A New Family of Constrained Codes with Applications in Data Storage

Ahmed Hareedy and Robert Calderbank
Electrical and Computer Engineering Department, Duke University, Durham, NC 27705 USA
ahmed.hareedy@duke.edu and robert.calderbank@duke.edu

***Abstract*—Line codes make it possible to mitigate interference, to prevent short pulses, and to generate streams of bipolar signals with no direct-current (DC) power content through balancing. They find application in magnetic recording (MR) devices, in Flash devices, and in optical recording devices. This paper introduces a new family of fixed-length, binary constrained codes, named lexicographically-ordered constrained codes (LOCO codes), for bipolar non-return-to-zero signaling. LOCO codes are capacity achieving, the lexicographic indexing enables simple, practical encoding and decoding, and this simplicity is demonstrated through analysis of circuit complexity. Experimental results demonstrate a gain of up to 10% in rate achieved by LOCO codes with respect to practical run-length-limited codes designed for the same purpose. Simulation results suggest that it is possible to achieve channel density gains of about 20% in MR systems by using a LOCO code to encode only the parity bits of a low-density parity-check code before writing.**

I. Introduction

From data storage to data transmission, line codes are employed in many systems. An early example, introduced in [1], is the family of run-length-limited (RLL) codes used to mitigate inter-symbol interference (ISI) in magnetic recording (MR) systems by appropriately separating transitions. RLL codes are associated with bipolar non-return-to-zero inverted (NRZI) signaling, where a 0 is represented by no transition and a 1 is represented by a transition, with the transitions being from $-A$ to $+A$ and vice versa. RLL codes are characterized by (d, k), where d (resp., k) is the minimum (resp., maximum) number of 0's between adjacent 1's. A $(2, 7)$ RLL code appeared in early IBM disk drives [2].

For simplicity, we abbreviate a run of r consecutive 0's (resp., 1's) to $\mathbf{0}^r$ (resp., $\mathbf{1}^r$). A $\mathcal{T}_x$-constrained code is a code that forbids the patterns in $\mathcal{T}_x \triangleq \{0\mathbf{1}^y0, 1\mathbf{0}^y1 \mid 1 \leq y \leq x\}$ from appearing in any codeword. $\mathcal{T}_x$-constrained codes are associated with bipolar non-return-to-zero (NRZ) signaling, where a 0 is represented by level $-A$ and a 1 is represented by level $+A$. The parameter x separates transitions, which mitigates ISI, serving the same purpose as the parameter d in RLL codes. We focus in this paper on $\mathcal{T}_x$-constrained codes.

Constrained codes continue to be used in modern MR systems [3] to improve the performance of sequence detection on partial response (PR) channels [4], [5]. PR channels with equalization targets that follow the channel impulse response [6] require forbidden patterns to be symmetric. Moreover, constrained codes improve the performance on low resolution media [7] by preventing short pulses. The requirement that the power spectrum of a line code vanishes at frequency zero, i.e., the code is direct-current-free (DC-free), is important in optical recording [8]. This requirement is typically accomplished by balancing signal signs in the stream of transmitted (written) codewords (for a frequency domain approach, see [9]).

Constrained codes also find application in Flash memories. Consider a single-level cell (SLC) Flash memory system. The pattern 101 can result in inter-cell interference (ICI) caused by an unintentional increase of the charge level in the inner cell. The pattern 010 is typically less detrimental, but it can cause problems when block erasure is not adopted. See [10] for a study of balanced constrained codes that alleviate ICI in Flash systems. Line codes also find application in computer standards for data transmission [11].

The idea of lexicographic indexing can be traced back to [1] and to [12]. The RLL codes constructed in [13] are based on [12], and the rates achieved improve upon those of earlier RLL codes. However, these gains are only realized at relatively large code lengths, and therefore at a significant cost in terms of complexity, storage overhead, and error performance. Techniques based on lookup tables incur significant encoding and decoding complexity.

In this paper, we return to the presentation of lexicographic indexing in [1], and develop the idea in the context of a new family of codes, named lexicographically-ordered $\mathcal{T}_x$-constrained codes (LOCO codes). We develop a simple rule for encoding and decoding LOCO codes based on lexicographic indexing. This rule reduces the encoding-decoding of LOCO codes to low-complexity mapping-demapping between the index and the codeword itself. We demonstrate that LOCO codes are capacity achieving codes, and that at moderate lengths, they provide a rate gain of up to 10% compared with practical RLL codes used to achieve the same goals. We also demonstrate density gains of about 20% in modern MR systems by using a LOCO code to protect only the parity bits of a low-density parity-check (LDPC) code via alleviating ISI.

The rest of the paper is organized as follows. In Section II, LOCO codes are formally defined and analyzed. The mapping-demapping between the index of a codeword and the codeword itself is introduced in Section III. Next, the rates of LOCO codes in addition to the practical encoding and decoding algorithms are presented in Section IV. LOCO codes are applied to MR systems in Section V. Finally, the paper is concluded in Section VI.

II. Analysis of LOCO Codes

We start with the formal definition of the proposed fixed-length LOCO codes.

Definition 1. *A LOCO code $\mathcal{C}_{m,x}$, with parameters m and x, is defined by the following properties:*

1) *Each codeword $\mathbf{c} \in \mathcal{C}_{m,x}$ is binary and of length m.*
2) *Codewords in $\mathcal{C}_{m,x}$ are ordered lexicographically.*
3) *Each codeword $\mathbf{c} \in \mathcal{C}_{m,x}$ does not contain any pattern in the set $\mathcal{T}_x$, where:*

$$\mathcal{T}_x \triangleq \{010, 101, 0110, 1001, \ldots, 0\mathbf{1}^x0, 1\mathbf{0}^x1\}; \quad (1)$$

therefore, $|\mathcal{T}_x| = 2x$, with $x \in \{1, 2, \ldots\}$.

978-1-5386-6900-6/19/$31.00 ©2019 IEEE

TABLE I
ALL THE CODEWORDS OF TWO LOCO CODES, $\mathcal{C}_{m,1}$, $m \in \{5,6\}$. THE FOUR DIFFERENT GROUPS OF CODEWORDS ARE EXPLICITLY ILLUSTRATED FOR THE CODE $\mathcal{C}_{6,1}$.

Codeword index $g(\mathbf{c})$	Codewords of the code $\mathcal{C}_{m,1}$		
	$m=5$	$m=6$	
0	00000	000000	Group 1
1	00001	000001	
2	00011	000011	
3	00110	000110	
4	00111	000111	
5	01100	001100	
6	01110	001110	
7	01111	001111	
8	10000	011000	Group 4
9	10001	011001	
10	10011	011100	
11	11000	011110	
12	11001	011111	
13	11100	100000	Group 3
14	11110	100001	
15	11111	100011	
16		100110	
17		100111	
18		110000	Group 2
19		110001	
20		110011	
21		111000	
22		111001	
23		111100	
24		111110	
25		111111	
Code cardinality	$N(5,1)=16$	$N(6,1)=26$	

4) *Codewords in $\mathcal{C}_{m,x}$ are all the codewords satisfying the previous three conditions.*

Throughout the paper, NRZ (resp., NRZI) signaling is adopted for LOCO (resp., RLL) codes.

Remark 1. *In the case of Flash systems, the level $-A$ is replaced by the erasure level E.*

Observe the connection between the forbidden patterns, i.e., the patterns in $\mathcal{T}_x$, and the physics of different data storage systems. As x increases, ISI (resp., ICI) is more alleviated in MR (resp., Flash) systems, and the minimum width of a pulse increases. However, increasing x reduces the rate.

Table I presents the LOCO codes $\mathcal{C}_{m,1}$, $m \in \{5,6\}$. These LOCO codes have $x=1$ and $\mathcal{T}_1 = \{010, 101\}$.

We partition the codewords in $\mathcal{C}_{m,x}$ into four distinct groups:

Group 1: Codewords here start with 00 from the left, i.e., in their left-most bits (LMBs).

Group 2: Codewords here start with 11 from the left.

Group 3: Codewords here start with 10^{x+1} from the left.

Group 4: Codewords here start with 01^{x+1} from the left.

The four groups are shown in Table I for the code $\mathcal{C}_{6,1}$.

Remark 2. *In order to satisfy Condition 3 in Definition 1 for a stream of codewords of a LOCO code $\mathcal{C}_{m,x}$, a bridging pattern, to be discussed later, needs to be added between any two consecutively transmitted (written) codewords in this stream.*

First, we determine the cardinality of $\mathcal{C}_{m,x}$.

Theorem 1. *Let $N(m,x)$ be the cardinality (size) of the LOCO code $\mathcal{C}_{m,x}$, i.e., $N(m,x) = |\mathcal{C}_{m,x}|$. Define:*

$$N(m,x) \triangleq 2, \ m \leq 1. \tag{2}$$

Then, the following recursive formula gives $N(m,x)$:

$$N(m,x) = N(m-1,x) + N(m-x-1,x), \ m \geq 2. \tag{3}$$

Proof: We use the above group structure to prove Theorem 1. See [14], which is the long version of the paper. ■

The value of Theorem 1 is the insight it provides into the structure of $\mathcal{C}_{m,x}$. Not only does Theorem 1 perform enumeration via simple recursion, it also significantly contributes to the low-complexity encoding and decoding schemes, which are based on the lexicographic ordering. Note that $N(m,x)$ is always even.

For $x=1$, the cardinalities form a Fibonacci sequence as (3) becomes:

$$N(m,1) = N(m-1,1) + N(m-2,1). \tag{4}$$

The cardinalities $N(m,1)$ for $m \in \{5,6\}$ are given in the last row of Table I.

Example 1. *Consider the LOCO code $\mathcal{C}_{6,1}$ illustrated in the last column of Table I. From Theorem 1, $N(0,1) \triangleq 2$, $N(1,1) \triangleq 2$, $N(2,1) = 4$, $N(3,1) = 6$, and $N(4,1) = 10$. Thus, and from (3), the cardinality of $\mathcal{C}_{6,1}$ is:*

$$N(6,1) = N(5,1) + N(4,1) = 16 + 10 = 26.$$

We now use the group structure of LOCO codes to define a lexicographic indexing of codewords.

Define the index of a codeword $\mathbf{c} \in \mathcal{C}_{m,x}$ as $g(m,x,\mathbf{c}) \in \{0,1,\ldots,N(m,x)-1\}$, which we also abbreviate to $g(\mathbf{c})$ when the context is clear. Since the four groups can be defined for a LOCO code of any length, we define them for $\mathcal{C}_{m+1,x}$. For Groups 1 and 2 in $\mathcal{C}_{m+1,x}$, let $\mathbf{c} \in \mathcal{C}_{m,x}$ be the corresponding codeword to $\mathbf{c}' \in \mathcal{C}_{m+1,x}$, i.e., the m RMBs in $\mathbf{c}'$ are $\mathbf{c}$. Moreover, for Groups 3 and 4 in $\mathcal{C}_{m+1,x}$, let $\mathbf{c}'' \in \mathcal{C}_{m-x,x}$ be the corresponding codeword to $\mathbf{c}' \in \mathcal{C}_{m+1,x}$, i.e., the $m-x$ RMBs in $\mathbf{c}'$ are $\mathbf{c}''$.

We define the **shift in codeword indices** for different groups in $\mathcal{C}_{m+1,x}$ as follows:

$$\zeta_\ell \triangleq \begin{cases} g(m+1,x,\mathbf{c}') - g(m,x,\mathbf{c}), & \ell \in \{1,2\}, \\ g(m+1,x,\mathbf{c}') - g(m-x,x,\mathbf{c}''), & \ell \in \{3,4\}, \end{cases} \tag{5}$$

where ℓ is the group index. Observe that this shift is fixed for all the codewords in the same group in $\mathcal{C}_{m+1,x}$.

The following lemma gives the values of the shift for all the four groups.

Lemma 1. *The shift in codeword indices defined in (5) for different groups in a LOCO code $\mathcal{C}_{m+1,x}$ is given by:*

$$\zeta_\ell = \begin{cases} 0, & \ell = 1, \\ N(m-x,x), & \ell = 2, \\ \frac{1}{2}N(m+1,x), & \ell = 3, \\ \frac{1}{2}\left[N(m,x) - N(m-x,x)\right], & \ell = 4. \end{cases} \tag{6}$$

Proof: See [14]. ■

Example 2. *The values of ζ_ℓ, $\ell \in \{1,2,3,4\}$, for the LOCO code $\mathcal{C}_{6,1}$ given in the last column of Table I are $\zeta_1 = 0$, $\zeta_2 = N(4,1) = 10$, $\zeta_3 = \frac{1}{2}N(6,1) = 13$, and $\zeta_4 = \frac{1}{2}\left[N(5,1) - N(4,1)\right] = 3$. Note that here $m+1 = 6$, i.e., $m = 5$, and $x = 1$.*

III. PRACTICAL ENCODING AND DECODING OF LOCO CODES

In this section, we describe how lexicographic indexing supports simple, practical encoding and decoding of LOCO codes. The following theorem is fundamental to the encoding and decoding algorithms.

In the following, we define a codeword $\mathbf{c} \in \mathcal{C}_{m,x}$ as $\mathbf{c} \triangleq [c_{m-1} \ c_{m-2} \ \ldots \ c_0]$, where $c_i \in \{0,1\}$, for all i. We also define a decimal variable a_i such that $a_i \triangleq 1$ if $c_i = 1$, and

$a_i \triangleq 0$ if $c_i = 0$. The same applies for $\mathbf{c}' \in \mathcal{C}_{m+1,x}$ and $\mathbf{c}'' \in \mathcal{C}_{m-x,x}$. Note that codeword indexing is trivial for the case of $m = 1$.

Theorem 2. *Consider a LOCO code $\mathcal{C}_{m,x}$ with $m \geq 2$. The index $g(\mathbf{c})$ of a codeword $\mathbf{c} \in \mathcal{C}_{m,x}$ is derived from $\mathbf{c}$ itself according to the following equation:*

$$g(\mathbf{c}) = \frac{1}{2}\left[a_{m-1}N(m,x) + \sum_{i=0}^{m-2} a_i N(i-x+1,x)\right]. \quad (7)$$

Here, we use the abbreviated notation $g(\mathbf{c})$ for simplicity.

Proof: We use Lemma 1 to prove Theorem 2 by induction. See [14]. ∎

The value of Theorem 2 is that it provides the foundation for the practical encoding and decoding algorithms of our LOCO codes via lexicographic indexing. In particular, this theorem introduces a simple one-to-one mapping from $g(\mathbf{c})$ to $\mathbf{c}$, which is the encoding, and a simple one-to-one demapping from $\mathbf{c}$ to $g(\mathbf{c})$, which is the decoding. In summary, Theorem 2 provides the encoding-decoding rule for LOCO codes.

Example 3. *We illustrate Theorem 2 by applying (7) to two codewords in $\mathcal{C}_{6,1}$ given in Table I. The first codeword is the one with the index 9. This codeword has $c_{m-1} = 0$; thus,*

$$g(\mathbf{c}) = \frac{1}{2}\left[0 + \sum_{i=0}^{4} a_i N(i,x)\right]$$
$$= \frac{1}{2}\left[N(0,1) + N(3,1) + N(4,1)\right] = 9.$$

The second codeword is the one with the index 24. This codeword has $c_{m-1} = 1$; thus,

$$g(\mathbf{c}) = \frac{1}{2}\left[N(6,1) + \sum_{i=0}^{4} a_i N(i,1)\right]$$
$$= \frac{1}{2}\left[26 + N(1,1) + N(2,1) + N(3,1) + N(4,1)\right] = 24.$$

Example 3 shows how the index, which implies the original message, can be recovered from the LOCO codeword.

Remark 3. *Lexicographically-ordered RLL (LO-RLL) codes can be constructed via the ideas in [1]. Define the binary difference vector $\mathbf{v}$ of a codeword $\mathbf{c}$ in a LOCO code $\mathcal{C}_{m,x}$, $m \geq 2$, as $\mathbf{v} \triangleq [v_{m-2}\ v_{m-3}\ \ldots\ v_0]$, with $v_i \triangleq c_{i+1} + c_i$, for all i. Thus, a (d,∞) LO-RLL code with $d = x$ and length $m-1$ can also be derived from the LOCO code $\mathcal{C}_{m,x}$ by computing the difference vectors for all the codewords in $\mathcal{C}_{m,x}$ starting with 0 from the left (the remaining difference vectors will be repeated because of symmetry). Consequently, the cardinality of a (d,∞) LO-RLL code with $d = x$ and length $m-1$ is given by:*

$$N^{\text{RLL}}(m-1,d) = \frac{1}{2}N(m,x),\ d = x, \quad (8)$$

which is consistent with the results in [1] (see [14]).

IV. Rate Discussion and Algorithms

We first discuss **bridging patterns**. Consider the following scenario. The codeword at transmission (writing) instance t is ending with 00 from the right, while the codeword at instance $t+1$ is starting with 10 from the left. The stream containing the two codewords will then have the pattern 010, which is a forbidden pattern for any LOCO code. Bridging patterns prevent forbidden patterns from appearing across two consecutive codewords. If the patterns in $\mathcal{T}_x$ are prevented (Condition 3 in Definition 1 is satisfied), any two consecutive transitions will be separated by at least $x+1$ successive bit durations. Transitions are either from 0 to 1, i.e, $-A$ to $+A$, or from 1 to 0, i.e., $+A$ to $-A$.

Define the symbol z as the no transmission (no writing) symbol. Define also the notation $\mathbf{z}^r$ to represent a run of r consecutive z symbols. Our method of bridging is simply to add the bridging pattern $\mathbf{z}^x$ between each two consecutive LOCO codewords. Bridging patterns are ignored at the decoding.

Remark 4. *In the case of Flash systems, transitions are either from 0 to 1, i.e, E to $+A$, or from 1 to 0, i.e., $+A$ to E. Moreover, the no writing symbol z represents the state when the cell is programmed to a charge level about the mid point between E and $+A$.*

An important requirements in constrained codes is **self-clocking** [2], [5]. In particular, the receiver should be capable of retrieving the clock of the transmitter from the signal itself. This requires avoiding long runs of 0's ($-A$'s) and long runs of 1's ($+A$'s). Thus, we construct the following code.

Definition 2. *A self-clocked LOCO (C-LOCO) code $\mathcal{C}^{\text{c}}_{m,x}$ is the code resulting from removing the all 0's and the all 1's codewords from the LOCO code $\mathcal{C}_{m,x}$. In particular:*

$$\mathcal{C}^{\text{c}}_{m,x} \triangleq \mathcal{C}_{m,x} \setminus \{\mathbf{0}^m, \mathbf{1}^m\}, \quad (9)$$

where $m \geq 2$. The cardinality of $\mathcal{C}^{\text{c}}_{m,x}$ is given by:

$$N^{\text{c}}(m,x) = N(m,x) - 2. \quad (10)$$

Now, there exists at least one transition in each codeword in $\mathcal{C}^{\text{c}}_{m,x}$. Define $k^{\text{c}}_{\text{eff}}$ as the maximum number of successive bit durations without a transition in a stream of C-LOCO codewords that belong to $\mathcal{C}^{\text{c}}_{m,x}$, with each two consecutive codewords separated by a bridging pattern. For the sake of abbreviation, we here use the format "codeword at t − bridging pattern − codeword at $t+1$". The scenarios under which $k^{\text{c}}_{\text{eff}}$ is acheieved, using our bridging method, are:

$$1\mathbf{0}^{m-1} - \mathbf{z}^x - \mathbf{0}^{m-1}1 \ \text{ and } \ 0\mathbf{1}^{m-1} - \mathbf{z}^x - \mathbf{1}^{m-1}0.$$

Consequently, we get:

$$k^{\text{c}}_{\text{eff}} = 2(m-1) + x. \quad (11)$$

We are now ready to discuss the rate of C-LOCO codes. A C-LOCO code $\mathcal{C}^{\text{c}}_{m,x}$, with x bridging symbols associated to each codeword, has rate:

$$R^{\text{c}}_{\text{LOCO}} = \frac{\lfloor \log_2 N^{\text{c}}(m,x) \rfloor}{m+x} = \frac{\lfloor \log_2 (N(m,x)-2) \rfloor}{m+x}, \quad (12)$$

where $N(m,x)$ is obtained from the recursive relation (3). The numerator, which is $\lfloor \log_2 (N(m,x)-2) \rfloor$, is the length of the messages $\mathcal{C}^{\text{c}}_{m,x}$ encodes.

Observe that a C-LOCO code $\mathcal{C}^{\text{c}}_{m,x}$ consists of all codewords of length m, with the exception of the two codewords $\mathbf{0}^m$ and $\mathbf{1}^m$, that do not contain any of the forbidden patterns in $\mathcal{T}_x$. Moreover, the number of added symbols for bridging is function of x only, and thus does not grow with m. Therefore, it follows that C-LOCO codes are **capacity-achieving constrained codes**.

Example 4. *Consider again the LOCO code $\mathcal{C}_{6,1}$ in Table I. From (11), the C-LOCO code $\mathcal{C}^{\text{c}}_{6,1}$ derived from $\mathcal{C}_{6,1}$ has:*

$$k^{\text{c}}_{\text{eff}} = 2(6-1) + 1 = 11.$$

TABLE II
RATES AND ADDER SIZES OF C-LOCO CODES $\mathcal{C}^{\mathrm{c}}_{m,x}$ FOR DIFFERENT VALUES OF m AND x. THE CAPACITY IS 0.6942 FOR $x = 1$ AND 0.5515 FOR $x = 2$.

Values of m and x	$R^{\mathrm{c}}_{\mathrm{LOCO}}$	Adder size
$m = 8$, $x = 1$	0.6667	6 bits
$m = 18$, $x = 1$	0.6842	13 bits
$m = 31$, $x = 1$	0.6875	22 bits
$m = 44$, $x = 1$	0.6889	31 bits
$m = 54$, $x = 1$	0.6909	38 bits
$m = 90$, $x = 1$	0.6923	63 bits
$m = 6$, $x = 2$	0.5000	4 bits
$m = 13$, $x = 2$	0.5333	8 bits
$m = 24$, $x = 2$	0.5385	14 bits
$m = 33$, $x = 2$	0.5429	19 bits
$m = 42$, $x = 2$	0.5455	24 bits
$m = 91$, $x = 2$	0.5484	51 bits

Moreover, the length of the messages $\mathcal{C}^{\mathrm{c}}_{6,1}$ encodes is $\lfloor \log_2 (N(6,1) - 2) \rfloor = \lfloor \log_2 24 \rfloor = 4$. From (12), the rate of $\mathcal{C}^{\mathrm{c}}_{6,1}$ is:

$$R^{\mathrm{c}}_{\mathrm{LOCO}} = \frac{\lfloor \log_2 24 \rfloor}{6+1} = \frac{4}{7} = 0.5714.$$

Table II shows the rates of C-LOCO codes $\mathcal{C}^{\mathrm{c}}_{m,x}$ for different values of m and for $x \in \{1, 2\}$. Table II demonstrates that C-LOCO codes have rates up to 0.6923 (resp., 0.5484) for the case of $x = 1$ (resp., $x = 2$) with moderate code lengths. From the literature, the capacity of a $\mathcal{T}_x$-constrained code with $x = 1$ (resp., $x = 2$) is 0.6942 (resp., 0.5515) [4], [5]. The table shows that the rate of the C-LOCO code $\mathcal{C}^{\mathrm{c}}_{90,1}$ (resp., $\mathcal{C}^{\mathrm{c}}_{91,2}$) is within only 0.3% (resp., 0.6%) from the capacity. In fact, these rates even increase with an informed increase in the value of m until they reach the capacity.

From the definition of a LOCO code, an RLL code with parameter d has similar performance to a LOCO code with parameter x. Because of their practicality, we focus on RLL codes generated via finite state machines (FSMs), and decoded via sliding window decoders [2], [4], [5]. For $d = x$, there are three main advantages of LOCO codes over FSM-based RLL codes: 1) LOCO codes achieve higher rates. 2) LOCO codes are immune against error propagation from a codeword to another. 3) Balancing LOCO codes is not only simple, but also incurs a very limited rate loss (see [14, Section VI]).

As for the rate advantage, a practical FSM-based RLL code with $d = 1$ (resp., $d = 2$) typically has a rate of 0.6667 (resp., 0.5000) [2], [4], [5]. The rate gain of moderate-length C-LOCO codes over practical FSM-based RLL codes, where $d = x$, is up to 10% as shown in Table II.

We introduce now the encoding and decoding algorithms of our C-LOCO codes, which are based on Theorem 2. The encoding algorithm is [14, Algorithm 1], and the decoding algorithm is Algorithm 1. An example on how the encoding algorithm works is [14, Example 5]. Example 3 in Section III already showed how the decoding algorithm works.

The size of the adders used to perform different encoding and decoding tasks is $\log_2$ the maximum value $g(\mathbf{c})$ can take that corresponds to a message, and it is given by:

$$s^{\mathrm{c}} = \lfloor \log_2 (N(m,x) - 2) \rfloor . \tag{13}$$

Table II links the rate of a C-LOCO code with its encoding and decoding complexity through the size of the adders to be used. For example, for a C-LOCO code with $x = 1$ (resp., $x = 2$), if a rate of 0.6842 (resp., 0.5333) is needed, adders of size 13 (resp., 8) bits should be used. Note that the multiplication by $\frac{1}{2}$ is just a right shift by one unit in binary, and it can be done only once at the beginning of the encoding-decoding.

Algorithm 1 Decoding C-LOCO Codes

Inputs: Incoming stream of binary C-LOCO codewords, in addition to m, x, and s^{c}.
Use (2) and (3) to compute $N(i,x)$, $i \in \{2, 3, \ldots, m\}$.
for each incoming codeword $\mathbf{c}$ of length m **do**
  Initialize $g(\mathbf{c})$ with 0.
  if $c_{m-1} = 1$ **then**
    $g(\mathbf{c}) \leftarrow g(\mathbf{c}) + \frac{1}{2} N(m,x)$.
  end if
  for $i \in \{m-2, m-3, \ldots, 0\}$ **do**
    if $c_i = 1$ **then**
      $g(\mathbf{c}) \leftarrow g(\mathbf{c}) + \frac{1}{2} N(i-x+1, x)$.
    end if
  end for
  Compute $\mathbf{b} = \mathrm{binary}(g(\mathbf{c}) - 1)$, which has length s^{c}. *(decimal to binary)*
  Ignore the next x bridging symbols.
end for
Output: Outgoing stream of binary messages.

Remark 5. *Observe that (8) in Remark 3 shows that LOCO codes are more efficient compared with LO-RLL codes in the finite-length regime. The reason is that from (8) and (3), the difference between the cardinalities of a LOCO code $\mathcal{C}_{m,x}$ and a (d, ∞) LO-RLL code with $d = x$ and length m is:*

$$\begin{aligned} N(m,x) - N^{\mathrm{RLL}}(m,d) &= N(m,x) - \frac{1}{2} N(m+1, x) \\ &= \frac{1}{2} \left[N(m,x) - N(m-x, x) \right] . \end{aligned} \tag{14}$$

Thus, if the same number of bits is used for bridging, the LOCO code can achieve higher rates at the same code length or lower complexities at the same rate (see [14]).

V. DENSITY GAINS IN MR SYSTEMS

The details of our MR system model are in [14, Section V]. An LDPC encoder is followed by a LOCO encoder, which encodes only the parity bits of a spatially-coupled (SC) LDPC codeword via a C-LOCO code to significantly increase their reliability. NRZ signaling is adopted. The MR channel effects are ISI, jitter, and electronic noise. The channel density, which is the ratio of the read-head pulse duration at half the amplitudes to the bit duration, is swept to generate the plots. The SNR is 13.00 dB. The PR equalization target is $[8\ 14\ 2]$. Observe that this PR target behaves in a way similar to the channel impulse response [6], which is an important reason why we are here adopting the set $\mathcal{T}_x$ of symmetric forbidden patterns. Detection is performed via a BCJR detector. There is an outer looping between the detector, a LOCO decoder, and an LDPC decoder at the receiving end. The LDPC decoder is a fast Fourier transform based q-ary sum-product algorithm (FFT-QSPA) LDPC decoder, with q being set to 2 here.

Lemma 2 gives the overall rate of the LDPC-LOCO coding scheme applied in our system.

Lemma 2. *Consider the following LDPC-LOCO coding scheme. A C-LOCO code of rate $R^{\mathrm{c}}_{\mathrm{LOCO}}$ is used to encode only the parity bits of an LDPC code of rate R_{LDPC}. The overall rate of this scheme is:*

$$R_{\mathrm{ov}} \approx \frac{R_{\mathrm{LDPC}} R^{\mathrm{c}}_{\mathrm{LOCO}}}{R_{\mathrm{LDPC}} R^{\mathrm{c}}_{\mathrm{LOCO}} + (1 - R_{\mathrm{LDPC}})}. \tag{15}$$

Proof: See [14]. ∎

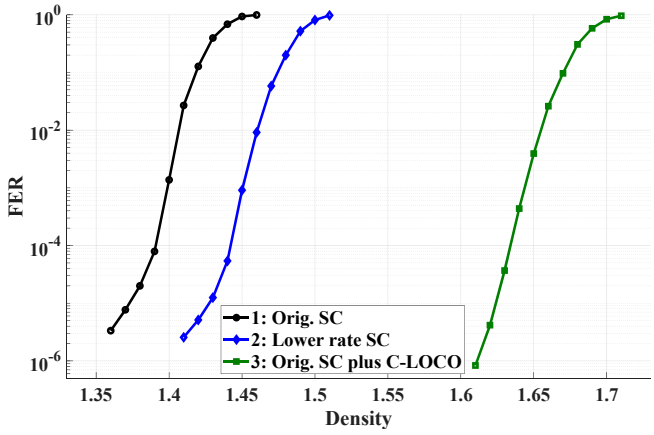

Fig. 1. Density gains achieved by LOCO codes in MR systems.

Lemma 2 demonstrates that the rate loss due to integrating a C-LOCO code in the MR system the way we do it is limited.

The binary SC codes used in our simulations are constructed according to [15], which provides a method to design high performance SC codes for MR systems. SC Code 1 (resp., SC Code 2) has column weight $= 4$, maximum row weight $= 17$ (resp. $= 13$), circulant size $= 37$ (resp., $= 47$), memory $= 1$, and coupling length $= 6$ (resp., $= 7$). Thus, SC Code 1 (resp., SC Code 2) has block length $= 3774$ (resp., $= 4277$) bits and rate ≈ 0.725 (resp., ≈ 0.648).

The C-LOCO code we use in the simulations is the code $\mathcal{C}^{\mathrm{c}}_{18,1}$. This code has $m = 18$ and $x = 1$. Thus, from (11), $\mathcal{C}^{\mathrm{c}}_{18,1}$ has $k^{\mathrm{c}}_{\mathrm{eff}} = 2(17) + 1 = 35$. Moreover, $\mathcal{C}^{\mathrm{c}}_{18,1}$ has $N^{\mathrm{c}}(18, 1) = 8362$, which means the message length is $s^{\mathrm{c}} = \lfloor \log_2 8360 \rfloor = 13$. Thus, from (12), the rate of $\mathcal{C}^{\mathrm{c}}_{18,1}$ is $\frac{13}{18+1} = 0.6842$ since one symbol z is used for bridging.

We generate three plots, as shown in Fig. 1, for the following three simulation setups: 1) SC Code 1 (original SC code) is used for error correction, and no C-LOCO code is applied. 2) SC Code 2 (lower rate SC code) is used for error correction, and no C-LOCO code is applied. 3) SC Code 1 is combined with the C-LOCO code $\mathcal{C}^{\mathrm{c}}_{18,1}$ such that only the parity bits of SC Code 1 are encoded via $\mathcal{C}^{\mathrm{c}}_{18,1}$. The energy per input data bit in all three setups is the same. For Setup 3, we have $R^{\mathrm{c}}_{\mathrm{LOCO}} = 0.6842$, overall length $= 4258$, and overall rate $R_{\mathrm{ov}} \approx 0.643$ from (15). Thus, the overall length and rate in Setup 3 are similar to the length and rate of SC Code 2 in Setup 2.

The frame error rate (FER) versus density plots for the three setups are shown in Fig. 1. The density gain of Setup 3 over Setup 1 (resp., Setup 2) is about 20% (resp., 16%) at FER $\approx 10^{-6}$. The density gain achieved in Setup 3 over Setup 2 implies that exploiting the additional redundancy by applying a C-LOCO code is significantly more effective compared with exploiting this redundancy by adding more parity bits. An intriguing observation from Fig. 1 is that the error floor slope in Setup 3 is sharper than the slope in the other two setups.

While applying the C-LOCO code to the entire LDPC codeword provides higher density gains, the overall rate loss becomes very high since the rate in this case becomes $R_{\mathrm{ov}} \approx R_{\mathrm{LDPC}} R^{\mathrm{c}}_{\mathrm{LOCO}}$. For example, if $\mathcal{C}^{\mathrm{c}}_{18,1}$ is applied to the entire codeword of SC Code 1, the overall rate becomes $R_{\mathrm{ov}} \approx 0.496$, which is a lot lower than R_{ov} in Setup 3. Additionally, the density gains achieved diminish gradually with more bits being encoded via the C-LOCO code. In summary, the proposed idea in Setup 3 offers a better rate-density gain trade-off.

Setup 3 is motivated by the following intuition. Even though only a group of bits in the LDPC codeword (the bits encoded via the LOCO code) have highly reliable LRs while decoding, the information in these highly reliable LRs will be spread to all bits during the message passing procedure. Therefore, the LDPC decoder experiences a better version of the channel, which results in the decoder, aided by the detector and the LOCO decoder, kicking-off its operation at higher densities.

VI. Conclusion

We introduced LOCO codes, where the combination of recursive structure and lexicographic indexing of codewords enables simple mapping-demapping between the index and the codeword itself. We showed that this mapping-demapping enables low complexity encoding and decoding algorithms. We also showed that LOCO codes are capacity achieving, and that at moderate lengths, they provide a rate gain of up to 10% compared with practical RLL codes that are used to achieve the same goals. We demonstrated density gains of about 20% in modern MR systems by integrating a LOCO code with an LDPC code. We suggest that LOCO codes can improve the performance in a wide variety of data storage systems. Ongoing work include asymmetric and non-binary LOCO codes.

Acknowledgment

This research was supported in part by NSF under grant CCF 1717602.

References

[1] D. T. Tang and R. L. Bahl, "Block codes for a class of constrained noiseless channels," *Inf. and Control*, vol. 17, no. 5, pp. 436–461, 1970.
[2] P. Siegel, "Recording codes for digital magnetic storage," *IEEE Trans. Magn.*, vol. 21, no. 5, pp. 1344–1349, Sep. 1985.
[3] B. Vasic and E. Kurtas, *Coding and Signal Processing for Magnetic Recording Systems*. CRC Press, 2005.
[4] R. Karabed and P. H. Siegel, "Coding for higher-order partial-response channels," in *Proc. SPIE 2605, Coding and Signal Process. for Inf. Storage*, Philadelphia, PA, USA, Dec. 1995, pp. 115–127.
[5] K. E. S. Immink, P. H. Siegel, and J. K. Wolf, "Codes for digital recorders," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2260–2299, Oct. 1998.
[6] A. Hareedy, B. Amiri, R. Galbraith, and L. Dolecek, "Non-binary LDPC codes for magnetic recording channels: error floor analysis and optimized code design," *IEEE Trans. Commun.*, vol. 64, no. 8, pp. 3194–3207, Aug. 2016.
[7] K. Harada, N. Maeto, A. Yamazaki, and A. Takeo, "Robust modulation of PWM-based multi-level perpendicular magnetic recording for conventional media," *IEEE Comm. Letters*, vol. 22, no. 4, pp. 724–727, Apr. 2018.
[8] K. A. S. Immink, " Modulation systems for digital audio discs with optical readout," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Process. (ICASSP)*, Atlanta, Georgia, USA, Mar.–Apr. 1981, pp. 587–589.
[9] A. R. Calderbank and J. E. Mazo, "Baseband line codes via spectral factorization," *IEEE J. Sel. Areas Commun.*, vol. 7, no. 6, pp. 914–928, Aug. 1989.
[10] M. Qin, E. Yaakobi, and P. H. Siegel, "Constrained codes that mitigate inter-cell interference in read/write cycles for flash memories," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 836–846, Apr. 2014.
[11] J. Saadé, A. Goulahsen, A. Picco, J. Huloux, and F. Pétrot, "Low overhead, DC-balanced and run length limited line coding," in *Proc. IEEE 19th Workshop on Signal and Power Integrity (SPI)*, Berlin, Germany, May 2015, pp. 1–4.
[12] T. Cover, "Enumerative source encoding," *IEEE Trans. Inf. Theory*, vol. 19, no. 1, pp. 73–77, Jan. 1973.
[13] K. A. S. Immink, "A practical method for approaching the channel capacity of constrained channels," *IEEE Trans. Inf. Theory*, vol. 43, no. 5, pp. 1389–1399, Sep. 1997.
[14] A. Hareedy and R. Calderbank, "LOCO codes: lexicographically-ordered constrained codes," Feb. 2019. [Online]. Available: https://arxiv.org/abs/1902.10898
[15] A. Hareedy, R. Wu, and L. Dolecek, "A channel-aware combinatorial approach to design high performance spatially-coupled codes for magnetic recording systems," Sep. 2018. [Online]. Available: https://arxiv.org/abs/1804.05504