Moment Balancing Templates: Constructions to Add Insertion/Deletion Correction Capability to Error Correcting or Constrained Codes

Hendrik C. Ferreira, *Member, IEEE*, Khaled A. S. Abdel-Ghaffar, *Member, IEEE*, Ling Cheng, Theo G. Swart, *Member, IEEE*, and Khmaies Ouahada, *Member, IEEE*

***Abstract*—Templates are constructed to extend arbitrary additive error correcting or constrained codes, i.e., additional redundant bits are added in selected positions to balance the moment of the codeword. The original codes may have error correcting capabilities or constrained output symbols as predetermined by the usual communication system considerations, which are retained after extending the code. Using some number theoretic constructions in the literature, insertion/deletion correction can then be achieved. If the template is carefully designed, the number of additional redundant bits for the insertion/deletion correction can be kept small—in some cases of the same order as the number of parity bits in a Hamming code of comparable length.**

***Index Terms*—Constrained code, extended codes, insertions/deletions, moment function, number theoretic codes.**

I. Introduction

THE topic of correcting bit or symbol insertions/deletions in coded sequences has been studied since the 1960s, see, e.g., [1]–[13]. However, it remains a difficult field with relatively few published results. Some recent results on relevant coding techniques may be found in, for example, [9]–[11].

This work deals with deterministic code constructions, as introduced by Levenshtein [6]. These constructions guarantee some insertion/deletion correction capability within a fixed length codeword. In contrast, the coding techniques presented in [12] and [13] correct multiple insertions/deletions, but only in a probabilistic sense.

One drawback of the deterministic code constructions in [3]–[6] is that the codeword boundaries must be known. For this purpose, periodic markers or synchronization sequences can be inserted between codewords. In fact, many practical communication systems do insert these after a long sequence containing many short codewords. However, if a synchronization word is to be inserted after every single transmitted short codeword, the amount of additional redundancy or overhead may become unattractive.

Manuscript received June 23, 2008; revised February 20, 2009. Current version published July 15, 2009. This paper was presented in part at the IEEE International Symposium on Information Theory, Nice, France, June 2007 and in part at the IEEE Information Theory Workshop, Porto, Portugal, May 2008.

H. C. Ferreira, L. Cheng, T. G. Swart, and K. Ouahada are with the Department of Electrical and Electronic Engineering Science, University of Johannesburg, Auckland Park, 2006, South Africa (e-mail: hcferreira@uj.ac.za; lcheng@uj.ac.za; tgswart@uj.ac.za; kouahada@uj.ac.za).

K. A. S. Abdel-Ghaffar is with the Department of Electrical and Computer Engineering, University of California, Davis, CA 95616 USA (e-mail: ghaffar@ece.ucdavis.edu).

Communicated by H.-A. Loeliger, Associate Editor for Coding Techniques.

Digital Object Identifier 10.1109/TIT.2009.2023682

Lately, however there has been much interest in codes with very long codewords and superior additive error correction. Note that, in this context, we sometimes also refer to an additive error as a "reversal" or "substitution" error. These long codes include the important class of low-density parity-check (LDPC) codes. For such long codes, marking the boundaries by inserting a short synchronization sequence after each codeword may be tolerable in terms of overhead.

Note also that the powerful long LDPC codes can operate at very low signal-to-noise ratios, still with significant additive error correction gain—a noise environment where bit synchronization failures, leading to insertions/deletions, may indeed occur more frequently. We see this work as a step towards codes that will work in such extreme conditions. Another application for insertion/deletion codes is in magnetic recoding systems that use bit patterned media, where the possibility of write synchronization errors exist [14].

Data storage systems such as magnetic or optical recording systems furthermore have channel input constraints which can usually be modeled as either (d, k) constraints or as spectral constraints.

In most communication or data storage systems, additive errors dominate. For example, on some magnetic recording channels, errors which can be modeled as reversal errors can occur with probability 10^{-6}, while errors which can be modeled as insertions or deletions occur with probability a few orders of magnitude lower. However, if insertions/deletions do occur, the consequences can be catastrophic, with large blocks or files of data being lost. In view of the above mentioned error probabilities, it seems appropriate to concentrate most of the error correcting code redundancy to correct reversal errors, but also to make limited provision for insertion/deletion correction.

In this paper, we thus develop coding systems in which the inner code on the channel may be firstly specified to comply with the communication system's additive error correction or constrained symbol requirements. Next, the inner channel code is extended by adding some additional parity or redundant bits, based on moment balancing templates, to create the option of correcting insertions/deletions, if such errors are detected by means of short synchronization sequences inserted between long codewords. Our constructions can be applied to error correcting codes as well as constrained codes serving as inner channel codes.

The moment balancing templates in this paper were inspired by the constructions in [15] which were developed for the systematic encoding of Levenshtein and Constantin-Rao codes.

The new templates in this paper can be used for the systematic encoding of different classes of number theoretic codes in the literature such as in [3]–[6]. Other work involving systematic encoding of insertion/deletion correcting codes appears in [3], [4], and [16]. Especially when applied to constrained codes or when extending linear error correcting codes, the terminology "moment balancing" may be a more explicit description than "systematic encoding."

The rest of this paper is organized as follows. In Section II we present the concept of universal moment balancing templates, illustrated with a simple numerical example. In Section III we present templates for Levenshtein's two classes of insertion/deletion correcting codes and in Section IV for Tenengolts' binary insertion/deletion correcting codes. We next consider templates for input constrained channels, namely dc-free templates in Section V and (d,k)-constrained templates in Section VI. The results and possible future work are discussed in Section VII.

II. Moment Balancing Templates

Most binary codes correcting insertion/deletion errors impose a condition of the form $\sum_{i=1}^{n} v_i x_i \equiv a \pmod m$ for some integers v_i, a and m, where v_i is the weight assigned to index i and $(x_1 x_2 \ldots x_n)$ is a codeword. Therefore, it is convenient to define the moment of a binary sequence as follows.

Definition 1: Let $\mathbf{x} = (x_1 x_2 \ldots x_n)$ be a binary sequence of length n. The (first-order) moment of $\mathbf{x}$ is defined as

$$\sum_{i=1}^{n} v_i x_i \tag{1}$$

where v_i is the weight assigned to index i.

Typically $v_i = i$ or $v_i = i-1$. Therefore, we define $\sigma(\mathbf{x}) = \sum_{i=1}^{n} i x_i$ and $\sigma'(\mathbf{x}) = \sum_{i=1}^{n} (i-1) x_i$. In the following, we develop the case $v_i = i$ as the other case can be similarly developed.

Definition 2: Let C be a binary code of length n and $N_C(i)$ be the number of codewords $\mathbf{x}$ such that $\sigma(\mathbf{x}) = i$. Let

$$S_C = \{i : N_C(i) > 0\} \tag{2}$$

be the set of moments of C. The moment spectrum of the code C can be represented by the polynomial

$$M_C(y) = \sum_{i \in S_C} N_C(i) y^i. \tag{3}$$

Let C be a (K,k) binary code, i.e., a binary code, which is not necessarily linear, of length K that has 2^k codewords. We are interested in mapping each codeword $\mathbf{c} = (c_1 c_2 \ldots c_K)$ in the code C into a distinct sequence $\mathbf{x} = (x_1 x_2 \ldots x_n)$ whose moment is congruent to some fixed integer modulo some number. This is possible if and only if 2^k is at most equal to the number of sequences of length n satisfying this condition. However, our work is motivated by practical considerations which require both the map and its inverse to be easily implementable. For this reason, we focus on maps in which the code bits $c_1 c_2 \ldots c_K$ appear in fixed positions in the sequence $\mathbf{x}$, i.e., $c_i = x_{\gamma(i)}$ for some $1 \le \gamma(1) < \gamma(2) < \cdots < \gamma(K) \le n$. The remaining bits in $\mathbf{x}$, which are called balancing bits and denoted by $b_1, b_2, \ldots, b_{n-K}$, are positioned in the $n-K$ positions that are not occupied by code bits. In particular, $b_i = x_{\beta(i)}$ where $1 \le \beta(1) < \beta(2) < \cdots < \beta(n-K) \le n$ and $\cup_i \gamma(i)$ and $\cup_i \beta(i)$ are disjoint sets whose union is $\{1, 2, \ldots, n\}$. Let $\sigma_c(\mathbf{x}) = \sum_{i=1}^{K} \gamma(i) c_i$ and $\sigma_b(\mathbf{x}) = \sum_{i=1}^{n-K} \beta(i) b_i$. Then, $\sigma_c(\mathbf{x})$ and $\sigma_b(\mathbf{x})$, which we also denote by σ_c and σ_b if $\mathbf{x}$ is not specified, indicate the contribution of the code bits and the balancing bits, respectively, to the moment of $\mathbf{x}$, defined in (1). In particular, $\sigma_c(\mathbf{x}) + \sigma_b(\mathbf{x}) = \sigma(\mathbf{x})$.

TABLE I
Codebook and Moments for LDPC (7,3) Code

$i=$	1	2	3	4	5	6	7	σ
	0	0	0	0	0	0	0	0
	0	0	1	0	1	1	1	21
	0	1	0	1	1	1	0	17
	0	1	1	1	0	0	1	16
	1	0	0	1	0	1	1	18
	1	0	1	1	1	0	0	13
	1	1	0	0	1	0	1	15
	1	1	1	0	0	1	0	12

Definition 3: We define a *universal* moment balancing template as a sequence of code bit positions and balancing bit positions such that σ_b can assume any integer value modulo m for some fixed positive integer m.

In Section III we shall elaborate on the role of m.

In particular, the moment balancing template is specified by the number and the positions of the balancing bits. For each sequence of code bits, the balancing bits are chosen such that $\sigma_c + \sigma_b$ assumes a specific value modulo m. The collection of all sequences of code bits extended with balancing bits forms an (n,k) code C_{Ext}. Since the balancing bits can be chosen so that their contribution to the moment assumes any given integer value modulo m, the number of balancing bits, $n-K$, in a universal moment balancing template is lower bounded by

$$n - K \ge \lceil \log_2 m \rceil. \tag{4}$$

Example 1: Consider the $(n,k) = (7,3)$ LDPC code C from [17, p. 858]. In Table I, we list all codewords, and their corresponding moments, σ. Note, from (2) and (3), that $S_C = \{0, 12, 13, 15, 16, 17, 18, 21\}$ and

$$M_C(y) = 1 + y^{12} + y^{13} + y^{15} + y^{16} + y^{17} + y^{18} + y^{21}.$$

In Table II, we extend the code to an $(11,3)$ code, C_{Ext}, by using a universal moment balancing template with four balancing bits b_i at positions 1, 2, 4, and 8 and in the extended code set $x_1 = b_1^1$, $x_2 = b_2^2$, $x_4 = b_3^4$ and $x_8 = b_4^8$. The code bits $c_1^3, c_2^5, \ldots, c_7^{11}$ from Table I are written into the remaining positions. The superscripts are added for the reader's benefit to indicate the position of each bit in the sequence. Note that we have chosen the balancing bits in order to have $S_{C_{\text{Ext}}} = \{0, 36\}$ and

$$M_{C_{\text{Ext}}}(y) = 1 + 7y^{36}.$$

TABLE II
LDPC (7,3) CODE WITH BALANCING BITS AND MOMENTS

b_1^1	b_2^2	c_1^3	b_3^4	c_2^5	c_3^6	c_4^7	b_4^8	c_5^9	c_6^{10}	c_7^{11}	σ_c	σ_b	σ
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	1	1	1	36	0	36
1	0	0	1	1	0	1	0	1	1	0	31	5	36
1	1	0	1	1	1	1	0	0	0	1	29	7	36
1	0	1	1	0	0	1	0	0	1	1	31	5	36
1	1	1	0	0	1	1	1	1	0	0	25	11	36
0	0	1	0	1	0	0	1	1	0	1	28	8	36
0	0	1	1	1	1	0	1	0	1	0	24	12	36

In Sections III–VI we develop moment balancing templates in order to extend arbitrary linear additive error correcting codes or constrained codes, to endow them with the option of insertion/deletion correction.

III. TEMPLATES FOR LEVENSHTEIN'S INSERTION/DELETION CORRECTING CODES

Levenshtein presented two classes of single insertion/deletion correcting codes in [6]. A code of length n in each class consists of binary sequences $\mathbf{x} = (x_1 x_2 \ldots x_n)$ satisfying $\sigma(\mathbf{x}) = \sum_{i=1}^{n} i x_i \equiv a \pmod m$ where a and m are fixed integers.

A. Levenshtein's First Class of Insertion/Deletion Correcting Codes

In this class $m \geq n+1$ and the code can correct a single insertion or deletion error. The code's cardinality can be maximized by setting $m = n+1$ and $a = 0$, which we will assume in the following.

As a universal moment balancing template, we take $\beta(i) = 2^{i-1}$ for $i = 1, 2, \ldots, \lceil \log_2(n+1) \rceil$. Clearly,

$$\sum_{i=1}^{\lceil \log_2(n+1) \rceil} \beta(i) b_i = \sum_{i=1}^{\lceil \log_2(n+1) \rceil} 2^{i-1} b_i$$

assumes any given value in $\{0, 1, \ldots, n\}$ for some choice of the b_i's. Notice, from (4), that the number of balancing bits, $\lceil \log_2(n+1) \rceil$, is minimal for a universal moment balancing template with parameter $m = n+1$. The moment balancing template is represented symbolically by the sequence

$$b_1^1 b_2^2 c_1^3 b_3^4 c_2^5 c_3^6 c_4^7 b_4^8 c_5^9 \ldots b_{\lceil \log_2(n+1) \rceil}^{2^{\lceil \log_2(n+1) \rceil - 1}} c_{2^{\lceil \log_2(n+1) \rceil - 1} - \lceil \log_2(n+1) \rceil + 1}^{2^{\lceil \log_2(n+1) \rceil - 1} + 1} \cdots c_K^n.$$

This template can be used to construct an (n, k) code C_{Ext} that can correct a single insertion/deletion from any (K, k) code C, where $K = n - \lceil \log_2(n+1) \rceil$. The rate of C_{Ext}, k/n, approaches the rate, k/K, of the code C as its length, K, increases.

Note that for the $n = 11$ code in Table II, $\sigma(\mathbf{x}) \equiv 0 \pmod{12}$ hence we have extended the $(7, 3)$ code to become a single insertion/deletion correcting $(11, 3)$ code. It can also be considered as a subcode of the $n = 11$ Levenshtein code, or following [15], the $(11, 3)$ code is obtained by systematic encoding of the $n = 11$ Levenshtein code, where we wrote the code bits of the $(7, 3)$ code into the information bit positions of the Levenshtein code.

TABLE III
REDUNDANCY RATES OF THE EXTENDED FIRST-CLASS LEVENSHTEIN CODE WITH $m = n+1$ AND $a = 0$

Redundancy Rate	K	n
44.44%	5	9
28.57%	10	14
10.71%	50	56
6.54%	100	107
1.77%	500	509
0.99%	1000	1010

Table III represents the redundancy rates of Levenshtein's first class of insertion/deletion correcting codes constructed by using the moment balancing template. This indicates a fast redundancy rate decrease with increasing information length.

B. Levenshtein's Second Class of Insertion/Deletion Correcting Codes

In this class $m \geq 2n$ and the code can correct either a single insertion/deletion error or a single reversal error. The code's cardinality can be maximized by setting $m = 2n$ and $a = 0$. By appending one additional balancing bit, we can set up a corresponding moment balancing template. As a universal moment balancing template, we take $\beta(i) = 2^{i-1}$ for $i = 1, 2, \ldots, \lceil \log_2 n \rceil$ and $\beta(\lceil \log_2 n \rceil + 1) = n$. Indeed

$$\sum_{i=1}^{\lceil \log_2 n \rceil + 1} \beta(i) b_i = \sum_{i=1}^{\lceil \log_2 n \rceil} 2^{i-1} b_i + n b_{\lceil \log_2 n \rceil + 1}$$

assumes any given value in $\{0, 1, \ldots, 2n-1\}$ for some choice of the b_i's. Notice, from (4), that the number of balancing bits, $\lceil \log_2 n \rceil + 1$, is minimal for a universal moment balancing template with parameter $m = 2n$. The moment balancing template is represented symbolically by the sequence

$$b_1^1 b_2^2 c_1^3 b_3^4 c_2^5 c_3^6 c_4^7 b_4^8 c_5^9 \ldots b_{\lceil \log_2 n \rceil}^{2^{\lceil \log_2 n \rceil - 1}} c_{2^{\lceil \log_2 n \rceil - 1} - \lceil \log_2 n \rceil + 1}^{2^{\lceil \log_2 n \rceil - 1} + 1} \cdots c_K^{n-1} b_{\lceil \log_2 n \rceil + 1}^n.$$

This template can be used to construct an (n, k) code C_{Ext} that can correct either a single insertion/deletion or a single reversal error from a (K, k) code C, where $K = n - \lceil \log_2 n \rceil - 1$. The rate of C_{Ext}, k/n, approaches the rate, k/K, of the code C as its length K increases.

IV. TEMPLATES FOR TENENGOLTS' BINARY INSERTION/DELETION CORRECTING CODES

This class of codes, published in [3] can correct a single deletion and an error in the preceding bit. Here $\mathbf{x}$ is a codeword if

$$\sigma'(\mathbf{x}) = \sum_{i=1}^{n} (i-1) x_i \equiv a \pmod{2n-2} \tag{5}$$

and

$$\sum_{i=1}^{n} x_i \equiv b \pmod 2 \tag{6}$$

where a and b are fixed integers. As a universal moment balancing template, we take $\beta(1) = 1$, $\beta(i) = 2^{i-2} + 1$ for

$i = 2, 3, \ldots, \lceil \log_2(n-1) \rceil + 1$, and $\beta(\lceil \log_2(n-1) \rceil + 2) = n$. Indeed

$$\sum_{i=1}^{\lceil \log_2(n-1) \rceil + 2} (\beta(i) - 1) b_i = \sum_{i=2}^{\lceil \log_2(n-1) \rceil + 1} 2^{i-2} b_i + (n-1) b_{\lceil \log_2(n-1) \rceil + 2}$$

assumes any given value in $\{0, 1, \ldots, 2n-3\}$ for some choice of the b_i's for $i = 2, 3, \ldots, \lceil \log_2(n-1) \rceil + 2$ to satisfy (5), while b_1 can be independently specified to satisfy (6). The number of balancing bits in this template is $\lceil \log_2(n-1) \rceil + 2$.

The moment balancing template is represented symbolically by the sequence

$$b_1^1 b_2^2 b_3^3 c_1^4 b_4^5 c_2^6 c_3^7 c_4^8 b_5^9 \ldots b_{\lceil \log_2(n-1) \rceil + 1}^{2^{\lceil \log_2(n-1) \rceil - 1} + 1} \ldots c_{2^{\lceil \log_2(n-1) \rceil - 1} - \lceil \log_2(n-1) \rceil + 1}^{2^{\lceil \log_2(n-1) \rceil - 1} + 2} \ldots c_K^{n-1} b_{\lceil \log_2(n-1) \rceil + 2}^{n}.$$

This template can be used to construct an (n, k) code C_{Ext} that can correct a single deletion and an error in the preceding bit from a (K, k) code C, where $K = n - \lceil \log_2(n-1) \rceil - 2$. The rate of C_{Ext}, n/k, approaches the rate, k/K, of the code C as its length, K, increases.

We should note that Tenengolts in [3] presented a systematic version of his code that can be used to construct an (n, k) code from the (K, k) code where $n = K + \lceil \log_2(K-1) \rceil + 6$. For example, with $K = 6$, this version requires a code of length $n = 15$, which includes two bits that act as markers between codewords, while our code C_{Ext} has length $n = 12$.

V. Templates for DC-Free Codes

In this section, we consider a class of constrained codes, namely, dc-free codes. A dc-free code is characterized by the fact that it has zero power at zero frequency [18]. We consider the class of codes that achieve this property by virtue of being weight balanced, i.e., each codeword contains an equal number of 0's and 1's. To preserve this property, a new approach by adding a fixed number of paired bits at determinate indices of the codeword is used to balance the moment of the codeword.

Let $c_1 c_2 \ldots c_K$ be the original dc-free codeword, where $c_i \in \{0, 1\}$, satisfies

$$\sum_{i=1}^{K} c_i = \frac{K}{2}$$

and K is a positive even integer. We propose a pairwise template to moment balance this dc-free codeword. In other words, we insert a fixed number of paired moment balancing bits in the sequence. It is evident that, if each pair of the moment balancing bits appears in the complementary form of "01" or "10", the final codeword is still dc-free. The length of the balanced codeword n, after the insertion of the balancing bits, is assumed to be an even number such that $n - 2^{\lfloor \log_2 n \rfloor} \neq 0$ or 2, i.e., n is neither a power of two, nor two more than a power of two. We will design a template of length n with $t = \lceil \log_2(n+1) \rceil$ pairs of moment balancing bits that supports the encoding of dc-free codewords from a code C of length $K = n - 2t = n - 2\lceil \log_2(n+1) \rceil$ into a code C_{Ext} that can correct a single insertion/deletion.

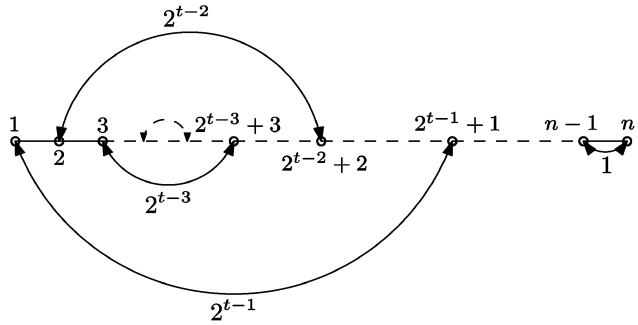

Fig. 1. Indices of the pairwise moment balancing bits in a moment balanced dc-free codeword.

We set the balancing bits in the template according to

$$\beta(i) = \begin{cases} i, & \text{for } 1 \leq i \leq t-1 \\ 2^{i-t+1} + 2t - i - 1, & \text{for } t \leq i \leq 2t-2 \\ n-1, & \text{for } i = 2t-1 \\ n, & \text{for } i = 2t. \end{cases}$$

Symbolically the template can be represented as follows:

$$b_1^1 b_2^2 \ldots b_i^i \ldots b_{t-1}^{t-1} c_1^t b_t^{t+1} b_{t+1}^{t+2} c_2^{t+3} c_3^{t+4} b_{t+2}^{t+5} c_4^{t+6} c_5^{t+7} \ldots b_i^{2^{i-t+1}+2t-i-1} \ldots b_{2t-2}^{2^{t-1}+1} c_{2^{t-1}-2t+4}^{2^{t-1}+2} \ldots c_K^{n-2} b_{2t-1}^{n-1} b_{2t}^{n}. \quad (7)$$

The template can also be illustrated by Fig. 1, where it can be seen which balancing bits are paired. The values on the arcs indicate the difference between the index values for the paired bits.

To simplify the notation and make the β's pair up, let

$$b'_i = b_{2t-i-1} \text{ and } \beta'(i) = \beta(2t-i-1) \quad (8)$$

for $1 \leq i \leq t-1$. Then the bit in position $\beta(i)$ is paired with the bit in position $\beta'(i)$ and to have complementary bits, we set

$$b_i = \overline{b'_i}. \quad (9)$$

In templates from previous sections, only one bit (either a 1 or a 0) was put in a balancing position, where the contribution, $\beta(i)b_i$, to σ_b was either $\beta(i)$ or 0. The difference in σ_b between the two choices is trivially

$$\beta(i) - 0 = \beta(i).$$

However, in this template a pair of bits (either 01 or 10) must be placed in paired balancing positions, therefore the contribution, $\beta(i)b_i + \beta'(i)b'_i$, to σ_b is either $\beta(i)$ or $\beta'(i)$. The difference in σ_b is

$$\beta'(i) - \beta(i) = 2^{t-i} \quad (10)$$

for $i \in \{1, 2, \ldots, t-1\}$.

Theorem 1: Let n be an even positive integer such that $n - 2^{\lfloor \log_2 n \rfloor} \neq 0$ or 2. Then, the sequence (7) is a valid template

that can be used to encode any dc-free sequence $c_1c_2\ldots c_K$ into a dc-free code that can correct a single insertion/deletion by setting $(b_i,b_i')=(0,1)$ or $(b_i,b_i')=(1,0)$ for $i\in\{1,2,\ldots,t-1\}$ and $(b_{2t-1},b_{2t})=(0,1)$ or $(b_{2t-1},b_{2t})=(1,0)$.

Proof: First we show that the template in (7) is valid, i.e., the positions of the balancing bits are well defined. For $1\le i\le t-1$, $\beta(i)=i$ and $\beta(i)$ assumes different values in the range $1\le\beta(i)\le t-1$. For $t\le i\le 2t-2$, $\beta(i)=2^{i-t+1}+2t-i-1$ and $\beta(i)$ assumes different values in the range $t+1\le\beta(i)\le 2^{t-1}+1$. We also have $\beta(2t-1)=n-1$ and $\beta(2t)=n$. We can write $n=2^s+p$ where $s=\lfloor\log_2 n\rfloor$ and $p=n-2^{\lfloor\log_2 n\rfloor}$. Since n is even and $n-2^{\lfloor\log_2 n\rfloor}\neq 0$ or 2, then $p\ge 4$. Hence, $t=\lceil\log_2(n+1)\rceil=s+1$ and $2^{t-1}+1=2^s+1<2^s+p-1=n-1=\beta(2t-1)$. This proves that all balancing bits occupy distinct positions between 1 and n. Next, we show that the contribution of the balancing bits to the moment, σ_b, can assume any integer value modulo $n+1$ by proper choice of b_i' for $i=1,2,\ldots,t-1$ and b_{2t}. Taking (8) into account, the net contribution from the balancing pairs is

$$\begin{aligned}\sigma_b&=\sum_{i=1}^{2t}\beta(i)b_i\\&=\sum_{i=1}^{t-1}\left[\beta(i)b_i+\beta'(i)b_i'\right]+(n-1)b_{2t-1}+nb_{2t}.\end{aligned}\tag{11}$$

Considering (9) and that $b_{2t-1}=\overline{b}_{2t}$, combine (10) with (11) to get

$$\begin{aligned}\sigma_b&=\sum_{i=1}^{t-1}\left[\beta(i)b_i+(2^{t-i}+\beta(i))b_i'\right]+(n-1)\overline{b}_{2t}+nb_{2t}\\&=\sum_{i=1}^{t-1}\beta(i)(\overline{b_i'}+b_i')+\sum_{i=1}^{t-1}2^{t-i}b_i'\\&\quad+(n-1)(\overline{b}_{2t}+b_{2t})+b_{2t}\\&=\sum_{i=1}^{t-1}\beta(i)+\sum_{i=1}^{t-1}2^{t-i}b_i'+n-1+b_{2t}\\&=\Lambda+\sum_{i=1}^{t-1}2^{t-i}b_i'+\overline{b}_{2t}\end{aligned}\tag{12}$$

where

$$\Lambda=\frac{t(t-1)}{2}+n-1.$$

It is evident form (12) that the terms added to the constant Λ can enumerate all the values in $\{0,1,\ldots,2^t-1\}$ by judiciously choosing $b_{2t-2}=b_1',b_{2t-3}=b_2',\ldots b_t=b_{t-1}'$ and b_{2t}. As $2^t=2^{\lceil\log_2(n+1)\rceil}\ge n+1$, σ_b can assume any desired value modulo $n+1$ by proper choice of the balancing bits. □

We notice that the last pair (b_{2t-1},b_{2t}) of moment balancing bits can be placed at any indices that are adjacent but unoccupied by other balancing bits. To simplify notation, we choose to set them in the last two positions.

The template in (7) can be used to construct an (n,k) code C_{Ext} satisfying the dc-free constraint that can correct a single insertion/deletion from a (K,k) code C satisfying the dc-free constraint, where $K=n-2\lceil\log_2(n+1)\rceil$ for large n. The rate of C_{Ext}, k/n, approaches the rate, k/K, of the code C as its length, K, increases.

VI. Templates for (d,k)-Constrained Codes

We next develop templates for (d,k)-constrained codes, where d is the minimum number of 0's and k is the maximum number of 0's between any two 1's in the coded sequences. Codes with $d=1$ and $d=2$ find the widest application in practical recording systems [18]. Usually the d restriction should be enforced, while the k restriction can be somewhat relaxed in the coded sequences.

In the following example, we assume that the code C is $(d=1)$-constrained. Our objective is to maintain this constraint while giving the code the capability to correct a single insertion/deletion by using sequences in Levenshtein's first class of insertion/deletion correcting code. We use the template defined by $\beta(i)=i$ for $i=1,2,\ldots,5$ and $\beta(i)=2^{i-3}$ for $i=6,7,\ldots,\lceil\log_2(n+1)\rceil+2$. Furthermore, we place one zero before and after each balancing bit if the neighboring bit, to the left and to the right, respectively, is not a balancing bit. The bits $b_1,\ldots,b_5$ can always be chosen to satisfy the $(d=1)$-constraint and contribute the moments $0,1,\ldots,7$ as follows:

$$\sigma(b_1\ldots b_5)=\begin{cases}0, & \text{for } b_1b_2b_3b_4b_5=00000\\ j, & b_j=1 \text{ for } 1\le j\le 5,\ b_i=0 \text{ for } i\neq j\\ 6, & \text{for } b_1b_2b_3b_4b_5=10001\\ 7, & \text{for } b_1b_2b_3b_4b_5=01001.\end{cases}$$

Hence

$$\sum_{i=1}^{\lceil\log_2(n+1)\rceil+2}\beta(i)b_i=\sigma(b_1b_2\ldots b_5)+\sum_{i=6}^{\lceil\log_2(n+1)\rceil+2}2^{i-3}b_i$$

assumes any given value in $\{0,1,\ldots,n\}$ for some choice of the b_i's. The moment balancing template is represented symbolically by the sequence

$$b_1^1b_2^2b_3^3b_4^4b_5^500b_6^80c_1^{10}c_2^{11}\ldots c_4^{14}0b_7^{16}0\ldots 0b_{\lceil\log_2(n+1)\rceil+2}^{2^{\lceil\log_2(n+1)\rceil-1}}0\,c_{2^{\lceil\log_2(n+1)\rceil-1}-3\lceil\log_2(n+1)\rceil+5}^{2^{\lceil\log_2(n+1)\rceil-1}+2}\ldots c_K^n.$$

This template can be used to construct an (n,k) code C_{Ext} satisfying the $(d=1)$-constraint that can correct a single insertion/deletion from a (K,k) code C satisfying the $(d=1)$-constraint, where $K=n-3\lceil\log_2(n+1)\rceil+3$ for large n. The rate of C_{Ext}, k/n, approaches the rate, k/K, of the code C as its length, K, increases.

A similar construction can be further devised to satisfy the $(d=2)$-constraint using the template

$$b_1^1b_2^2b_3^3b_4^4b_5^500b_6^800c_1^{11}c_2^{12}c_4^{13}00b_7^{16}00\ldots00b_{\lceil\log_2(n+1)\rceil+2}^{2^{\lceil\log_2(n+1)\rceil-1}}00\,c_{2^{\lceil\log_2(n+1)\rceil-1}-5\lceil\log_2(n+1)\rceil+13}^{2^{\lceil\log_2(n+1)\rceil-1}+3}\ldots c_K^n$$

where two zeros are placed before and after each balancing bit if the neighboring bit, to the left and to the right, respectively, is not a balancing bit. The bits $b_1,\ldots,b_5$ and b_i's are chosen as before. This template can be used to construct an (n,k) code C_{Ext} satisfying the $(d=2)$-constraint that can correct a single

insertion/deletion from a (K, k) code C satisfying the $(d = 2)$-constraint, where $K = n - 5\lceil \log_2(n+1) \rceil + 10$ for large n. The rate of C_{Ext}, k/n, approaches the rate, k/K, of the code C as its length, K, increases.

VII. Discussion

Generalizing and extending our previous work on the systematic encoding of some number theoretic codes, we have developed several universal moment balancing templates. We have shown that these templates can be applied to unconstrained error correcting codes, or to constrained channel codes. The small redundancy added by our moment balancing template may make longer codes attractive for some practical applications.

Note that the moment balancing templates can also be used with convolutional codes. Long coded sequences can be generated and the shift register flushed with 0's. By applying bidirectional Viterbi decoding [13] up to the index with an insertion/deletion, some reversal error correction may be possible.

In this work, we considered *universal* moment balancing templates. These templates guarantee that every sequence of a given length K can be encoded into a codeword $\mathbf{x}$ in C_{Ext} by inserting a number of balancing bits in given positions such that $\sigma(\mathbf{x})$ can be made to assume any integer value modulo some positive number m regardless of the value of σ_c. We can also consider *optimized* moment balancing templates, for a given code C of length K and a given integer a, that guarantees that every sequence from the code C can be encoded into a codeword $\mathbf{x}$ in C_{Ext} by inserting a number of balancing bits in given positions such that $\sigma(\mathbf{x}) \equiv a \pmod m$. Since the values assumed by σ_c may be restricted depending on the code C, and since a is given, it is not necessary, in an optimized moment balancing template to have σ_b assuming every integer value modulo m. In particular, the number of balancing bits $n - K$ in an optimized moment balancing template may be less than the lower bound (4). Therefore, it is possible that the number of balancing bits in an optimized moment balancing template is less than the number of balancing bits in a universal moment balancing template and using optimized templates may lead to higher code rates. Constructing optimized moment balancing templates, for specific codes or classes of codes, is an interesting topic for future research.

Although we were primarily interested in applying the moment balancing templates for insertion/deletion correction, these templates may also be used for the systematic encoding of other codes constructed using the moment function. Future applications may include codes correcting asymmetrical errors, constant weight codes, and spectral shaping codes.

References

[1] E. N. Gilbert, "Synchronization of binary messages," *IRE Trans. Inf. Theory*, vol. 6, no. 4, pp. 470–477, Sep. 1960.

[2] R. P. Varshamov and G. M. Tenengolts, "Correction code for single asymmetric errors," *Avtom. Telemekh.*, vol. 26, no. 2, pp. 288–292, 1965.

[3] G. M. Tenengolts, "Class of codes correcting bit loss and errors in the preceding bit," *Avtom. Telemekh.*, vol. 37, no. 5, pp. 174–179, 1976.

[4] G. M. Tenengolts, "Nonbinary codes, correcting single deletion or insertion," *IEEE Trans. Inf. Theory*, vol. IT-30, pp. 766–769, Sep. 1984.

[5] V. I. Levenshtein, "Binary codes capable of correcting spurious insertions and deletions of ones," *Probl. Pered. Inform.*, vol. 1, no. 1, pp. 12–25, 1965.

[6] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Sov. Phys.–Dokl.*, vol. 10, no. 8, pp. 707–710, Feb. 1966.

[7] A. S. J. Helberg, W. A. Clarke, H. C. Ferreira, and A. J. H. Vinck, "A class of DC free, synchronization error correcting codes," *IEEE Trans. Magn.*, vol. 29, pp. 4048–4049, Nov. 1993.

[8] A. S. J. Helberg and H. C. Ferreira, "On multiple insertion/deletion correcting codes," *IEEE Trans. Inf. Theory*, vol. 48, pp. 305–308, Jan. 2002.

[9] P. A. H. Bours, "Construction of fixed-length insertion/deletion correcting runlength-limited codes," *IEEE Trans. Inf. Theory*, vol. 40, pp. 1841–1856, Nov. 1994.

[10] W. A. Clarke and H. C. Ferreira, "A new linear, quasicyclic multiple insertion/deletion correcting code," in *Proc. Pacific Rim Conf. Communications, Computers and Signal Processing* , Victoria, BC, Canada, 2003, pp. 899–902.

[11] L. Dolecek and V. Anantharam, "A synchronization technique for array-based LDPC codes in channels with varying sampling rate," in *Proc. IEEE Int. Symp. Information Theory*, Seattle, WA, Jul. 2006, pp. 2057–2061.

[12] M. C. Davey and D. J. C. MacKay, "Reliable communication over channels with insertions, deletions and substitutions," *IEEE Trans. Inf. Theory*, vol. 47, pp. 687–698, Feb. 2001.

[13] L. Cheng, H. C. Ferreira, and T. G. Swart, "Bidirectional Viterbi decoding using the Levenshtein distance metric for deletion channels," in *Proc. Information Theory Workshop*, Chengdu, China, 2006, pp. 254–258.

[14] J. Hu, T. M. Duman, E. M. Kurtas, and M. F. Erden, "Bit-patterned media with written-in errors: Modeling, detection, and theoretical limits," *IEEE Trans. Magn.*, vol. 43, pp. 3517–3524, Aug. 2007.

[15] K. A. S. Abdel-Ghaffar and H. C. Ferreira, "Systematic encoding of the Varshamov-Tenengolts codes and the Constantin-Rao codes," *IEEE Trans. Inf. Theory*, vol. 44, pp. 340–345, Jan. 1998.

[16] K. Saowapa, H. Kaneko, and E. Fujiwara, "Systematic deletion/insertion error correcting codes with random error correction capability," in *Proc. Int. Symp. Defect and Fault Tolerance in VLSI Systems*, Albuquerque, NM, 1999, pp. 284–292.

[17] S. Lin and D. J. Costello, Jr*, Error Control Coding*, 2nd ed. Upper Saddle River, NJ: Pearson Prentice-Hall, 2004.

[18] K. A. S. Immink, *Codes for Mass Data Storage Systems*. The Netherlands: Shannon Foundation, 1999.

Hendrik C. Ferreira (M'80) was born and educated in South Africa. He received the D.Sc. (Eng.) degree from the University of Pretoria, South Africa, in 1980.

From 1980 to 1981, he was a Postdoctoral Researcher with the Linkabit Corporation, San Diego, CA. In 1983, he joined the Rand Afrikaans University (now the University of Johannesburg), Johannesburg, South Africa, where he was promoted to Professor in 1989. He has served two terms as Chairman of the Department of Electrical and Electronic Engineering, from 1994 to 1999. His research interests are in digital communications and information theory, especially coding techniques. From 1989 to 1993, he held a "Presidential Award for Young Investigators," a prestigious research grant from the South African Foundation for Research Development. Since 1984, he has been a visiting researcher with various institutions in the United States and Europe.

Dr. Ferreira is a Past Chairman of the Communications and Signal Processing Chapter of the IEEE South Africa section, and was Editor-in-Chief of the *Transactions of the South African Institute of Electrical Engineers* from 1997 to 2006. He has served as chairman of several conferences, including the international 1999 IEEE Information Theory Workshop , Kruger National Park, South Africa.

Khaled A. S. Abdel-Ghaffar (M'94) received the B.Sc. degree from Alexandria University, Alexandria, Egypt, in 1980, and the M.S. and Ph.D. degrees from the California Institute of Technology, Pasadena, in 1983 and 1986, respectively, all in electrical engineering.

In 1988, he joined the University of California, Davis, where he is now a Professor of Electrical and Computer Engineering. He did research at the IBM Almaden Research Center, San Jose, CA, Delft University of Technology, The Netherlands, the University of Bergen, Norway, and Alexandria University. His main interest is coding theory.

Dr. Abdel-Ghaffar served as an Associate Editor for Coding Theory for the IEEE Transactions on Information Theory from 2002 to 2005. He is a corecipient of the IEEE Communications Society 2007 Stephen O. Rice Prize paper award.

Ling Cheng received the B.Eng. degree in electrical and electronic engineering from the Huazhong University of Science and Technology, Wuhan, Hubei, China, in 1995. He received the M.Eng. degree in electrical and electronic engineering from the University of Johannesburg, South Africa, in 2005.

From 1995 to 2002, he was with China Mobile Communications Corporation. From 2003 to 2004, he was with Telematic Technologies. He is currently studying toward the D.Eng. degree with the University of Johannesburg, South Africa. His research interests include power-line communications, error correction coding and spectral shaping techniques.

Theo G. Swart (M'05) received the B.Eng. and M.Eng. degrees in electric and electronic engineering from the Rand Afrikaans University, South Africa, in 1999 and 2001, respectively, and the D.Eng. degree from the University of Johannesburg, South Africa, in 2006.

He is currently employed as a Senior Researcher with the Telecommunications Research Group, University of Johannesburg. His research interests include digital communications, power-line communications, and error-correction coding.

Khmaies Ouahada (M'04) received the B.Eng. degree in electric and electronic engineering from the University of Khartoum, Sudan, in 1995 and the M.Eng. degree in electric and electronic engineering from the Rand Afrikaans University, South Africa, in 2002.

He is currently pursuing the D.Eng. degree with the University of Johannesburg, South Africa. His research interests include digital communications, error correction coding, and spectral shaping techniques.