Fault-Tolerant Dot-Product Engines

Ron M. Roth, *Fellow, IEEE*

***Abstract*—Coding schemes are presented that provide the ability to correct and detect computational errors while using dot-product engines for integer vector–matrix multiplication. Both the L_1-metric and the Hamming metric are considered.**

***Index Terms*—Analog arithmetic circuits, Berlekamp codes, In situ computing, Lee metric, Vector–matrix multiplication.**

I. Introduction

We consider the following computational model. For an integer $q \geq 2$, let Σ_q denote the subset $[q\rangle = \{0, 1, \ldots, q-1\}$ of the integer set $\mathbb{Z}$. Also, let ℓ and n be fixed positive integers. A *dot-product engine* (in short, DPE) is a device which accepts as input an $\ell \times n$ matrix $A = (a_{i,j})_{i \in [\ell\rangle, j \in [n\rangle}$ over Σ_q and a row vector $\boldsymbol{u} = (u_i)_{i \in [\ell\rangle} \in \Sigma_q^\ell$, and computes the vector–matrix product $\boldsymbol{c} = \boldsymbol{u}A$, with addition and multiplication carried out over $\mathbb{Z}$. Thus, $\boldsymbol{c} = (c_j)_{j \in [n\rangle}$ is an integer vector in $\mathbb{Z}^n$ (more specifically, over $\Sigma^n_{\ell(q-1)^2+1}$). In the applications of interest, the matrix A is modified much less frequently than the input vector $\boldsymbol{u}$ (in some applications, the matrix A is determined once and then remains fixed, in which case only $\boldsymbol{u}$ is seen as input). Typically, the alphabet size[1] q is a power of 2.

In recent proposals of nanoscale implementations of a DPE, the matrix A is realized as a crossbar array consisting of ℓ row conductors, n columns conductors, and programmable nanoscale resistors (e.g., memristors) at the junctions, with the resistor at the junction (i,j) set to have conductance, $G_{i,j}$, that is proportional to $a_{i,j}$. Each entry u_i of $\boldsymbol{u}$ is fed into a digital-to-analog converter (DAC) to produce a voltage level that is proportional to u_i. The product, $\boldsymbol{u}A$, is then computed by reading the currents at the (grounded) column conductors, after being fed into analog-to-digital converters (ADCs); see Figure 1. For early implementations and applications of DPE's, as well as recent ones, see, for example, [4], [11], [12], [18], and [23].

Inaccuracies while programming the resistors in the crossbar and noise while reading the currents are examples of factors that can affect the accuracy of the computation. Specifically, the actually-read row vector, $\boldsymbol{y} = (y_j)_{j \in [n\rangle} \in \mathbb{Z}^n$, may differ from the correct vector, $\boldsymbol{c} = \boldsymbol{u}A$. The error vector is defined as the following vector in $\mathbb{Z}^n$:

$$\boldsymbol{e} = (e_j)_{j \in [n\rangle} = \boldsymbol{y} - \boldsymbol{u}A \ .$$

In such context of errors, we find it natural to define the number of errors to be the L_1-norm of $\boldsymbol{e}$:

$$\|\boldsymbol{e}\| = \|\boldsymbol{e}\|_1 = \sum_{j \in [n\rangle} |e_j| \ .$$

In our case—where $\boldsymbol{e}$ is an integer—this norm is also referred to as the Manhattan weight of $\boldsymbol{e}$, which equals the Manhattan distance between $\boldsymbol{u}A$ and $\boldsymbol{y}$. The L_1-metric has been studied quite extensively in the coding literature, along with its finite-field Lee-metric variant: see [2, Ch. 9] and [22, Ch. 10] (and Subsection IV-A below).

Another source of computational errors could be junctions in the crossbar becoming shorted due to faults in the programming process[2]. In this case, the current read in the shorted column will be above some compliance limit ("∞"), which, in turn, will flag the respective entry in $\boldsymbol{y}$ as "unavailable" or as an *erasure*.

In the other extreme, a junction in the array may become non-programmable or get stuck at an open state, in which cases

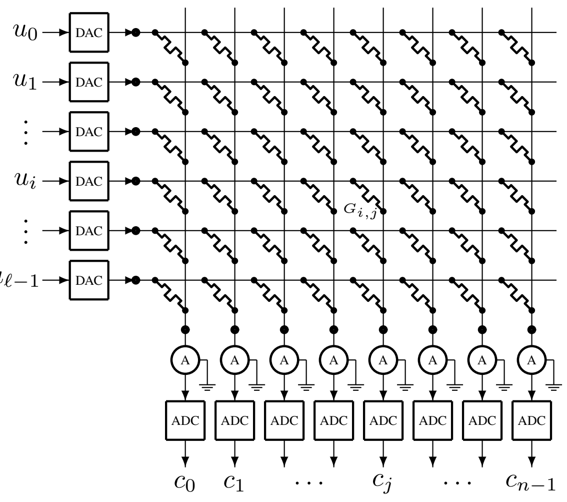

Fig. 1. Schematic diagram of a DPE implementation of the computation $\boldsymbol{u} \mapsto \boldsymbol{c} = \boldsymbol{u}A$ using an $\ell \times n$ crossbar array of memristors. The conductance $G_{i,j}$ of the memristor at each junction (i,j) is proportional to $a_{i,j}$. The circles marked "A" represent analog current measuring devices (such as transimpedance amplifiers). The current measurements can be carried out in parallel (as shown), or serially, column-by-column, using only one measuring device.

Ron M. Rothis with the Computer Science Department, Technion, Haifa 3200003, Israel. This work was done in part while he was visiting Hewlett Packard Laboratories, 1501 Page Mill Road, Palo Alto, CA 94304.. Email: ronny@cs.technion.ac.il.

Parts of this work were presented at the IEEE Int'l Symposium on Information Theory, Vail, Colorado (June 2018).

[1]One can consider the broader problem where the matrix A and the vector $\boldsymbol{u}$ are over different integer alphabets. Yet, for the sake of simplicity, we assume hereafter that those alphabets are the same. It is primarily the alphabet of the matrix that will affect the coding schemes that will be presented in this work.

[2]Shorts could also result from manufacturing defects, although conceivably these can be detected before the DPE is put into operation.

the respective entry in $\boldsymbol{y}$ could be off the correct value by as much as $\pm(q-1)^2$. Such errors could be counted through their contribution to the L_1-norm of the error vector. Alternatively, if this type of errors is predominant, one could consider the Hamming metric instead, whereby the figure of merit is the Hamming weight of $\boldsymbol{e}$, equaling the number of positions in which $\boldsymbol{y}$ differs from $\boldsymbol{u}A$ (disregarding the extent at which the values of the respective entries actually differ[3]). This Hamming metric is suitable for handling erasures as well.

In this work, we propose methods for using the DPE computational power to self-protect the computations against errors. The first k $(< n)$ entries in $\boldsymbol{c} = \boldsymbol{u}A$ will carry the (ordinary) result of the computation of interest, while the remaining $n-k$ entries of $\boldsymbol{c}$ will contain redundancy symbols, which can be used to detect or correct computational errors, assuming that the number of the latter (counted with respect to either the L_1-metric or the Hamming metric) is bounded from above by some design parameter. Specifically, the programmed $\ell \times n$ matrix A will have the structure

$$A = (A' \mid A'') \;,$$

where A' is an $\ell \times k$ matrix over Σ_q consisting of the first k columns of A, and A'' consists of the remaining $n-k$ columns; the computed output row vector for an input vector $\boldsymbol{u} \in \Sigma_q^\ell$ will then be $\boldsymbol{c} = (\boldsymbol{c}' \mid \boldsymbol{c}'')$, where the k-prefix $\boldsymbol{c}' = \boldsymbol{u}A'$ $(\in \mathbb{Z}^k)$ represents the target computation while the $(n-k)$-suffix $\boldsymbol{c}'' = \boldsymbol{u}A''$ $(\in \mathbb{Z}^{n-k})$ is the redundancy part. In this setting, A' and $\boldsymbol{u}$ are the actual inputs, and A'' will need to be computed from A', e.g., by a dedicated circuitry, prior to—or while—programming A' and A'' into the crossbar array (yet recall that it is expected that A' will be modified much less frequently than $\boldsymbol{u}$). The error decoding mechanism will be implemented by dedicated circuitry too. Clearly, we will aim at minimizing $n-k$ given the designed error correction capability.

Example 1. Let us consider the simplest case where we would like to be able to only detect one L_1-metric error. In this case, we select $n = k+1$ and let the $\ell \times 1$ matrix $A'' = (a_{i,k})_{i \in [\ell\rangle}$ be obtained from $A' = (a_{i,j})_{i \in [\ell\rangle, j \in [k\rangle}$ by

$$a_{i,k} = \Bigl(\sum_{j \in [k\rangle} a_{i,j} \Bigr) \textsc{ MOD } 2 \;, \quad i \in [\ell\rangle \;,$$

where "MOD" stands for (the binary operation of) remaindering; thus, the entries of A'' are in fact over Σ_2, and the sum of entries along each row of A is even. It follows by linearity that the sum of entries of (an error-free) $\boldsymbol{c} = (c_j)_{j \in [n\rangle} = \boldsymbol{u}A$ must be even. On the other hand, if $\boldsymbol{e} \in \mathbb{Z}^n$ is an error vector with $\|\boldsymbol{e}\| = 1$ then the sum of entries of $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e}$ will be odd. □

Observe that the contents of A'' depends on A', but should *not* depend on $\boldsymbol{u}$. In particular, A'' should be set so that the specified error correction–detection capabilities hold when $\boldsymbol{u}$ is taken to be a unit vector. Thus, for every row index i, the set of (at least) q^k possible contents of row i in A must form a subset of Σ_q^n that, by itself (and independently of the contents of the other rows in A), meets the correction–detection capabilities.

Secondly, note that a given computed k-prefix $\boldsymbol{c}' = \boldsymbol{u}A'$ can be associated with different $(n-k)$-suffixes (redundancy symbols) $\boldsymbol{c}'' = \boldsymbol{u}A''$, depending on $\boldsymbol{u}$. This is different from the common coding theory setting, where the redundancy symbols are uniquely determined by the information symbols (the latter being the counterparts of the entries of $\boldsymbol{c}'$ in our setting)[4]. For instance, if A in Example 1 is

$$A = \left(\begin{array}{ccc|c} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{array} \right)$$

(where $q = 2$, $k = 3$, and $n = 4$), then, for $\boldsymbol{u} = (0\;0\;1)$,

$$(0\;0\;1)\,A = (1\;1\;0\;0)$$

while for $\boldsymbol{u} = (1\;1\;0)$,

$$(1\;1\;0)\,A = (1\;1\;0\;2)$$

(in both cases, $\boldsymbol{c}' = (1\;1\;0)$). Indeed, we will see in the sequel some coding schemes where we will be able to recover $\boldsymbol{c}'$ correctly out of $\boldsymbol{y}$ (which will suffice for our purposes), yet we will not necessarily recover $\boldsymbol{c}''$. This means that we will need to present the error correction–detection specification of a DPE coding scheme slightly differently than usual; we do this in Section II below.

In Section III, we present methods for single-error correction and double-error detection in the L_1-metric. Methods for multiple-error correction for that metric are then discussed in Section IV. Finally, the Hamming metric is considered in Section V. We will mainly focus on a regime where the number τ of correctable errors is fixed (i.e., small) while n grows. Under these conditions, the required redundancy, $n-k$, of our methods will be of the order of $\tau \cdot \log_q n$ in the case of the L_1-metric, and approximately twice that number in the case of the Hamming metric. Moreover, both the encoding and decoding can be efficiently implemented; in particular, the decoding requires a number of integer (or finite field) arithmetic operations which is proportional to τn (and the implementation can be parallelized to a latency proportional to τ), where the operands are of the order of $\log_2 n$ bits long. In our model, we assume that the encoding and decoding circuitry are error-free (unlike [25]). In practical scenarios, this assumption could be justified by the relatively small size of the (encoding and) decoding circuitry compared to the whole array: it scales (only) linearly with n, as opposed to ℓn (moreover, it can also be pipelined to serve several arrays simultaneously).

We end this section by mentioning some classical papers on fault-tolerant vector–matrix multiplication under the Hamming metric, primarily in the context of concurrent computing structures, consisting of arrays of processors. The first paper on this subject appears to be by Huang and Abraham [14],

[3]Yet we will also consider a more general setting, where that difference is bounded by some prescribed constant.

[4]Moreover, while systematic encoding is a matter of preference in ordinary coding applications, in our setting it is actually a necessity: the benefits of using the DPE would diminish if post-processing of its output were required even when the output were error-free.

which introduced fault tolerance of one error into matrix-to-matrix multiplication by using product codes (such a scheme required redundancy in two dimensions and, therefore, did not handle the case where one of the matrices is a vector). Subsequent papers (see [1], [16], and the tutorial [24]) proposed schemes based on variants of Reed–Solomon codes, under the assumption that the error value is within $\pm\lfloor(q-1)/2\rfloor$ and q is large (e.g., $q > n$). Much more recently, this model has been revived to handle "stragglers," i.e., slow nodes that are flagged as erasures; see [5], [6], [19], [20], [26]. In contrast to the aforementioned papers, in our work we deal with the challenge of restricting to small alphabet sizes q (due to the limited precision of the computing devices at the junctions of the crossbar array); in particular, the contents of the redundancy columns—namely, the matrix A''—must be over the prescribed small alphabet too[5]. And, in addition to the Hamming metric, we study the L_1-metric in great detail as well.

II. Definitions

For integer vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ of the same length, we denote by $\mathsf{d}_{\mathcal{L}}(\boldsymbol{x}_1, \boldsymbol{x}_2)$ the L_1-distance between them, namely, $\mathsf{d}_{\mathcal{L}}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|$. The *Manhattan sphere* of radius t centered at $\boldsymbol{y} \in \mathbb{Z}^n$ is defined as the set of all vectors in $\mathbb{Z}^n$ at L_1-distance at most t from $\boldsymbol{y}$:

$$\mathcal{S}_{\mathcal{L}}(\boldsymbol{y}, t) = \{\boldsymbol{x} \in \mathbb{Z}^n \;:\; \mathsf{d}_{\mathcal{L}}(\boldsymbol{x}, \boldsymbol{y}) \le t\} \;.$$

The *volume* (size) of $\mathcal{S}_{\mathcal{L}}(\boldsymbol{y}, t)$ is known to be [8], [9]:

$$V_{\mathcal{L}}(n, t) = \sum_{i=0}^{\min\{t,n\}} 2^i \binom{n}{i}\binom{t}{i} \;. \tag{1}$$

In particular, $V_{\mathcal{L}}(n, 1) = 2n + 1$, and for any fixed t and sufficiently large n we have $V_{\mathcal{L}}(n, t) = O(n^t)$, where the hidden constant depends on t.

Turning to the Hamming metric, we denote by $\mathsf{d}_{\mathcal{H}}(\boldsymbol{x}_1, \boldsymbol{x}_2)$ the Hamming distance between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, and the Hamming sphere of radius t centered at $\boldsymbol{y} \in \mathbb{Z}^n$ is defined by

$$\mathcal{S}_{\mathcal{H}}(\boldsymbol{y}, t) = \{\boldsymbol{x} \in \mathbb{Z}^n \;:\; \mathsf{d}_{\mathcal{H}}(\boldsymbol{x}, \boldsymbol{y}) \le t\}$$

(which has infinite size when $t > 0$). In what follows, we will sometimes omit the identifier "$\mathcal{L}$" or "$\mathcal{H}$" from $\mathsf{d}(\cdot, \cdot)$ and $\mathcal{S}(\cdot, \cdot)$, if the text applies to both metrics.

Given Σ_q and positive integers ℓ, n, and $k < n$, a DPE *coding scheme* is a pair $(\mathcal{E}, \mathcal{D})$, where

- $\mathcal{E} : \Sigma_q^{\ell\times k} \to \Sigma_q^{\ell\times n}$ is an *encoding mapping* such that for every $A' \in \Sigma_q^{\ell\times k}$, the image $A = \mathcal{E}(A')$ has the form $(A' \mid A'')$ for some $A'' \in \Sigma_q^{\ell\times(n-k)}$. The set
$$\mathcal{C} = \left\{\boldsymbol{u}\,\mathcal{E}(A') \;:\; A' \in \Sigma_q^{\ell\times k},\; \boldsymbol{u} \in \Sigma_q^{\ell}\right\}$$
is the *code induced by* $\mathcal{E}$ and its members are called *codewords*. Thus, $\mathcal{C} \subseteq \Sigma_Q^n$, where $Q = \ell(q-1)^2 + 1$.
- $\mathcal{D} : \Sigma_Q^n \to \Sigma_Q^k \cup \{\text{"e"}\}$ is a *decoding mapping* (the return value "e" will designate a decoding failure).

[5]Compare with the recent work [13], which presents a nonsystematic scheme for handling stragglers where the matrix A' is (say) binary and A is over a larger—yet still relatively small—alphabet.

Note that in the above definition, the decoding mapping $\mathcal{D}$ is not a function of A' (yet one could consider also a different setting where A' is known to the decoder).

Borrowing (somewhat loosely) classical coding terms, we will refer to n and k as the *length* and *dimension*, respectively, of the coding scheme. In the context of a given coding scheme, the k-prefix (respectively, $(n-k)$-suffix) of a vector $\boldsymbol{x} \in \mathbb{Z}^n$ will be denoted hereafter by $\boldsymbol{x}'$ (respectively, $\boldsymbol{x}''$). This notational convention extends to $\ell\times n$ matrices over $\mathbb{Z}$, with A' (respectively, A'') standing for the sub-matrix consisting of the first k columns (respectively, last $n-k$ columns) of an $\ell \times n$ matrix A over $\mathbb{Z}$. Denoting row i of a matrix X by X_i, we then have $(A')_i = (A_i)'$ and $(A'')_i = (A_i)''$, for every $i \in [\ell\rangle$.

Given nonnegative integers τ and σ, a coding scheme $(\mathcal{E}, \mathcal{D})$ is said to *correct* τ *errors and detect* $\tau + \sigma$ *errors* (in the L_1-metric or the Hamming metric, depending on the context) if the following conditions hold for every computed vector $\boldsymbol{c} = \boldsymbol{u}A \in \mathcal{C}$ and the respective read vector[6] $\boldsymbol{y} \in \Sigma_Q^n$.

- *(Correction condition)* If $\mathsf{d}(\boldsymbol{y}, \boldsymbol{c}) \le \tau$, then $\mathcal{D}(\boldsymbol{y}) = \boldsymbol{c}'$.
- *(Detection condition)* Otherwise, if $\mathsf{d}(\boldsymbol{y}, \boldsymbol{c}) \le \tau+\sigma$, then $\mathcal{D}(\boldsymbol{y}) \in \{\boldsymbol{c}', \text{"e"}\}$.

That is, if the number of errors is τ or less, then the decoder must produce the correct result of the target computation; otherwise, if the number of errors is $\tau+\sigma$ or less, the decoder can flag decoding failure instead (but it cannot produce an incorrect result).

So, unlike the respective conditions for ordinary codes, the sphere $\mathcal{S}(\boldsymbol{y}, \tau)$ may contain multiple codewords of $\mathcal{C}$, yet they all must agree on their k-prefixes. Similarly, the sets $\mathcal{S}(\boldsymbol{y}, \tau)$ and $\mathcal{S}(\boldsymbol{y}, \tau+\sigma) \setminus \mathcal{S}(\boldsymbol{y}, \tau)$ may both contain codewords of $\mathcal{C}$, yet these codewords must agree on their k-prefixes.

By properly defining the *minimum distance* of $\mathcal{C}$, we can extend to our setting the well known relationship between minimum distance and correction capability. Specifically, the minimum distance of $\mathcal{C}$, denoted $\mathsf{d}(\mathcal{C})$ (with an identifier of the particular metric used), is defined as the smallest distance between any two codewords in $\mathcal{C}$ having distinct k-prefixes:

$$\mathsf{d}(\mathcal{C}) = \min_{\substack{\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{C}:\\ \boldsymbol{c}_1' \ne \boldsymbol{c}_2'}} \mathsf{d}(\boldsymbol{c}_1, \boldsymbol{c}_2) \;.$$

The following result then extends from the ordinary coding setting setting [22, p. 14, Prop. 1.5] (for completeness, we include a proof in Appendix A).

Proposition 1. *Let $\mathcal{E} : \Sigma_q^{\ell\times k} \to \Sigma_q^{\ell\times n}$ be an encoding mapping with an induced code $\mathcal{C}$, and let τ and σ be nonnegative integers such that*

$$2\tau + \sigma < \mathsf{d}(\mathcal{C}) \;.$$

Then there exists a decoding mapping $\mathcal{D} : \Sigma_Q^n \to \Sigma_Q^k \cup \{\text{"e"}\}$ such that the coding scheme $(\mathcal{E}, \mathcal{D})$ can correct τ errors and detect $\tau + \sigma$ errors.

For the special case of the Hamming metric, Proposition 1 can be generalized to handle erasures as well (see [22, p. 16, Prop. 1.7] and Appendix A).

[6]It is assumed hereafter that the entries of the received vector remain in the same alphabet, Σ_Q, as of the computed vector; while errors could push the entries to outside that range, they can always be coerced back into Σ_Q.

Proposition 2. *With $\mathcal{E}$ and $\mathcal{C}$ as in Proposition 1, let τ, σ, and ρ be nonnegative integers such that*

$$2\tau + \sigma + \rho < \mathsf{d}_{\mathcal{H}}(\mathcal{C}) \; .$$

Then there exists a decoding mapping $\mathcal{D} : \Sigma_Q^n \to \Sigma_Q^k \cup \{\text{“e”}\}$ such that the coding scheme $(\mathcal{E}, \mathcal{D})$ can correct τ errors and detect $\tau + \sigma$ errors, in the presence of up to ρ erasures.

The coding schemes that we present in upcoming sections are based on known codes, in particular known schemes for the Lee and Manhattan metrics—primarily Berlekamp codes [2, Ch. 9], [22, Ch. 10]. Yet certain adaptations are needed due to the fact that the computation of the redundancy symbols of the codewords in the induced code $\mathcal{C} = \{\boldsymbol{c} = \boldsymbol{u}\,\mathcal{E}(A')\}$ has to be done only through the computation of $A' \mapsto \mathcal{E}(A')$ (which is independent of $\boldsymbol{u}$). Moreover, the alphabet, Σ_q, of the entries of $\mathcal{E}(A')$ is smaller than the alphabet, Σ_Q, of the codewords in $\mathcal{C}$. Our coding schemes will be *separable*, in the sense that for each row index $i \in [\ell\rangle$, the contents $(\mathcal{E}(A'))_i$ will only be a function of A'_i (and not of the rest of the rows in A'); in fact, that function will be the same for all i, and will not depend on ℓ. It is expected, however, that the designed number of correctable errors, τ, will tend to increase with ℓ.

III. Single error correction in the L_1-metric

In this section, we describe a DPE coding scheme, $(\mathcal{E}_1, \mathcal{D}_1)$, for correcting one L_1-metric error; this scheme will then be extended (in Subsection III-B) to also detect two errors.

A. The coding scheme

Given an alphabet size $q \geq 2$ and a code length n, we let $m = \lceil \log_q(2n+1) \rceil$ and $k = n - m$ (thus, m will be the redundancy). Let

$$\boldsymbol{\alpha} = (\alpha_0 \; \alpha_1 \; \ldots \; \alpha_{n-1})$$

be a vector in $\mathbb{Z}^n$ that satisfies the following properties.

(i) The entries of $\boldsymbol{\alpha}$ are nonzero distinct elements in $[2n{+}1\rangle$.

(ii) For any two indexes $i, j \in [n\rangle$,
$$\alpha_i + \alpha_j \neq 2n + 1 \; .$$

(iii) $\alpha_{k+j} = q^j$, for $j \in [m\rangle$.

We will refer to the entries of $\boldsymbol{\alpha}$ as *code locators*. Code locators that satisfy conditions (i)–(iii) can be easily constructed for every $q \geq 2$ and n, except[7] when q is even and $n = q^{m-1}/2$: e.g., when $q^{m-1} \leq n$, we can take

$$\{\alpha_j\}_{j \in [n\rangle} = \{1, 2, 3, \ldots, n\} \; ,$$

otherwise,

$$\{\alpha_j\}_{j \in [n\rangle} = (\{1, 2, 3, \ldots, n\} \setminus \{2n{+}1{-}q^{m-1}\}) \cup \{q^{m-1}\}$$

(note that $q^{m-1} < 2n + 1$ and that $q^{m-2} \leq n$, yet q^{m-1} may be larger than n; in fact, this will always be the case when $q = 2$).

The encoding mapping $\mathcal{E}_1 : \Sigma_q^{\ell \times k} \to \Sigma_q^{\ell \times n}$ is defined as follows: for every $A' = (a_{i,j})_{i \in [\ell\rangle, j \in [k\rangle}$, the last m columns in $A = (A' \mid A'') = \mathcal{E}_1(A')$ are set so that

$$\begin{aligned} \sum_{j \in [m\rangle} a_{i,k+j} \underbrace{\alpha_{k+j}}_{q^j} & \\ = \; \Big(-\sum_{j \in [k\rangle} a_{i,j}\alpha_j \Big) & \; \textsc{mod} \; (2n+1) \; , \quad i \in [\ell\rangle \; , \end{aligned} \tag{2}$$

where the remainder (the result of the “MOD” operation) is taken to be in $[2n{+}1\rangle$. Simply put, $a_{i,n-1}a_{i,n-2}\ldots a_{i,k+1}a_{i,k}$ is the representation to base q (from the most-significant digit down to the least) of the remainder in $[2n{+}1\rangle$ of the (nonpositive) integer $-\sum_{j \in [k\rangle} a_{i,j}\alpha_j$, when divided by $2n{+}1$.

It follows from (2) that each $A = \mathcal{E}_1(A')$ satisfies

$$A\boldsymbol{\alpha}^{\mathsf{T}} \equiv \mathbf{0} \quad (\mathrm{mod}\; (2n+1)) \; ,$$

where $(\cdot)^{\mathsf{T}}$ denotes transposition and the congruence holds component-wise. Hence, for each codeword $\boldsymbol{c} = \boldsymbol{u}A$ in the induced code $\mathcal{C}$ we have

$$\boldsymbol{c} \cdot \boldsymbol{\alpha}^{\mathsf{T}} \equiv \boldsymbol{u}A\boldsymbol{\alpha}^{\mathsf{T}} \equiv 0 \quad (\mathrm{mod}\; (2n+1)) \; . \tag{3}$$

This, in turn, implies that for every two distinct codewords $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{C}$,

$$(\boldsymbol{c}_1 - \boldsymbol{c}_2) \cdot \boldsymbol{\alpha}^{\mathsf{T}} \equiv 0 \quad (\mathrm{mod}\; (2n+1)) \; ,$$

and, therefore, by conditions (i)–(ii) we get that $\mathsf{d}_{\mathcal{L}}(\boldsymbol{c}_1, \boldsymbol{c}_2) = \|\boldsymbol{c}_1 - \boldsymbol{c}_2\| > 2$, namely, that $\mathsf{d}_{\mathcal{L}}(\mathcal{C}) \geq 3$. We conclude from Proposition 1 that when using the encoding mapping defined by (2) to map A' into $A = \mathcal{E}_1(A')$, we should be able to correct one error; alternatively, we should be able to detect two errors. We demonstrate next a single-error-correcting decoding mapping.

Let $\boldsymbol{y} = (y_j)_{j \in [n\rangle} = \boldsymbol{c} + \boldsymbol{e} = \boldsymbol{u}A + \boldsymbol{e}$ be the read vector at the output of the DPE, where $\boldsymbol{e} \in \mathbb{Z}^n$ is an error vector having at most one nonzero entry, equaling ± 1. The decoding will start by computing the syndrome of $\boldsymbol{y}$, which is defined by

$$\begin{aligned} s &= \big(\boldsymbol{y} \cdot \boldsymbol{\alpha}^{\mathsf{T}}\big) \; \textsc{mod} \; (2n+1) \\ &= \Big(\sum_{j \in [n\rangle} y_j \alpha_j\Big) \; \textsc{mod} \; (2n+1) \; . \end{aligned}$$

We then have,

$$s \equiv \boldsymbol{u}A\boldsymbol{\alpha}^{\mathsf{T}} + \boldsymbol{e} \cdot \boldsymbol{\alpha}^{\mathsf{T}} \overset{(3)}{\equiv} \boldsymbol{e} \cdot \boldsymbol{\alpha}^{\mathsf{T}} \quad (\mathrm{mod}\; (2n+1)) \; .$$

It follows that $s = 0$ when $\boldsymbol{e} = \mathbf{0}$; otherwise, if $\boldsymbol{e}$ has ± 1 at position j (and is zero otherwise), then

$$s \equiv \pm \alpha_j \quad (\mathrm{mod}\; (2n+1)) \; .$$

Hence, due to conditions (i)–(ii), the syndrome s identifies the error location j and the error sign uniquely.

[7] We mention that we could relax condition (ii) so that it does not have to hold for indexes $i, j \in [n\rangle \setminus [k\rangle$: with such a relaxation, we might not be able to decode an error occurring in the redundancy part (recall that this is not considered a decoding failure according to our correction–detection model). However, we have chosen to keep the (slightly) stronger requirement to make a more transparent connection to Berlekamp codes. In fact, the stronger requirement makes a difference only in the exception noted here, namely, when q is even and $n = q^{m-1}/2$, in which case $2n{+}1 = q^{m-1} + 1 = \alpha_k + \alpha_{n-1}$. In fact, the dimension k in this case is $n{-}m$, which is also the dimension when the code length is $n{-}1$, namely, the same error correction capability can be achieved with one less redundancy.

```
Input: $\ell \times k$ matrix $A' = (a_{i,j})_{i\in[\ell\rangle, j\in[k\rangle}$ over $\Sigma_q$.
Output: $\ell \times n$ matrix $(A' \mid A'') = (a_{i,j})_{i\in[\ell\rangle, j\in[n\rangle}$ over $\Sigma_q$.
// $m = \lceil \log_q(2n+1) \rceil$, $k = n - m$.
// $\boldsymbol{\alpha}$ satisfies conditions (i)–(iii).
For all $i \in [\ell\rangle$ do {
    Set $(a_{i,n-1}\ a_{i,n-2}\ \ldots\ a_{i,k+1}\ a_{i,k})$ to satisfy Eq. (2).
}
```

Fig. 2. Encoding mapping $\mathcal{E}_1 : A' \mapsto (A' \mid A'')$ for single-error correction (or double-error detection).

```
Input: $\boldsymbol{y} = (\boldsymbol{y}' \mid \boldsymbol{y}'') \in \Sigma_q^n$.
Output: $\boldsymbol{w} = (w_j)_{j\in[k\rangle} \in \Sigma_q^k$, or “e” (decoding failure).
// Parameters are as defined in Figure 2.
Let $\boldsymbol{w} \leftarrow \boldsymbol{y}'$;
Let
        $s \leftarrow (\boldsymbol{y} \cdot \boldsymbol{\alpha}^{\mathsf{T}}) \text{ MOD } (2n+1)$ ;
If $s = 0$ then {    // $\boldsymbol{y}$ is error-free
    ;
}
Else if $s = \alpha_j$ for some $j \in [n\rangle$, then {
    If $j \in [k\rangle$ then let $w_j \leftarrow w_j - 1$;
}
Else if $s = 2n+1-\alpha_j$ for some $j \in [n\rangle$, then {
    If $j \in [k\rangle$ then let $w_j \leftarrow w_j + 1$;
}
Else {
    Return “e”.
}
```

Fig. 3. Decoding mapping $\mathcal{D}_1 : \boldsymbol{y} \mapsto \boldsymbol{w}$ for single-error correction.

The encoding and decoding procedures for our single-error correction scheme are summarized in Figures 2 and 3.

Example 2. Let $q = 2$ and $n = 15$, in which case $m = 5$ and $k = 10$. Select $\ell = 3$ and

$$\boldsymbol{\alpha} = \left(3\ 5\ 6\ 7\ 9\ 10\ 11\ 12\ 13\ 14 \,\middle|\, 1\ 2\ 4\ 8\ 16 \right)$$

(which satisfies conditions (i)–(iii)). Suppose that A' is the following 3×10 matrix:

$$A' = \begin{pmatrix} 1\,0\,1\,1\,0\,1\,0\,0\,1\,0 \\ 0\,0\,0\,1\,0\,1\,1\,0\,0\,1 \\ 0\,1\,0\,0\,0\,1\,0\,1\,1\,1 \end{pmatrix} .$$

For $i = 0, 1, 2$, the values at the right-hand side of (2) are given by

$$\begin{aligned} -(3+6+7+10+13) \text{ MOD } 31 &= 23 \\ -(7+10+11+14) \text{ MOD } 31 &= 20 \\ -(5+10+12+13+14) \text{ MOD } 31 &= 8 , \end{aligned}$$

and, so,

$$A = \mathcal{E}_1(A') = \left(\begin{array}{c|c} 1\,0\,1\,1\,0\,1\,0\,0\,1\,0 & 1\,1\,1\,0\,1 \\ 0\,0\,0\,1\,0\,1\,1\,0\,0\,1 & 0\,0\,1\,0\,1 \\ 0\,1\,0\,0\,0\,1\,0\,1\,1\,1 & 0\,0\,0\,1\,0 \end{array} \right) .$$

For $\boldsymbol{u} = (1\ 1\ 1)$, we get

$$\boldsymbol{c} = \boldsymbol{u}A = \left(1\ 1\ 1\ 2\ 0\ 3\ 1\ 1\ 2\ 2 \,\middle|\, 1\ 1\ 2\ 1\ 2 \right) .$$

Suppose that the read vector is

$$\boldsymbol{y} = \left(1\ 1\ 1\ 2\ 0\ 2\ 1\ 1\ 2\ 2 \,\middle|\, 1\ 1\ 2\ 1\ 2 \right) .$$

The syndrome of $\boldsymbol{y}$ is given by

$$\begin{aligned} s &= \left(\boldsymbol{y} \cdot \boldsymbol{\alpha}^{\mathsf{T}} \right) \text{ MOD } 31 \\ &= (3{+}5{+}6{+}2{\cdot}7{+}0{\cdot}9{+}2{\cdot}10{+}11{+}12 \\ &\qquad {+}2{\cdot}13{+}2{\cdot}14{+}1{+}2{+}2{\cdot}4{+}8{+}2{\cdot}16) \text{ MOD } 31 \\ &= 21 . \end{aligned}$$

Namely, $s = 31 - 21 = 10 = \alpha_5$, indicating that the error location is $j = 5$ (the sixth entry) and the error value is -1 (corresponding to changing the value 3 into 2). □

If one is interested in detecting two errors (instead of correcting a single error), the decoding mapping $\mathcal{D}_1$ in Figure 3 is simplified to returning “e” if and only if $s \neq 0$.

We end this subsection by demonstrating that a redundancy of $n - k = \lceil \log_q(2n+1) \rceil$ is within one symbol from the smallest possible for any coding scheme that corrects one error in the L_1-metric. Recall that by taking $\boldsymbol{u}$ as a unit vector it follows that for any row index i, the set of the q^k possible contents of A_i forms an (ordinary) code $\mathcal{C}_i \subseteq \Sigma_q^n$ that is capable of correcting one error. Hence, by a sphere-packing argument we conclude that for distinct $\boldsymbol{c} \in \mathcal{C}_i$, the (truncated) spheres $\mathcal{S}_{\mathcal{L}}(\boldsymbol{c}, 1) \cap \Sigma_q^n$ must be disjoint subsets of Σ_q^n. Yet $|\mathcal{S}_{\mathcal{L}}(\boldsymbol{c}, 1) \cap \Sigma_q^n| \geq n + 1$, and, so, $q^k = |\mathcal{C}_i| \leq q^n/(n+1)$, namely, we have the lower bound

$$n - k \geq \lceil \log_q(n+1) \rceil .$$

B. Allowing additional error detection

The presented coding scheme can be easily enhanced so that the induced code has minimum distance 4; namely, the scheme can detect two errors on top of correcting a single error, or, alternatively, it can detect three errors with no attempt to correct any error. We do this by extending the code length by 1 and adding a parity bit to each row of $A = \mathcal{E}_1(A')$, as we did in Example 1 (with A playing the role of A' therein). This, in turn, allows the decoder to recover the parity of the number of errors (whether it was even or odd). An odd number is seen as one error, and the algorithm in Figure 3 is then applied. Otherwise, a zero syndrome will indicate no errors, while a nonzero syndrome indicates two errors (which will be flagged by “e”).

When $q > 2$, this extra error detection capability can sometimes be achieved *without* increasing the redundancy. To see this, consider first the case where q is odd: redefine m to be $\lceil \log_q(4n{+}2) \rceil$ (depending on n, the value of m may remain unchanged by this redefinition), and modify condition (i)–(ii) as follows.

(i') The entries of $\boldsymbol{\alpha}$ are *odd* distinct elements in $[4n{+}2\rangle$.
(ii') For any two indexes $i, j \in [n\rangle$,

$$\alpha_i + \alpha_j \neq 4n + 2 .$$

(Note that condition (iii), which remains unchanged, is consistent with condition (i'). Also, condition (ii') disqualifies $2n+1$ to be an entry[8] of $\boldsymbol{\alpha}$.) The encoding is similar to (2), except that the remainder at the right-hand side is now computed modulo $4n+2$. Accordingly, during decoding, the syndrome is redefined to

$$s \leftarrow \left(\boldsymbol{y} \cdot \boldsymbol{\alpha}^{\mathsf{T}}\right) \text{ MOD } (4n+2) \; ,$$

and, so, the parity of the syndrome equals the parity of the number of errors. An odd syndrome indicates that one error has occurred, in which case the error location and sign can be recovered from the value of s. A nonzero even syndrome s indicates that two errors have occurred.

Assume now that q is an even integer greater than 2. In this case, condition (i') would contradict condition (iii), as the latter requires that the last $m-1$ entries of $\boldsymbol{\alpha}$ be even. To overcome this impediment, we will modify the definition of m and rewrite condition (iii). Specifically, we let m be the smallest integer that satisfies $f_m(q) \ge 4n+2+(-1)^m$ where, for every nonnegative $j \in \mathbb{Z}$,

$$f_j(q) = \frac{q^{j+1} + (-1)^j}{q+1} \; . \tag{4}$$

Note that $f_0(q) = 1$ and that for every $j > 0$,

$$f_j(q) = (q-1) \sum_{i \in [j\rangle} f_i(q) + \begin{cases} 1 & \text{if } j \text{ is even} \\ 0 & \text{otherwise} \end{cases} \; ,$$

which means that every integer in $[4n+2\rangle$ has a representation of the form[9] $\sum_{j \in [m\rangle} b_j f_j(q)$, for $b_j \in \Sigma_q$. Moreover, the values $f_j(q)$ are all odd. Hence, rewriting condition (iii) as follows will be consistent[10] with condition (i'):

(iii') $\alpha_{k+j} = f_j(q)$, for $j \in [m\rangle$.

From this point onward, we proceed as in the case of odd q. (We point out that since $f_m(q) < q^m$ for $m > 0$, the inequality $f_m(q) \ge 4n+2+(-1)^m$ is generally stronger than $m \ge \log_q(4n+2)$; however, the ratio $f_m(q)/q^m$ does approach 1 as $q \to \infty$.)

Example 3. Suppose that $q = 8$ and $n = 13$. Since $f_1(8) = 7$ and $f_2(8) = 57$, we can take $m = 2$ and

$$\boldsymbol{\alpha} = \left(3\ 5\ 9\ 11\ 13\ 15\ 17\ 19\ 21\ 23\ 25 \;\middle|\; 1\ 7\right) \; ,$$

resulting in a single-error-correcting double-error-detecting coding scheme. The redundancy $n-k$ will be only 2 in this case (as opposed to 3 had we added a parity bit to the construction in Subsection III-A for $n = 13$). □

IV. Larger minimum L_1-distances

In this section, we show how to extend the construction of Section III to correct more errors in the L_1-metric. Our coding schemes will make use of known construction for the Lee metric, specifically Berlekamp codes, to be recalled in the next subsection.

[8] Yet the coding scheme will work also when $(\alpha_{n-1} =)\ q^{m-1} = 2n+1$, in spite of violating condition (ii').

[9] The sequence $(f_j(q))_j$ can be seen as a generalization of the Jacobsthal sequence: see for instance [10].

[10] The coding scheme will work also when $(\alpha_{n-1} =)\ f_{m-1}(q) = 2n+1$, in spite of violating condition (ii').

A. Lee-metric codes

Let p be an odd prime and let $F = \mathrm{GF}(p)$. Representing the elements of F as $0, 1, 2, \ldots, p-1$, the last $(p-1)/2$ elements in this list will be referred to as the "negative" elements in F. The *Lee metric* over F is defined similarly to the L_1-metric over $\mathbb{Z}$, using the following definition of the absolute value (in $\mathbb{Z}$) of an element $z \in F$:

$$|z| = \begin{cases} z & \text{if } z \text{ is "nonnegative"} \\ p - z & \text{otherwise} \end{cases} \; .$$

Let n and τ be positive integers such that $2\tau < p$, and let $h = \lceil \log_p(2n+1) \rceil$. Also, let

$$\boldsymbol{\beta} = (\beta_0\ \beta_1\ \ldots\ \beta_{n-1})$$

be a vector of length n over the extension field $\Phi = \mathrm{GF}(p^h)$ whose entries are nonzero and distinct and satisfy $\beta_i + \beta_j \ne 0$ for every $i, j \in [n\rangle$ (compare with conditions (i)–(ii) in Section III; it is easy to see that here, too, such a vector $\boldsymbol{\beta}$ always exists). The respective Berlekamp code, $\mathsf{C}_{\mathrm{Ber}} = \mathsf{C}_{\mathrm{Ber}}(\boldsymbol{\beta}, \tau)$, is defined as the set of all row vectors in F^n in the right kernel of the following $\tau \times n$ parity-check matrix, $H_{\mathrm{Ber}} = H_{\mathrm{Ber}}(\boldsymbol{\beta}, \tau)$, over Φ:

$$H_{\mathrm{Ber}} = \begin{pmatrix} \beta_0 & \beta_1 & \ldots & \beta_{n-1} \\ \beta_0^3 & \beta_1^3 & \ldots & \beta_{n-1}^3 \\ \beta_0^5 & \beta_1^5 & \ldots & \beta_{n-1}^5 \\ \vdots & \vdots & \vdots & \vdots \\ \beta_0^{2\tau-1} & \beta_1^{2\tau-1} & \ldots & \beta_{n-1}^{2\tau-1} \end{pmatrix} . \tag{5}$$

That is,

$$\mathsf{C}_{\mathrm{Ber}} = \left\{ \boldsymbol{c} \in F^n \ : \ \boldsymbol{c} \cdot H_{\mathrm{Ber}}^{\mathsf{T}} = \boldsymbol{0} \right\} \; .$$

Thus, $\mathsf{C}_{\mathrm{Ber}}$ is a linear $[n,k]$ code over F with a redundancy $n-k$ of at most $\tau h = \tau \lceil \log_p(2n+1) \rceil$.

The minimum Lee distance of $\mathsf{C}_{\mathrm{Ber}}$ is known to be at least $2\tau+1$, and there are known efficient algorithms for decoding up to τ Lee-metric errors (see [2, Ch. 9] and [22, §10.6]). These decoders typically start with computing the syndrome, $\boldsymbol{s} = \boldsymbol{y} H_{\mathrm{Ber}}^{\mathsf{T}}$, of the received vector $\boldsymbol{y} \in F^n$, and then implement a function $\mathsf{D}_{\mathrm{Ber}} : \Phi^\tau \to F^n$ which maps $\boldsymbol{s}$ to the error vector $\boldsymbol{e} = \mathsf{D}_{\mathrm{Ber}}(\boldsymbol{s})$. The condition $2\tau < p$ implies that (1) is also the volume of a Lee-metric sphere of radius t in F^n. Hence, by sphere-packing arguments, the size of any Lee-metric τ-error-correcting code in F^n is bounded from above by $p^n / V(n, \tau)$ [22, p. 318]. Thus, up to an additive term that depends on τ (but not on n), the dimension of $\mathsf{C}_{\mathrm{Ber}}$ is the largest possible, for a given length n and number τ of Lee-metric errors to be corrected.

There is a close relationship between the construction presented in Section III and Berlekamp codes. Specifically, when n is taken so that $p = 2n+1$ is a prime, then each row of $A = \mathcal{E}_1(A')$ is a codeword of $\mathsf{C}_{\mathrm{Ber}}(\boldsymbol{\alpha}, 1)$, assuming that the entries of A and $\boldsymbol{\alpha}$ are seen as elements of $\Phi = F = \mathrm{GF}(2n+1)$. Consequently, the induced code $\mathcal{C}$ forms a subset of $\mathsf{C}_{\mathrm{Ber}}(\boldsymbol{\alpha}, 1)$.

With this relationship in mind, we will next present coding schemes whose induced codes have minimum L_1-distances 5 and above.

- $\mathcal{E}_1$ is the encoding mapping in Figure 2, with n therein replaced by n_1 (in particular, the remainder in the right-hand side of (2) is computed modulo p).
- $\varphi_2 : \Sigma_q^{\ell\times n_1} \to \Sigma_q^{\ell\times n_2}$ maps an $\ell \times n_1$ matrix $A' = (a_{i,j})_{i\in[\ell\rangle, j\in[n_1\rangle}$ over Σ_q to $A = (A' \mid A'')$, where the last m columns in A are set so that
$$\sum_{j\in[m\rangle} a_{i,n_1+j} q^j = \Big(\sum_{j\in[n_1\rangle} a_{i,j}\alpha_j^3\Big) \text{ MOD } p \;, \quad i \in [\ell\rangle \;. \tag{6}$$
- $\hat{\varphi}_2 : \Sigma_q^{\ell\times n_2} \to \Sigma_q^{\ell\times n}$ is the parity mapping defined on the last m columns of the argument matrix; that is, an $\ell \times n_2$ matrix $A' = (a_{i,j})_{i\in[\ell\rangle, j\in[n_2\rangle}$ over Σ_q is mapped to $A = (A' \mid A'')$, where the entries of the last column in A are given by
$$a_{i,n_2} = \Big(\sum_{j\in[m\rangle} a_{i,n_1+j}\Big) \text{ MOD } 2 \;, \quad i \in [\ell\rangle \;.$$

Fig. 4. Component functions of a double-error-correcting encoding mapping $\mathcal{E}_2 = \hat{\varphi}_2 \circ \varphi_2 \circ \mathcal{E}_1$.

B. Double-error-correcting coding scheme

In this subsection, we present a DPE coding scheme for correcting two errors in the L_1-metric (namely, the induced code will have minimum L_1-distance at least 5). This scheme will then be extended to also detect three errors (minimum distance 6).

Given the alphabet Σ_q and the number of rows ℓ, let $p > 3$ be a prime, and define $n_1 = (p-1)/2$, $m = \lceil \log_q p \rceil$, $n_2 = n_1 + m$, and $n = n_2 + 1$. Our coding scheme will have dimension $k = n_1 - m$, length[11] n and, therefore, redundancy $n - k = 2m + 1$. The encoding mapping, $\mathcal{E}_2 : \Sigma_q^{k\times\ell} \to \Sigma_Q^{n\times\ell}$, takes the form of a composition
$$\mathcal{E}_2 = \hat{\varphi}_2 \circ \varphi_2 \circ \mathcal{E}_1 \;,$$
where the component functions are defined in Figure 4.

Example 4. Let $q = 2$ and $p = 31$, in which case $n_1 = 15$, $m = 5$, $n_2 = 20$, $n = 21$, and $k = 10$. Select $\ell = 3$ and let $\boldsymbol{\alpha}$ and A' be as in Example 2. For $i = 0, 1, 2$, the values at the right-hand side of (6) are
$$\begin{aligned}
(3^3{+}6^3{+}7^3{+}10^3{+}13^3{+}1^3{+}2^3{+}4^3{+}16^3) \text{ MOD } 31 &= 16\\
(7^3+10^3+11^3+14^3+4^3+16^3) \text{ MOD } 31 &= 30\\
(5^3+10^3+12^3+13^3+14^3+8^3) \text{ MOD } 31 &= 29 \;,
\end{aligned}$$
and, so,
$$\begin{aligned}
\mathcal{E}_2(A') &= \hat{\varphi}_2(\varphi_2(\mathcal{E}_1(A')))\\
&= \left(\begin{array}{c|c|c|c}
1011010010 & 11101 & 00001 & 1\\
0001011001 & 00101 & 01111 & 0\\
0100010111 & 00010 & 10111 & 0
\end{array}\right) .
\end{aligned}$$
□

[11]The seeming restriction on n imposed by requiring that $2n_1 + 1$ is a prime can be lifted by code shortening.

It follows from the definition of $\mathcal{E}_2$ that every codeword $\boldsymbol{c} = \boldsymbol{u}A$ in the code $\mathcal{C}_2$ induced by $\mathcal{E}_2$ satisfies the following congruences:
$$\sum_{j\in[n_1\rangle} c_j\alpha_j \equiv 0 \pmod p \tag{7}$$
$$\sum_{j\in[n_1\rangle} c_j\alpha_j^3 \equiv \sum_{j\in[m\rangle} c_{n_1+j}q^j \pmod p \tag{8}$$
$$\sum_{j\in[m+1\rangle} c_{n_1+j} \equiv 0 \pmod 2 \;. \tag{9}$$

Proposition 3. *The induced code $\mathcal{C}_2$ satisfies*
$$\mathsf{d}_{\mathcal{L}}(\mathcal{C}_2) \geq 5 \;.$$

Proof. Let $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e}$ be the read vector at the DPE output, where $\boldsymbol{c} \in \mathcal{C}$ and $\|\boldsymbol{e}\| \leq 2$. Write $\boldsymbol{e} = (\boldsymbol{e}_1 \mid \boldsymbol{e}_2)$ and $\boldsymbol{y} = (\boldsymbol{y}_1 \mid \boldsymbol{y}_2)$, where $\boldsymbol{e}_1$ (respectively, $\boldsymbol{y}_1$) is the n_1-prefix of $\boldsymbol{e}$ (respectively, $\boldsymbol{y}$). We associate with $\boldsymbol{y}$ the integer syndrome $(s_1\; s_2\; \hat{s}_2)$ computed as in Eqs. (10)–(12) in Figure 5.

We distinguish between the following cases.

Case 1: $s_1 = 0$. In this case, $\boldsymbol{e}_1 = \boldsymbol{0}$ (i.e., $\boldsymbol{y}_1$ is error-free), since $\|\boldsymbol{e}_1\| \in \{1, 2\}$ implies $s_1 \neq 0$.

Case 2: $s_1 \neq 0$ *and* $\hat{s}_2 = 0$. In this case, $\|\boldsymbol{e}_1\| \in \{1, 2\}$, and (by (9)), $\boldsymbol{y}_2$ contains an even number of errors, which means that $\boldsymbol{e}_2 = \boldsymbol{0}$. Therefore,
$$\begin{aligned}
s_2 &\equiv \sum_{j\in[n_1\rangle} y_j\alpha_j^3 - \sum_{j\in[m\rangle} y_{n_1+j}q^j\\
&\equiv \sum_{j\in[n_1\rangle} y_j\alpha_j^3 - \sum_{j\in[m\rangle} c_{n_1+j}q^j\\
&\equiv \sum_{j\in[n_1\rangle} (c_j + e_j)\alpha_j^3 - \sum_{j\in[m\rangle} c_{n_1+j}q^j\\
&\overset{(8)}{\equiv} \sum_{j\in[n_1\rangle} e_j\alpha_j^3 \pmod p \;.
\end{aligned}$$
On the other hand, from (7) we also have
$$s_1 \equiv \sum_{j\in[n_1\rangle} e_j\alpha_j \pmod p \;.$$
Hence, we can recover $\boldsymbol{e}_1$ by applying a decoder $\mathsf{D}_{\mathrm{Ber}}$ for $\mathsf{C}_{\mathrm{Ber}}(\boldsymbol{\alpha}, 2)$ to the syndrome[12] $(s_1\; s_2)$.

Case 3: $s_1 \neq 0$ *and* $\hat{s}_2 = 1$. This is possible only if $\|\boldsymbol{e}_1\| = \|\boldsymbol{e}_2\| = 1$, which means that we are able to recover $\boldsymbol{e}_1$ from s_1 (using the decoding mapping in Figure 3). □

Note that in case 1 in the last proof, $\boldsymbol{y}_2$ may contain one or two errors, yet we do not attempt to decode them; in fact, their decoding might not be unique. However, $\boldsymbol{y}_1$ still decodes correctly.

The proof of Proposition 3 implies the decoding mapping $\mathcal{D}_2$ shown in Figure 5.

We include in Appendix B an example of an application of the decoder in Figure 5; for self-containment, that example also recalls the decoding principles of Berlekamp codes.

[12]When applying this decoder, we regard $\boldsymbol{\alpha}$ and $(s_1\; s_2)$ as vectors over $\Phi = F = \mathrm{GF}(p)$. The decoder $\mathsf{D}_{\mathrm{Ber}}$ produces an error vector in F^{n_1}, which is mapped back to $\mathbb{Z}^{n_1}$ by changing each given entry $z \in F$ into $\pm|z|$, with the negative sign taken when z is a “negative” element of F.

```
Input: $\boldsymbol{y} = (\boldsymbol{y}_1 \mid \boldsymbol{y}_2) \in \Sigma_q^n$.
Output: $\boldsymbol{w} \in \Sigma_q^k$.
// $n_1 = (p-1)/2$, $m = \lceil \log_q p \rceil$, $n_2 = n_1 + m$.
// $n = n_2 + 1$, $k = n_1 - m$.
// $\boldsymbol{\alpha}$ satisfies conditions (i)–(iii).
// $\boldsymbol{y}_1$ is the $n_1$-prefix of $\boldsymbol{y}$.
// $\mathsf{D}_{\mathrm{Ber}}(\cdot)$ is a decoder for $\mathsf{C}_{\mathrm{Ber}}(\boldsymbol{\alpha}, 2)$.
Let
    $s_1 \leftarrow \Big( \sum_{j \in [n_1\rangle} y_j \alpha_j \Big) \text{ MOD } p$      (10)
    $s_2 \leftarrow \Big( \sum_{j \in [n_1\rangle} y_j \alpha_j^3 - \sum_{j \in [m\rangle} y_{n_1+j} q^j \Big) \text{ MOD } p$      (11)
    $\hat{s}_2 \leftarrow \Big( \sum_{j \in [m+1\rangle} y_{n_1+j} \Big) \text{ MOD } 2$ ;      (12)
If $s_1 = 0$ then {    // $\boldsymbol{y}_1$ is error-free
    Let $\boldsymbol{w} \leftarrow \boldsymbol{y}'$;    // $\boldsymbol{y}'$ is the $k$-prefix of $\boldsymbol{y}$
}
Else if $\hat{s}_2 = 0$ then {
    Let $\boldsymbol{e}_1 \leftarrow \mathsf{D}_{\mathrm{Ber}}(s_1 \; s_2)$;
    Let $\boldsymbol{w} \leftarrow (\boldsymbol{y}_1 - \boldsymbol{e}_1)'$;    // $\boldsymbol{w}$ is the $k$-prefix of $\boldsymbol{y}_1 - \boldsymbol{e}_1$
}
Else {
    Let $\boldsymbol{w} \leftarrow \mathcal{D}_1(\boldsymbol{y}_1)$.
}
```

Fig. 5. Decoding mapping $\mathcal{D}_2 : \boldsymbol{y} \mapsto \boldsymbol{w}$ for double-error correction.

The coding scheme $(\mathcal{E}_2, \mathcal{D}_2)$ can be extended to also detect three errors, by adding an overall parity bit to each row of the encoded matrix A, as was done in Subsection III-B. Moreover, the savings shown there when $q > 2$ carries over also to minimum distances 5 and 6.

Specifically, for odd q, we redefine m to be $\lceil \log_q(2p) \rceil = \lceil \log_q(4n_1+2) \rceil$, and require[13] $\boldsymbol{\alpha}$ to satisfy conditions (i')–(ii'). The encoding mapping $\mathcal{E}_2$ is redefined to just $\varphi_2 \circ \mathcal{E}_1$, with code length $n = n_2 = n_1 + m$ and redundancy $n - k = 2m$. The component functions $\mathcal{E}_1$ and φ_2 are as in Figure 4, except that all remainders modulo p are now computed modulo $2p$.

The function of the syndrome element $\hat{s}_2$ in the decoding process is replaced by the parities of s_1 and s_2, when computed as in (10) and (11), except that the remainders are taken modulo $2p$ (yet $\mathsf{C}_{\mathrm{Ber}}$ is still defined over $F = \mathrm{GF}(p)$, so when its decoder is applied to $(s_1 \; s_2)$, the syndrome entries are reduced first modulo p). Specifically, assuming that at most three errors have occurred, Table I presents the various combinations of parities of s_1 and s_2, and the corresponding L_1-norms of $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$. The first three rows in the table correspond, respectively, to the three cases in the proof of Proposition 3. The fourth row corresponds to three errors and therefore should result in a decoding failure. The last three rows correspond to three different combinations of number of errors: the distinction among them can be made by checking whether $s_2 \equiv s_1^3 \pmod{p}$, and, since $\mathsf{C}_{\mathrm{Ber}}(\boldsymbol{\alpha}, 2)$ has minimum Lee distance 5, this condition will be met only when one error has occurred (in which case it can be found using the decoding mapping in Figure 3).

TABLE I
DECODING TWO ERRORS AND DETECTING THREE ERRORS.

s_1	s_2	$s_2 \equiv s_1^3$?	$\lVert \boldsymbol{e}_1 \rVert$	$\lVert \boldsymbol{e}_2 \rVert$	Decoder output
0	–	–	0	–	$\boldsymbol{y}_1'$
even $\neq 0$	even	–	2	0	$(\boldsymbol{y}_1 - \mathsf{D}_{\mathrm{Ber}}(s_1 \; s_2))'$
odd	even	–	1	1	$\mathcal{D}_1(\boldsymbol{y}_1)$
even $\neq 0$	odd	–	2	1	"e"
odd	odd	yes	1	0	$\mathcal{D}_1(\boldsymbol{y}_1)$
odd	odd	no	3	0	"e"
odd	odd	no	1	2	"e"

When q is even and greater than 2, we will follow the same strategy as in Section III, namely, replacing the terms[14] q^j with $f_j(q)$ defined in (4), both in condition (iii) and in (6).

Example 5. Suppose that $q = 4$ and $p = 101$, corresponding to $n_1 = 50$. The values $f_j(4)$ for $j = 0, 1, 2, 3, 4$ are 1, 3, 13, 51, and 205, respectively, so we can take $m = 4$ and

$$\boldsymbol{\alpha} = \big(5\;7\;9\;11\;15\;17\;\ldots\;47\;49\;53\;55\;\ldots\;97\;99 \;\big|\; 1\;3\;13\;51 \big) .$$

The respective double-error-correcting triple-error-detecting coding scheme has length $n = n_1 + m = 54$ and redundancy $n - k = 2m = 8$ (and dimension $k = 46$). An example of an image of the encoding mapping $\mathcal{E}_2 = \varphi_2 \circ \mathcal{E}_1$ is given by

$$A = \mathcal{E}_2(A') = \left(\begin{array}{c|c|c} 1\,2\,3\,0\,1\,2\,0\,0 \ldots 0 & 2\,1\,0\,2 & 0\,1\,3\,2 \\ 0\,3\,0\,1\,2\,3\,0\,0 \ldots 0 & 3\,3\,2\,1 & 1\,2\,2\,3 \\ 2\,1\,1\,3\,2\,0\,0\,0 \ldots 0 & 2\,3\,0\,2 & 2\,0\,0\,2 \end{array} \right) ,$$

with

$$\boldsymbol{c} = \boldsymbol{u}A = \big(4\;14\;7\;6\;10\;13\;0\;0\;\ldots\;0 \;\big|\; 15\;14\;6\;9 \;\big|\; 5\;8\;12\;15 \big)$$

being an example of a codeword (which corresponds to the input vector $\boldsymbol{u} = (2\;3\;1)$).

Given a read vector $\boldsymbol{y} = (y_j)_{j \in [n\rangle} \in \Sigma_Q^n$ (where $Q = 28$ and $n = 54$), its syndrome is given by

$$s_1 = \Big(\sum_{j \in [n_1\rangle} y_j \alpha_j \Big) \text{ MOD } (2p)$$

$$s_2 = \Big(\sum_{j \in [n_1\rangle} y_j \alpha_j^3 - \sum_{j \in [m\rangle} y_{n_1+j} f_j(q) \Big) \text{ MOD } (2p)$$

(where $q = 4$, $p = 101$, $n_1 = 50$, and $m = 4$). Correction of two errors and detecting of three then proceeds by following Table I, where $\boldsymbol{y}_1$ and $\boldsymbol{y}_1'$ are the prefixes of $\boldsymbol{y}$ of lengths $n_1 = 50$ and $k = 46$, respectively. □

C. Recursive coding scheme

The construction in Subsection IV-B does not seem to generalize in a straightforward way to larger minimum L_1-distances. However, with some redundancy increase (which will be relatively mild for code lengths sufficiently large), we can construct coding schemes for any prescribed number of correctable errors. We show this next.

[13] See Footnote 8 for the case where $q = p$.

[14] See Footnote 10 for the case where $f_{m-1}(q) = p$.

Given the alphabet Σ_q, number of rows ℓ, and the designed number of correctable errors[15] τ, let $p > 2\tau$ be a prime, and define $n = (p-1)/2$ and $m = \lceil \log_q p \rceil$. Also, let $\boldsymbol{\alpha} = (\alpha_j)_{j \in [n\rangle}$ be an integer vector that satisfies conditions (i)–(iii) in Section III.

Given a matrix $A \in \Sigma_q^{\ell \times n}$ (which, at this point, is not assumed to be the result of any encoding), we can compute the following $\ell \times \tau$ *syndrome matrix* of A over $\mathbb{Z}$:

$$S = (s_{i,v})_{i \in [\ell\rangle, v \in [\tau\rangle} = A H_{\mathrm{Ber}}^{\mathsf{T}} \ \mathrm{MOD}\ p \ ,$$

where $H_{\mathrm{Ber}} = H_{\mathrm{Ber}}(\boldsymbol{\alpha}, \tau)$ is the parity-check matrix defined in (5), now seen as a matrix over $\mathbb{Z}$, and the remainder computed entry-wise. For a vector $\boldsymbol{u} \in \Sigma_q^{\ell}$, the syndrome $\boldsymbol{s} = \boldsymbol{s}(\boldsymbol{u})$ of $\boldsymbol{c} = \boldsymbol{c}(\boldsymbol{u}) = \boldsymbol{u}A$ is then given by

$$\begin{aligned} \boldsymbol{s} &= \boldsymbol{c} H_{\mathrm{Ber}}^{\mathsf{T}} \ \mathrm{MOD}\ p \\ &= \boldsymbol{u} A H_{\mathrm{Ber}}^{\mathsf{T}} \ \mathrm{MOD}\ p \\ &= \boldsymbol{u} S \ \mathrm{MOD}\ p \ . \end{aligned}$$

If the syndrome $\boldsymbol{s}$ is available to the decoder, then the decoder should be able to recover $\boldsymbol{c} = \boldsymbol{u}A$ from an erroneous copy $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e}$ ($\in \mathbb{Z}^n$), provided that $\|\boldsymbol{e}\| \le \tau$: this is simply because the syndrome $\hat{\boldsymbol{s}}$ of $\boldsymbol{e}$ is computable from $\boldsymbol{s}$ and the syndrome of $\boldsymbol{y}$,

$$\begin{aligned} \hat{\boldsymbol{s}} = \boldsymbol{e} H_{\mathrm{Ber}}^{\mathsf{T}} \ \mathrm{MOD}\ p &= (\boldsymbol{y} - \boldsymbol{c}) H_{\mathrm{Ber}}^{\mathsf{T}} \ \mathrm{MOD}\ p \\ &= (\boldsymbol{y} H_{\mathrm{Ber}}^{\mathsf{T}} - \boldsymbol{s}) \ \mathrm{MOD}\ p \ , \end{aligned}$$

and $\boldsymbol{e} \leftarrow \mathsf{D}_{\mathrm{Ber}}(\hat{\boldsymbol{s}})$, where $\mathsf{D}_{\mathrm{Ber}}(\cdot)$ is a decoder for $\mathsf{C}_{\mathrm{Ber}}(\boldsymbol{\alpha}, \tau)$. Thus, our encoding mapping will be designed so that, *inter alia*, the decoder is able to reconstruct a copy of $\boldsymbol{s}$.

Each entry in S, being an integer in $[p\rangle$, can be expanded to its base-q representation

$$s_{i,v} = \sum_{j \in [m\rangle} s_{i,v}^{(j)} q^j \ ,$$

where $s_{i,v}^{(j)} \in \Sigma_q$. For $j \in [m\rangle$, let $S^{(j)}$ be the $\ell \times \tau$ matrix $(s_{i,v}^{(j)})_{i \in [\ell\rangle, v \in [\tau\rangle}$ over Σ_q. Clearly, and, so,

$$\begin{aligned} \boldsymbol{s} &= \boldsymbol{u} S \ \mathrm{MOD}\ p \\ &= \Big(\sum_{j \in [m\rangle} q^j (\boldsymbol{u} S^{(j)}) \Big) \ \mathrm{MOD}\ p \\ &= \Big(\sum_{j \in [m\rangle} q^j \boldsymbol{s}^{(j)} \Big) \ \mathrm{MOD}\ p \ , \end{aligned}$$

where $\boldsymbol{s}^{(j)} = \boldsymbol{s}^{(j)}(\boldsymbol{u}) = \boldsymbol{u} S^{(j)}$ is a vector in Σ_Q^{τ}. Consider an encoding mapping $\mathcal{E} : \Sigma_q^{\ell \times n} \to \Sigma_q^{\ell \times (n + \tau m)}$ defined by

$$\mathcal{E} : A \mapsto \Big(A \mid S^{(0)} \mid S^{(1)} \mid \cdots \mid S^{(m-1)} \Big) \ .$$

Then, for $\boldsymbol{u} \in \Sigma_q^{\ell}$ we have

$$\boldsymbol{u}\, \mathcal{E}(A) = \Big(\boldsymbol{c} \mid \boldsymbol{s}^{(0)} \mid \boldsymbol{s}^{(1)} \mid \cdots \mid \boldsymbol{s}^{(m-1)} \Big)$$

(where $\boldsymbol{c} = \boldsymbol{c}(\boldsymbol{u})$ and $\boldsymbol{s}^{(j)} = \boldsymbol{s}^{(j)}(\boldsymbol{u})$). If $\boldsymbol{y} = \boldsymbol{u}\, \mathcal{E}(A) + \boldsymbol{e}$ where $\|\boldsymbol{e}\| \le \tau$, then, based on our previous discussion, we will be able to recover $\boldsymbol{c}$, *as long as the τm-suffix of $\boldsymbol{y}$ is error-free.*

The latter assumption (of an error-free suffix) can be guaranteed by applying a (second) encoding mapping to the $\ell \times \tau m$ matrix

$$\Big(S^{(0)} \mid S^{(1)} \mid \cdots \mid S^{(m-1)} \Big)$$

(over Σ_q) so that τ errors can be corrected. Note that the matrix now has $\tilde{n} = \tau m$ columns (instead of n), so we can base our encoding on a Berlekamp code over $\mathrm{GF}(\tilde{p})$, where $\tilde{p}$ is the smallest prime which is at least $2\tilde{n} + 1$. The size of the syndrome now will be $\tau \tilde{m}$, where $\tilde{m} = \lceil \log_q \tilde{p} \rceil$, namely, becoming *doubly-logarithmic* in n.

We can continue this process recursively; by just applying one more recursion level with a simple repetition encoding mapping (which copies its input $2\tau + 1$ times at the output), we obtain a total redundancy of

$$\begin{aligned} \tau m + (2\tau + 1)\tau \tilde{m} = \tau \lceil \log_q(2n+1) \rceil & \\ + O\left(\tau^2 \log_q(\tau \log_q n) \right) & \ . \qquad (13) \end{aligned}$$

Hence, for n large compared to τ, most of the redundancy is due to the first encoding level. In fact, by extending the sphere-packing argument presented at the end of Subsection III-A it follows that the redundancy (13) is optimal, up to the $O(\cdot)$ term.

Decoding is carried out backwards, starting with recovering the codeword that corresponds to the last encoding level, which, in turn, serves as the syndrome of the previous encoding level.

Reflecting now back on our constructions in Sections III and IV-B, if the matrix A ($\in \Sigma_q^{\ell \times n}$) is the output of the encoding scheme in Figure 2, then the first column of the syndrome matrix S is zero, and therefore so is the first column in each matrix $S^{(j)}$ (and the first entry in each vector $\boldsymbol{s}^{(j)}$). Hence, those zero columns can of course be removed. As for the second column, our construction in Subsection IV-B implies that it can be error-protected simply by a parity bit (or, when $q > 2$, by changing the modulus from p to $2p$ and selecting the entries of $\boldsymbol{\alpha}$ to be odd).

The approach of recursive encoding is not new, and has been used, for example, in the context of constrained coding (e.g., see [3], [7], [15], [17], [21]). In our setting, this approach allows us to use codes (namely, Berlekamp codes), which are originally defined over one alphabet of size p, while the result of the encoding (namely, the contents of the rows of the DPE matrix) are restricted to belong to another alphabet of size q (the challenge is evident when $q < p$). In the next subsection, we consider a more straightforward application of Berlekamp codes to construct a coding scheme for the case where q is large enough; this scheme may sometimes have a smaller redundancy than (13).

D. Coding scheme for large alphabets

We consider here the case where the number of correctable errors τ and the alphabet size q are such that there exists a prime p that satisfies

$$2\tau < p \le q \ .$$

[15]For simplicity, we assume that $\sigma = 0$, namely, no additional errors are to be detected.

```
Input: $\ell \times k$ matrix $A' = (a_{i,j})_{i\in[\ell\rangle, j\in[k\rangle}$ over $\Sigma_q$.
Output: $\ell \times n$ matrix $(A' \mid A'') = (a_{i,j})_{i\in[\ell\rangle, j\in[n\rangle}$ over $\Sigma_q$.
// $F = \mathrm{GF}(p)$, for a prime $p$ such that $2\tau+1 \le p \le q$.
// $\mathsf{C}_{\mathrm{Ber}}(\boldsymbol{\beta},\tau)$ is a Berlekamp code of length $n$ over $F$.
// $\mathsf{E}_{\mathrm{Ber}} : F^k \to \mathsf{C}_{\mathrm{Ber}}(\boldsymbol{\beta},\tau)$ is a systematic encoder.
For all $i \in [\ell\rangle$ do {
   Let $\boldsymbol{c} \leftarrow \mathsf{E}_{\mathrm{Ber}}(A'_i \text{ MOD } p)$;
   Let $A''_i \leftarrow \boldsymbol{c}''$.    // $F$ seen as a subset of $\Sigma_q$
}
```

Fig. 6. Encoding mapping $\mathcal{E} : A' \mapsto (A' \mid A'')$ for large alphabets.

We will then assume that p is the largest prime that does not exceed q, and we let F be the finite field $\mathrm{GF}(p)$.

We will use a systematic encoder $\mathsf{E}_{\mathrm{Ber}} : F^k \to \mathsf{C}_{\mathrm{Ber}}$, where $\mathsf{C}_{\mathrm{Ber}} = \mathsf{C}_{\mathrm{Ber}}(\boldsymbol{\beta},\tau)$ is a Berlekamp code of a prescribed length n over F and redundancy

$$\begin{aligned} n-k &\le \tau \cdot \lceil \log_p(2n+1) \rceil \\ &= \tau \cdot \lceil (\log_p q)\cdot \log_q(2n+1) \rceil \end{aligned}$$

(when n is sufficiently large compared to τ, the inequality is known to hold with equality). When $q = p$, this redundancy is smaller than (13); otherwise (when $q > p$), it will be larger for τ (much) smaller than n, due to the factor $\log_p q$ (e.g., for $q = 8$, this factor is approximately 1.07).

Our encoding mapping $\mathcal{E} : \Sigma_q^{\ell\times k} \to \Sigma_q^{\ell\times n}$ takes each row in the pre-image matrix $A' \in \Sigma_q^{\ell\times k}$, computes the remainder of each entry modulo p, regards the result as a vector in F^k, and applies to it the encoder $\mathsf{E}_{\mathrm{Ber}}$ to produce a codeword $\boldsymbol{c} \in \mathsf{C}_{\mathrm{Ber}}$. The $(n{-}k)$-suffix, $\boldsymbol{c}''$, of $\boldsymbol{c}$ becomes the $(n{-}k)$-suffix of the respective row in the image $A = (A' \mid A'') = \mathcal{E}(A')$ (see Figure 6).

It follows from the construction that the codewords of the induced code $\mathcal{C}$, when reduced modulo p, are codewords of $\mathsf{C}_{\mathrm{Ber}}$. Thus, we obtain a coding scheme that can correct τ errors.

V. Coding scheme for the Hamming metric

In this section, we present a coding scheme that handles errors in the Hamming metric; namely, the number of errors equals the number of positions in which the read vector $\boldsymbol{y} = (y_j)_{j\in[n\rangle} \in \Sigma_Q^n$ differs from the correct computation $\boldsymbol{c} = (c_j)_{j\in[n\rangle} = \boldsymbol{u}A$.

For the purpose of the exposition, we will introduce yet another design parameter, ϑ, which will be an assumed upper bound on the absolute value of any error value, namely, on

$$\max_{j\in[n\rangle} |y_j - c_j| \; .$$

Such an error model may be of independent interest in DPE applications, with the special case $\vartheta = Q{-}1 = \ell(q{-}1)^2$ being equivalent to the ordinary Hamming metric.

Given the alphabet Σ_q, number of rows ℓ, number of columns n, number of correctable errors[16] τ, and an upper bound ϑ on the error absolute value (where $\vartheta = \ell(q{-}1)^2$ for unconstrained error values), let p be a prime greater than 2ϑ and let $m = \lceil \log_q p \rceil$. We select a respective linear τ-error-correcting $[\tilde{n}, k]$ code C over $F = \mathrm{GF}(p)$ (in the Hamming metric), which is assumed to have an efficient bounded-distance decoder $\mathsf{D} : F^{\tilde{n}} \to F^{\tilde{n}}$: for a received word $\tilde{\boldsymbol{y}} \in F^{\tilde{n}}$, the decoder returns the true error vector $\tilde{\boldsymbol{e}} \in F^{\tilde{n}}$, provided that its Hamming weight $\mathsf{w}(\tilde{\boldsymbol{e}})$ was at most τ.

[16]We assume that $\sigma = 0$ (as in Subsection IV-C) and that there are no erasures.

```
Input: $\ell \times k$ matrix $A' = (a_{i,j})_{i\in[\ell\rangle, j\in[k\rangle}$ over $\Sigma_q$.
Output: $\ell \times n$ matrix $(A' \mid A'') = (a_{i,j})_{i\in[\ell\rangle, j\in[n\rangle}$ over $\Sigma_q$.
// $F = \mathrm{GF}(p)$, for a prime $p > 2\vartheta$.
// $m = \lceil \log_q p \rceil$ and $n = k + m(\tilde{n} - k)$.
// $\mathsf{C}$ is a linear $\tau$-error-correcting $[\tilde{n}, k]$ code over $F$.
// $\mathsf{E} : F^k \to \mathsf{C}$ is a systematic encoder.
For all $i \in [\ell\rangle$ do {
   Let $\tilde{\boldsymbol{c}} = (\tilde{c}_v)_{v\in[\tilde{n}\rangle} \leftarrow \mathsf{E}(A'_i \text{ MOD } p)$;
   For each $v \in [\tilde{n}{-}k\rangle$ do {
      Set $((A^{(0)})_{i,v}\ (A^{(1)})_{i,v}\ \ldots\ (A^{(m-1)})_{i,v})$ to be
         the base-$q$ representation of $\tilde{c}_{k+v}$.
   }
}
Let $A'' \leftarrow \big(A^{(0)} \mid A^{(1)} \mid \cdots \mid A^{(m-1)}\big)$.
```

Fig. 7. Encoding mapping $\mathcal{E} : A' \mapsto (A' \mid A'')$ for the Hamming metric.

The parameters n and $\tilde{n}$ are related by

$$n = k + m(\tilde{n} - k) \; .$$

Figure 7 presents an encoding mapping $\mathcal{E} : A' \mapsto (A' \mid A'')$, where each row of A', when reduced modulo p, is first extended by the systematic encoder for C into a codeword $\tilde{\boldsymbol{c}}$ of C, and then the $\tilde{n} - k$ redundancy symbols (over F) in $\tilde{\boldsymbol{c}}$ are expanded to their base-q representations to form the respective row in A''. Specifically,

$$A'' = \Big(A^{(0)} \mid A^{(1)} \mid \cdots \mid A^{(m-1)}\Big) \; ,$$

where each block $A^{(j)}$ is an $\ell \times (\tilde{n}{-}k)$ sub-matrix over Σ_q, such that the rows of the $\ell \times \tilde{n}$ matrix

$$\tilde{A} = \Big(A' \;\Big|\; \textstyle\sum_{j\in[m\rangle} q^j A^{(j)}\Big) \text{ MOD } p$$

form codewords of C.

Let the mapping $\lambda : \mathbb{Z}^n \to F^{\tilde{n}}$ be defined as follows: for a vector $\boldsymbol{x} = (x_v)_{v\in[n\rangle} \in \mathbb{Z}^n$, the entries of the image $\lambda(\boldsymbol{x}) = \tilde{\boldsymbol{x}} = (\tilde{x}_v)_{v\in[\tilde{n}\rangle} \in F^{\tilde{n}}$ are given by

$$\tilde{x}_v = x_v \text{ MOD } p \; , \quad \text{for } v \in [k\rangle \; , \tag{14}$$

and

$$\tilde{x}_{k+v} = \Big(\sum_{j\in[m\rangle} x_{k+v+j(\tilde{n}-k)} q^j\Big) \text{ MOD } p \; , \quad \text{for } v \in [\tilde{n}{-}k\rangle \; . \tag{15}$$

It is easy to see that each row in $\tilde{A}$ is obtained by applying the mapping λ to the respective row in $A = (A' \mid A'')$. Moreover,

```
Input: $\boldsymbol{y} \in \Sigma_q^n$.
Output: $\boldsymbol{w} \in \Sigma_q^k$.
// Parameters are as defined in Figure 7.
// $\lambda : \mathbb{Z}^n \to F^{\tilde{n}}$ is defined by (14)–(15).
// $\mathsf{D} : F^{\tilde{n}} \to F^{\tilde{n}}$ is a decoder for $\mathsf{C}$.
Let $\tilde{\boldsymbol{e}} = (\tilde{e}_j)_{j \in \tilde{n}} \leftarrow \mathsf{D}(\lambda(\boldsymbol{y}))$;
Set $\boldsymbol{e}' = (e_j)_{j \in [k\rangle}$ to
    $e_j \leftarrow \begin{cases} |\tilde{e}_j| & \text{if } \tilde{e}_j \text{ is ``nonnegative'' in } F \\ -|\tilde{e}_j| & \text{otherwise} \end{cases}$;
Let $\boldsymbol{w} \leftarrow \boldsymbol{y}' - \boldsymbol{e}'$.
```

Fig. 8. Decoding mapping $\mathcal{D} : \boldsymbol{y} \mapsto \boldsymbol{w}$ for the Hamming metric.

λ is a homomorphism in that it preserves vector addition and scalar multiplication: for every $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{Z}^n$ and $b_1, b_2 \in \mathbb{Z}$,

$$\lambda(b_1 \boldsymbol{x}_1 + b_2 \boldsymbol{x}_2) = \bar{b}_1 \lambda(\boldsymbol{x}_1) + \bar{b}_2 \lambda(\boldsymbol{x}_2) ,$$

where $\bar{b}_i = b_i \text{ MOD } p$ (seen as elements of F). Consequently, for every $\boldsymbol{u} \in \Sigma_q^k$,

$$\lambda(\boldsymbol{u}A) = \boldsymbol{u}\tilde{A} \text{ MOD } p \in \mathsf{C} .$$

The properties of $\lambda(\cdot)$ immediately imply a decoding algorithm (shown in Figure 8). Given the read vector $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e}$, where $\mathsf{w}(\boldsymbol{e}) \le \tau$, an application of λ to $\boldsymbol{y}$ yields:

$$\lambda(\boldsymbol{y}) = \lambda(\boldsymbol{c}) + \lambda(\boldsymbol{e}) ,$$

where $\lambda(\boldsymbol{c}) \in \mathsf{C}$ and $\mathsf{w}(\lambda(\boldsymbol{e})) \le \mathsf{w}(\boldsymbol{e}) \le \tau$. Hence, a decoder for C, when applied to $\lambda(\boldsymbol{y})$, will recover $\lambda(\boldsymbol{e})$. By the definition of λ, the vectors $\boldsymbol{e}$ and $\lambda(\boldsymbol{e})$ coincide, modulo p, on their k-prefix; and since the values of the entries of $\boldsymbol{e}$ are all within $\pm\vartheta$, the k-prefix of $\lambda(\boldsymbol{e})$ uniquely determines the k-prefix of $\boldsymbol{e}$.

Finally, we specialize to the case where C is a (normalized and possibly shortened) BCH code. In this case,

$$\tilde{n} - k \le \left\lceil 1 + \frac{p-1}{p}(2\tau-1) \right\rceil \cdot \lceil \log_p \tilde{n} \rceil$$

(see [22, p. 260, Problem 8.13]), and, so, the redundancy of our coding scheme is bounded from above by

$$n - k \le \left\lceil 1 + \frac{p-1}{p}(2\tau-1) \right\rceil \cdot \underbrace{\lceil \log_p \tilde{n} \rceil \cdot \lceil \log_q p \rceil}_{\approx \log_q n} .$$

For reference, recall that for every row index $i \in [\ell\rangle$, the possible contents of A_i must form an (ordinary) code over Σ_q of minimum Hamming distance at least $2\tau + 1$ (assuming that $\vartheta \ge q-1$). For n sufficiently large compared to τ, BCH codes over $\mathrm{GF}(q)$ are the best codes currently known for all prime powers q except 4 and 8. Hence, we should expect the redundancy of the coding scheme to be no less than

$$\left\lceil 1 + \frac{q-1}{q}(2\tau-1) \right\rceil \cdot \lceil \log_q n \rceil .$$

Acknowledgment

The author would like to thank Dick Henze, Naveen Muralimanohar, and John Paul Strachan for introducing him to the problem, and for the many helpful discussions. He would also like to thank the reviewers for their comments and suggestions.

Appendix A
Proofs

Proof of Proposition 1. For any two codewords $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{C}$ and a vector $\boldsymbol{y} \in \Sigma_Q^n$, we have

$$\mathsf{d}(\boldsymbol{c}_1, \boldsymbol{c}_2) \le \mathsf{d}(\boldsymbol{y}, \boldsymbol{c}_1) + \mathsf{d}(\boldsymbol{y}, \boldsymbol{c}_2) .$$

Hence, the inequalities

$$\mathsf{d}(\boldsymbol{y}, \boldsymbol{c}_1) \le \tau \quad \text{and} \quad \mathsf{d}(\boldsymbol{y}, \boldsymbol{c}_2) \le \tau + \sigma$$

can hold simultaneously, only if $\mathsf{d}(\boldsymbol{c}_1, \boldsymbol{c}_2) \le 2\tau + \sigma < \mathsf{d}(\mathcal{C})$, namely, only when $\boldsymbol{c}_1' = \boldsymbol{c}_2'$. This, in turn, implies that the following decoding mapping is well-defined and satisfies the correction and detection conditions above: for every $\boldsymbol{y} \in \Sigma_Q^n$,

$$\mathcal{D}(\boldsymbol{y}) = \begin{cases} \boldsymbol{c}' & \text{if there is } \boldsymbol{c} \in \mathcal{C} \text{ such that } \mathsf{d}(\boldsymbol{y}, \boldsymbol{c}) \le \tau \\ \text{``e''} & \text{otherwise} \end{cases} .$$

□

The condition on τ and σ in Proposition 1 is also necessary. Indeed, suppose (to the contrary) that there existed a coding scheme $(\mathcal{E}, \mathcal{D})$ that could correct τ errors and detect $\tau + \sigma$ errors, where $2\tau + \sigma \ge \mathsf{d}(\mathcal{C})$ and (without loss of generality) $\tau \le \mathsf{d}(\mathcal{C})$. Let $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{C}$ be such that $\mathsf{d}(\boldsymbol{c}_1, \boldsymbol{c}_2) = d = \mathsf{d}(\mathcal{C})$ (in particular, $\boldsymbol{c}_1' \ne \boldsymbol{c}_2'$); then there is a chain $\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_d \in \Sigma_Q^n$ such that $\boldsymbol{y}_0 = \boldsymbol{c}_1$, $\boldsymbol{y}_d = \boldsymbol{c}_2$, and $\mathsf{d}(\boldsymbol{y}_i, \boldsymbol{y}_{i+1}) = 1$ for $i \in [d\rangle$. On the one hand, we have $\mathsf{d}(\boldsymbol{c}_1, \boldsymbol{y}_\tau) \le \tau$, implying that the decoding mapping $\mathcal{D}$ would need to satisfy $\mathcal{D}(\boldsymbol{y}_\tau) = \boldsymbol{c}_1'$; yet, on the other hand, we also have $\mathsf{d}(\boldsymbol{c}_2, \boldsymbol{y}_\tau) \le d - \tau \le \tau + \sigma$, and that implies that $\mathcal{D}(\boldsymbol{y}_\tau) \in \{\boldsymbol{c}_2', \text{``e''}\}$, thereby reaching a contradiction.

Proof of Proposition 2. Let $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ be codewords in $\mathcal{C}$ with distinct k-prefixes. Ignoring the coordinates that have been erased, these codewords will still differ on at least $\mathsf{d}_{\mathcal{H}}(\boldsymbol{c}_1, \boldsymbol{c}_2) - \rho$ coordinates. Next apply Proposition 1, with $\mathsf{d}(\mathcal{C})$ therein replaced by $\mathsf{d}_{\mathcal{H}}(\mathcal{C}) - \rho$. □

Appendix B
Example

We include here an example of an execution of the decoder in Figure 5.

Example 6. Continuing Example 4, for $\boldsymbol{u} = (1\ 1\ 1)$, we get

$$\boldsymbol{c} = \boldsymbol{u}A = \left(1\ 1\ 1\ 2\ 0\ 3\ 1\ 1\ 2\ 2 \,\middle|\, 1\ 1\ 2\ 1\ 2 \,\middle|\, 2\ 0\ 1\ 2\ 1 \,\middle|\, 2\right) .$$

Suppose that the read vector is

$$\boldsymbol{y} = \left(1\ 1\ 1\ 2\ 0\ 2\ 1\ 1\ 2\ 2 \,\middle|\, 1\ 1\ 2\ 2\ 2 \,\middle|\, 2\ 0\ 1\ 2\ 1 \,\middle|\, 2\right) .$$

The syndrome of $\boldsymbol{y}$ is computed by (10)–(12) to yield

$$(s_1\ s_2\ \hat{s}_2) = (29\ 8\ 0) .$$

A nonzero s_1 indicates that (one or two) errors have occurred in the n_1-prefix, $\boldsymbol{y}_1$, of $\boldsymbol{y}$, and a zero $\hat{s}_2$ then indicates that the remaining part of $\boldsymbol{y}$ is error-free. Hence, we are in the scenario of case 2 in the proof of Proposition 3, i.e., $(s_1 \; s_2) = \boldsymbol{e}_1 H_{\mathrm{Ber}}^{\mathsf{T}} \text{ MOD } p$ (where $p = 31$), which allows us to find $\boldsymbol{e}_1$ using a decoder for $\mathsf{C}_{\mathrm{Ber}}$. Next, we recall the principles of the decoding algorithm for $\mathsf{C}_{\mathrm{Ber}}$, by demonstrating them on our particular example.

We first observe that if $\|\boldsymbol{e}_1\| = 1$ then necessarily $s_2 \equiv s_1^3 \pmod{31}$. Since this congruence does not hold in our case, we deduce that that two errors have occurred, say at positions $i, j \in [n_1\rangle$. We have

$$s_1 \equiv e_i\alpha_i + e_j\alpha_j \pmod{p} \qquad (16)$$

$$s_2 \equiv e_i\alpha_i^3 + e_j\alpha_j^3 \pmod{p} , \qquad (17)$$

where $e_i, e_j \in \{-1, 1\}$. Squaring both sides of (16) yields

$$s_1^2 \equiv \alpha_i^2 + 2(e_i\alpha_i)(e_j\alpha_j) + \alpha_j^2 \pmod{p} ,$$

while dividing each side of (17) by the respective side in (16) yields

$$\frac{s_2}{s_1} \equiv \alpha_i^2 - (e_i\alpha_i)(e_j\alpha_j) + \alpha_j^2 \pmod{p} ,$$

where $1/s_1$ stands for the inverse of s_1 modulo p. Subtracting each side of the last congruence from the respective sides of the previous congruence leads to

$$s_1^2 - \frac{s_2}{s_1} \equiv 3(e_i\alpha_i)(e_j\alpha_j) \pmod{p} . \qquad (18)$$

It follows from (16) and (18) that $e_i\alpha_i$ and $e_j\alpha_j$ are solutions of the following quadratic equation (in $F = \mathrm{GF}(p)$):

$$x^2 - s_1 x + \frac{1}{3}\left(s_1^2 - \frac{s_2}{s_1}\right) \equiv 0 \pmod{p} .$$

Specifically, in our case,

$$\frac{1}{s_1} = \frac{1}{29} \equiv 15 \pmod{31} \quad \text{and} \quad \frac{1}{3} \equiv 21 \pmod{31} ,$$

resulting in the quadratic equation

$$x^2 + 2x + 13 \equiv 0 \pmod{31} .$$

The two roots of this equation in GF(31) are 8 and 21: the first points at an error with a value 1 at location 13 (since $\alpha_{13} = 8$), and the second points at an error with a value -1 at location 5 (since $\alpha_5 = 10 = -21 \text{ MOD } 31$). $\square$

References

[1] C.J. Anfinson, F.T. Luk, "A linear algebraic model of algorithm-based fault tolerance," *IEEE Trans. Comput.,* 37 (1988), 1599–1604.

[2] E.R. Berlekamp, *Algebraic Coding Theory,* Revised Edition, Aegean Park Press, Laguna Hills, California, 1984.

[3] W.G. Bliss, "Circuitry for performing error correction calculations on baseband encoded data to eliminate error propagation," *IBM Tech. Discl. Bull.,* 23 (1981), 4633–4634.

[4] B.E. Boser, E. Sackinger, J. Bromley, Y. Le Cun, L.D. Jackel, "An analog neural network processor with programmable topology," *IEEE J. Solid-State Circuits,* 26 (1991), 2017-–2025.

[5] S. Dutta, V. Cadambe, P. Grover, ""Short-Dot": Computing large linear transforms distributedly using coded short dot products," *Proc. 29th Conf. Neural Inf. Processing Systems (NIPS 2016),* Barcelona, Spain (2016).

[6] M. Fahim, H. Jeong, F. Haddadpour, S. Dutta, V. Cadambe, P. Grover, "On the optimal recovery threshold of coded matrix multiplication," *Proc. 55th Annual Allerton Conference on Communication, Control, and Computing,* Allerton House, Monticello, Illinois (2017),1264–1270.

[7] J. Fan, R. Calderbank, "A modified concatenated coding scheme, with applications to magnetic data storage," *IEEE Trans. Inf. Theory,* 44 (1998), 1565–1574.

[8] S.W. Golomb, L.R. Welch, "Algebraic coding and the Lee metric," in *Error Correcting Codes,* H.B. Mann (Editor), Wiley, New York, 1968, pp. 175–194.

[9] S.W. Golomb, L.R. Welch, "Perfect codes in the Lee metric and the packing of polyominoes," *SIAM J. Appl. Math.,* 18 (1970), 302–317.

[10] A.F. Horadam, "Jacobsthal representation numbers," *Fibonacci Quart.,* 34 (1996), 40–54.

[11] M. Hu, C.E. Graves, C. Li, Y. Li, N. Ge, E. Montgomery, N. Davila, H. Jiang, R.S. Williams, J.J. Yang, Q. Xia, J.P. Strachan, "Memristor-based analog computation and neural classification with a dot product engine," *Adv. Mater.,* 2018, 30, 1705914.

[12] M. Hu, J.P. Strachan, Z. Li, E.M. Grafals, N. Davila, C. Graves, S. Lam, N. Ge, J. Yang, R.S. Williams, "Dot-product engine for neuromorphic computing: Programming 1T1M crossbar to accelerate matrix-vector multiplication," *Proc. 53rd Annual Design Automation Conference (DAC'16),* Austin, Texas (2016), Article No. 19.

[13] F. Haddadpour, V.R. Cadambe, "Codes for distributed finite alphabet matrix-vector multiplication," *Proc. IEEE Int. Symp. Inform. Theory (ISIT 2018),* Vail, Colorado (2018), 1625–1629.

[14] K.-H. Huang, J.A. Abraham, "Algorithm-based fault tolerance for matrix operations," *IEEE Trans. Comput.,* 43 (1984), 518–528.

[15] K.A.S. Immink, "A practical method for approaching the channel capacity of constrained channels," *IEEE Trans. Inf. Theory,* 43 (1997), 1389–1399.

[16] J.-Y. Jou, J.A. Abraham, "Fault-tolerant matrix arithmetic and signal processing on highly-concurrent computing structures," *Proc. IEEE,* 74 (1986), 732–741.

[17] D.E. Knuth, "Efficient balanced codes," *IEEE Trans. Inf. Theory,* 32 (1986), 51–53.

[18] F. Kub, K. Moon, I. Mack, F. Long, "Programmable analog vector–matrix multipliers," *IEEE J. Solid-State Circuits,* 25 (1990), 207–214.

[19] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Trans. Inf. Theory,* 64 (2018), 1514–1529.

[20] K. Lee, C. Suh, K. Ramchandran, "High-dimensional coded matrix multiplication," *Proc. IEEE Int. Symp. Inform. Theory (ISIT 2017),* Aachen, Germany (2017), 2418–2422.

[21] M. Mansuripur, "Enumerative modulation coding with arbitrary constraints and post-modulation error correction coding and data storage systems," *Proc. SPIE,* Vol. 1499 (1991), 72–86.

[22] R.M. Roth, *Introduction to Coding Theory,* Cambridge University Press, Cambridge, UK, 2006.

[23] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J.P. Strachan, M. Hu, R.S. Williams, V. Srikumar, "ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," *Proc. 43rd ACM/IEEE Int'l Symp. on Computer Architecture (ISCA 2016),* Seoul, Korea (2016), pp. 14–26.

[24] M. Vijay, R. Mittal, "Algorithm-based fault tolerance: a review," *Microprocess. Microsyst.,* 21 (1997), 151–161.

[25] Y. Yang, P. Grover, S. Kar, "Computing linear transformations with unreliable components," *IEEE Trans. Inf. Theory,* 63 (2017), 3729–3756.

[26] Q. Yu, M.A. Maddah-Ali, A.S. Avestimehr, "Polynomial codes: an optimal design for high-dimensional coded matrix multiplication," *Proc. 30st Conf. Neural Inf. Processing Systems (NIPS 2017),* Long Beach, CA (2017).

Ron M. Roth (M'88–SM'97–F'03) received the B.Sc. degree in computer engineering, the M.Sc. in electrical engineering, and the D.Sc. in computer science from Technion—Israel Institute of Technology, Haifa, Israel, in 1980, 1984, and 1988, respectively. Since 1988 he has been with the Computer Science Department at Technion, where he now holds the General Yaakov Dori Chair in Engineering. During the academic years 1989–91 he was a Visiting Scientist at IBM Research Division, Almaden Research Center, San Jose, California, and during 1996–97, 2004–05, and 2011–2012 he was on sabbatical leave at Hewlett–Packard Laboratories, Palo Alto, California. He is the author of the book *Introduction to Coding Theory*, published by Cambridge

University Press in 2006. Dr. Roth was an associate editor for coding theory in IEEE TRANSACTIONS ON INFORMATION THEORY from 1998 till 2001, and he is now serving as an associate editor in *SIAM Journal on Discrete Mathematics*. His research interests include coding theory, information theory, and their application to the theory of complexity.