can generate odd integers in the second coordinate. On the other hand, the rows are independent, as can be easily verified.

b) Next, consider $R = \mathbb{Z}/3\mathbb{Z}$, the finite ring of integers modulo 3. Here, the ring element 2 is a unit, since $2 \cdot 2 = 1$. By Theorem 13, $\mathbf{A}$ is invertible and, therefore, its rows generate R^2. In accordance with Theorem 14, the rows are independent.

c) Finally, consider $R = \mathbb{Z}/6\mathbb{Z}$, the finite ring of integers modulo 6. In this case, 2 is a zero divisor, so, by Theorem 13, the rows of $\mathbf{A}$ do not generate R^2. By Theorem 14, there is at least one dependent row. Specifically, multiplying the second row by the scalar $3 \in R$ yields the zero element $(0,0) \in R^2$.

Example 5 (Polynomial and quotient rings):

a) Consider the infinite polynomial ring $R = \mathbb{F}_2[x]$. Define the square matrix

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 0 & 1+x \end{bmatrix}.$$

Note that $\det(\mathbf{A}) = 1+x$, which is not a unit in R. Therefore, by Theorem 13, the matrix $\mathbf{A}$ is not invertible and its rows do not generate R^2. Yet, the rows are independent.

b) Consider the finite quotient ring $R = \mathbb{F}_2[x]/\langle x^N - 1\rangle$. In this ring, $1 + x$ is a zero divisor. Therefore, $\mathbf{A}$ is not invertible and the rows do not generate R^2. Since R is finite, Theorem 14 implies that the set of rows is dependent. Specifically, multiplying the second row by the scalar $x^{N-1} + x^{N-2} + \cdots + x + 1 \in R$ yields the zero element $(0,0) \in R^2$.

Acknowledgment

The authors gratefully acknowledge contributions to the proofs by Pascal Vontobel and Lance Small. The authors would also like to thank Dariush Divsalar for useful discussions.

References

[1] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA, USA: MIT Press, 1963.

[2] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vol. IT-27, no. 5, pp. 533–547, Sep. 1981.

[3] T. J. Richardson, "Error-floors of LDPC codes," in *Proc. 41st Annu. Allerton Conf.*, Monticello, IL, USA, Oct. 2003, pp. 1426–1435.

[4] X.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth Tanner graphs," *IEEE Trans. Inf. Theory*, vol. 51, no. 1, pp. 386–398, Jan. 2005.

[5] T. Tian, C. R. Jones, J. D. Villasenor, and R. D. Wesel, "Selective avoidance of cycles in irregular LDPC code construction," *IEEE Trans. Commun.*, vol. 52, no. 8, pp. 1242–1247, Aug. 2004.

[6] D. J. C. MacKay and M. C. Davey, "Evaluation of Gallager codes for short block length and high rate applications," in *Codes, Systems and Graphical Models (Minneapolis, MN, 1999)*, B. Marcus and J. Rosenthal, Eds. New York, NY, USA: Springer-Verlag, 2000, pp. 113–130.

[7] R. M. Tanner, D. Sridhara, A. Sridharan, T. E. Fuja, and D. J. Costello Jr., "LDPC block and convolutional codes based on circulant matrices," *IEEE Trans. Inf. Theory*, vol. 50, no. 12, pp. 2966–2984, Dec. 2004.

[8] M. P. C. Fossorier, "Quasi-cyclic low-density parity-check codes from circulant permutation matrices," *IEEE Trans. Inf. Theory*, vol. 50, no. 8, pp. 1788–1793, Aug. 2004.

[9] R. Smarandache and P. O. Vontobel, "Quasi-cyclic LDPC codes: Influence of proto-and Tanner-graph structure on minimum Hamming distance upper bounds," *IEEE Trans. Inf. Theory*, vol. 58, no. 2, pp. 585–607, Feb. 2012.

[10] D. Divsalar, S. Dolinar, and C. R. Jones, "Construction of protograph LDPC codes with linear minimum distance," in *Proc. IEEE Int. Symp. Inf. Theory*, Seattle, WA, USA, Jul. 2006, pp. 664–668.

[11] D. Divsalar, S. Dolinar, C. R. Jones, and K. Andrews, "Capacity-approaching protograph codes," *IEEE J. Sel. Areas Commun.*, vol. 27, no. 6, pp. 876–888, Aug. 2009.

[12] S. Abu-Surra, D. Divsalar, and W. E. Ryan, "Enumerators for protograph-based ensembles of LDPC and generalized LDPC codes," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 858–886, Feb. 2011.

[13] TM Synchronization and Channel Coding. Reston, VA, Consultative Committee for Space Data Systems, Recommended Standard 131.0-B-2, Aug. 2011 [Online]. Available: http://public.ccsds.org/publications/archive/131x0b2ec1.pdf

[14] J. Thorpe, Low density parity check (LDPC) codes constructed from protographs, Jet Propulsion Lab., Pasadena, CA, USA, Aug. 15, 2003, INP Progress Report 42–154.

[15] T. J. Richardson and R. L. Urbanke*, Modern Coding Theory*. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[16] S. L. Sweatlock, *Asymptotic weight analysis of LDPC code ensembles*, Ph.D. dissertation. Pasadena, USA, Dept. Appl. Comput. Math., California Inst. Technol., Apr. 2008 [Online]. Available: http://thesis.library.caltech.edu/1898/

[17] W. W. Peterson and E. J. Weldon, Jr.*, Error-Correcting Codes*, 2nd ed. Cambridge, MA, USA: MIT Press, 1972.

[18] C. C. Pinter*, A Book of Abstract Algebra*, 2nd ed. New York, NY, USA: McGraw-Hill, 1990.

[19] M. Artin*, Algebra*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1991.

[20] Y. Wang, S. C. Draper, and J. S. Yedidia, "Hierarchical and high-girth QC LDPC codes," *IEEE Trans. Inf. Theory*, accepted for publication.

[21] G. Richter, "Finding small stopping sets in the Tanner graphs of LDPC codes," in *Proc. 4th Int. Symp. Turbo Codes*, Munich, Germany, Apr. 2006, pp. 1–5.

[22] X.-Y. Hu, M. P. C. Fossorier, and E. Eleftheriou, "On the computation of the minimum distance of low-density parity-check codes," in *Proc. IEEE Int. Conf. Commun.*, Paris, France, Jun. 2004, vol. 2, pp. 767–771.

[23] D. Declercq and M. P. C. Fossorier, "Improved impulse method to evaluate the low weight profile of sparse binary linear codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Toronto, ON, Canada, Jul. 2008, pp. 1963–1967.

[24] H. Park, S. Hong, J.-S. No, and D.-J. Shin, "Protograph design with multiple edges for regular QC LDPC codes having large girth," in *Proc. IEEE Int. Symp. Inf. Theory*, St. Petersburg, Russia, Aug. 2011, pp. 918–922.

[25] R. M. Tanner, D. Sridhara, and T. E. Fuja, "A class of group-structured LDPC codes," in *Proc. Int. Symp. Commun. Theory Appl.*, Ambleside, U.K., Jul. 2001, pp. 365–370.

[26] E. H. Connell, *Elements of Abstract and Linear Algebra*, Mar. 2004, unpublished [Online]. Available: http://www.math.miami.edu/~ec/book/

Brian K. Butler (S'90–M'90–SM'01) received the B.S. degree in engineering from Harvey Mudd College, Claremont, CA, in 1989, and the M.S. degree in electrical engineering from Stanford University, Stanford, CA, in 1990. Presently, he is a Ph.D. Candidate in electrical and computer engineering at the University of California, San Diego, where he is affiliated with the Center for Magnetic Recording Research. His research interests include coding, modulation, and the design of wireless systems and receivers.

He was with QUALCOMM, Inc., San Diego, CA, as an Engineering Intern in 1989 and more permanently from 1990 to 2008. He worked on a variety of CDMA cellular and other wireless systems, including their product designs. He performed system simulation, design verification, algorithm design, and field testing of the early CDMA cellular system. He designed and led portions of the air-interface and the ASIC devices on the GLOBALSTAR CDMA-LEO satellite project. He contributed to the CDMA air-interface designs, phone designs, and the ASIC designs over many generations. From 1996 to 2005, he held the position of Vice President of Engineering in the ASIC division, leading the communication systems engineering department and the modem technology team. He led systems engineering on and was co-project-engineer of the first 3G-1x product and the 1x-EV-DV prototype developments. From 2005 to 2008, he contributed as a VP in QUALCOMM Corporate R&D, including as project engineer of the 4G-LTE (3GPP E-UTRA) effort.

Mr. Butler holds 37 issued U.S. patents and several foreign.

Paul H. Siegel (M'82–SM'90–F'97) received the S.B. and Ph.D. degrees in mathematics from the Massachusetts Institute of Technology (MIT), Cambridge, in 1975 and 1979, respectively.

He held a Chaim Weizmann Postdoctoral Fellowship at the Courant Institute, New York University. He was with the IBM Research Division in San Jose, CA, from 1980 to 1995. He joined the faculty of the School of Engineering at the University of California, San Diego, in July 1995, where he is currently Professor of Electrical and Computer Engineering. He is affiliated with the California Institute of Telecommunications and Information Technology, the Center for Wireless Communications, and the Center for Magnetic Recording Research, where he holds an endowed chair and served as Director from 2000 to 2011. His primary research interests lie in the areas of information theory and communications, particularly coding and modulation techniques, with applications to digital data storage and transmission.

Prof. Siegel was a member of the Board of Governors of the IEEE Information Theory Society from 1991 to 1996 and from 2009 to 2011. He was re-elected for a three-year term in 2012. He served as Co-Guest Editor of the May 1991 Special Issue on "Coding for Storage Devices" of the IEEE TRANSACTIONS ON INFORMATION THEORY. He served the same Transactions as Associate Editor for Coding Techniques from 1992 to 1995, and as Editor-in-Chief from July 2001 to July 2004. He was also Co-Guest Editor of the May/September 2001 two-part issue on "The Turbo Principle: From Theory to Practice" of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS. He was corecipient, with R. Karabed, of the 1992 IEEE Information Theory Society Paper Award and shared the 1993 IEEE Communications Society Leonard G. Abraham Prize Paper Award with B. Marcus and J. K. Wolf. With J. B. Soriaga and H. D. Pfister, he received the 2007 Best Paper Award in Signal Processing and Coding for Data Storage from the Data Storage Technical Committee of the IEEE Communications Society. He holds several patents in the area of coding and detection, and was named a Master Inventor at IBM Research in 1994. He is a member of Phi Beta Kappa and the National Academy of Engineering.