[9] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD: John Hopkins Univ. Press, 1996, p. 599.
[10] B. M. Hochwald and T. L. Marzetta, "Unitary space-time modulation for multiple-antenna communications in Rayleigh flat fading," *IEEE Trans. Inform. Theory*, vol. 46, pp. 543–564, Mar. 2000.
[11] B. M. Hochwald, T. L. Marzetta, T. J. Richardson, W. Sweldens, and R. Urbanke, "Systematic design of unitary space-time constellations," *IEEE Trans. Inform. Theory*, vol. 46, pp. 1962–1973, Sept. 2000.
[12] B. L. Hughes, "Differential space-time modulation," *IEEE Trans. Inform. Theory*, vol. 46, pp. 2567–2578, Nov. 2000.
[13] T. L. Marzetta and B. M. Hochwald, "Capacity of a mobile multiple-antenna communication link in Rayleigh flat fading," *IEEE Trans. Inform. Theory*, vol. 45, pp. 139–157, Jan. 1999.
[14] M. L. McCloud, M. Brehler, and M. K. Varanasi, "Signal design and convolutional coding for noncoherent space-time communication on the block-Rayleigh-fading channel," *IEEE Trans. Inform. Theory*, vol. 48, pp. 1186–1194, May 2002.
[15] V. Tarokh, H. Jafarkhani, and A. R. Calderbank, "Space-time block codes from orthogonal designs," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1456–1467, July 1999.
[16] V. Tarokh, N. Seshadri, and A. R. Calderbank, "Space-time codes for high data rate wireless communication: Performance criterion and code construction," *IEEE Trans. Inform. Theory*, vol. 44, pp. 744–765, Mar. 1998.
[17] V. Tarokh and M. Kim, "Existence and construction of noncoherent unitary space-time codes," *IEEE Trans. Inform. Theory*, vol. 48, pp. 3112–3117, Dec. 2002.
[18] İ. E. Telatar, "Capacity of multi-antenna gaussian channels," *Europ. Trans. Telecommun.*, vol. 10, no. 6, pp. 585–595, Nov. 1999.
[19] O. Tirkkonen and A. Hottinen, "Complex space-time block codes for four tx antennas," in *Proc. IEEE GLOBECOM*, San Francisco, CA, Nov. 2000, pp. 1005–1009.
[20] L. Zheng and D. N. C. Tse, "Communicating on the grassmann manifold: A geometric approach to the noncoherent multiple antenna channel," *IEEE Trans. Inform. Theory*, vol. 48, pp. 359–383, Feb. 2002.

Sliding-Block Decodable Encoders Between (d, k) Runlength-Limited Constraints of Equal Capacity

Navin Kashyap, *Member, IEEE,* and Paul H. Siegel, *Fellow, IEEE*

***Abstract*—We determine the pairs of (d, k)-constrained systems, $\mathcal{S}(d, k)$ and $\mathcal{S}(\hat{d}, \hat{k})$, of equal capacity, for which there exists a rate 1:1 sliding-block-decodable encoder from $\mathcal{S}(d, k)$ to $\mathcal{S}(\hat{d}, \hat{k})$. In all cases where there exists such an encoder, we explicitly describe the encoder and its corresponding sliding-block decoder.**

***Index Terms*—(d, k)-constrained systems, finite-state encoders, sliding-block decoders.**

I. Introduction

Given nonnegative integers d, k, with $d < k$, we say that a binary sequence is (d, k)-constrained if every run of zeros has length at most k and any two successive ones are separated by a run of zeros of length at least d. A (d, k)-*constrained system* is defined to be the set of all finite-length (d, k)-constrained binary sequences. The above definition can be extended to the case $k = \infty$ by not imposing an upper bound on the lengths of zero runs. In other words, a binary sequence is said to be (d, ∞)-constrained if any two successive ones are separated by at least d zeros, and a (d, ∞)-*constrained system* is defined to be the set of all finite-length (d, ∞)-constrained binary sequences. From now on, when we refer to (d, k)-constrained systems, we shall also allow k to be ∞. Note that the above definition allows finite-length (d, k)-constrained sequences to begin or end with a run of fewer than d zeros.

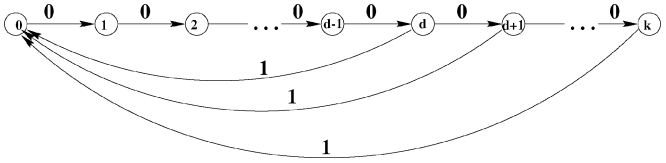

Fig. 1. Graph, $\mathcal{G}_{d,k}$, generating the (d, k)-constrained system $\mathcal{S}(d, k)$ for finite k.

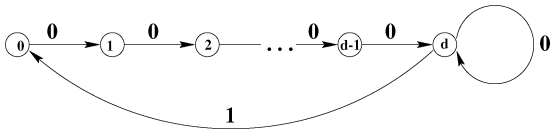

Fig. 2. Graph, $\mathcal{G}_{d,\infty}$, generating the (d, ∞)-constrained system $\mathcal{S}(d, \infty)$.

Binary sequences satisfying some (d, k) constraint are commonly used to encode information in digital and optical recording systems [1]. The parameter k is imposed to guarantee sufficient sign changes in the recorded waveform which are required to prevent clock drift during readback. The parameter d is needed to prevent intersymbol interference.

It is possible to give a convenient graphical description of (d, k)-constrained systems as follows (cf. [1], [2, Chs. 2, 3]). We define a *labeled graph*, $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{L})$, to be a finite directed graph with vertex set $\mathcal{V}$, edge set $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$, and edge labeling $\mathcal{L} : \mathcal{E} \to \Sigma$, where Σ is a finite alphabet. A labeled graph can be used to generate sequences of symbols from Σ by reading off the labels along paths in the graph. A *constrained system*, $\mathcal{S}$ or $\mathcal{S}(\mathcal{G})$, is the set of all finite-length[1] sequences obtained by reading off the labels along paths in a labeled graph $\mathcal{G}$. Any (d, k)-constrained system, $\mathcal{S}(d, k)$, can be generated from an appropriate labeled graph: for finite k, $\mathcal{S}(d, k)$ is the constrained system generated by the labeled graph $\mathcal{G}_{d,k}$ given in Fig. 1, while $\mathcal{S}(d, \infty)$ is generated by the labeled graph $\mathcal{G}_{d,\infty}$ shown in Fig. 2. Note that the edge labels for both these graphs come from the binary alphabet $\{0, 1\}$.

Before proceeding further, we would like to make a remark concerning the notation we shall use in this correspondence. While $\mathcal{S}(d, k)$ will be primarily used to denote the (d, k)-constrained system of finite sequences, we shall also occasionally use the same notation for the one-sided shift of infinite (d, k)-constrained sequences, or the shift space of bi-infinite (d, k)-constrained sequences. In such cases, it should be clear from the context which constrained system we mean to consider.

Manuscript received August 20, 2003; revised January 21, 2004. This work was supported by Applied Micro Circuits Corporation and by the Center for Magnetic Recording Research at the University of California, San Diego, and was performed while N. Kashyap was at the University of California, San Diego.

N. Kashyap is with the Department of Mathematics and Statistics, Queen's University, Kingston, ON K7L 3N6, Canada (e-mail: nkashyap@mast.queensu.ca).

P. H. Siegel is with the Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093-0407 USA (e-mail: psiegel@ece.ucsd.edu).

Communicated by Ø. Ytrehus, Associate Editor for Coding Techniques.

Digital Object Identifier 10.1109/TIT.2004.828145

[1]Sometimes, we shall also find it necessary to consider the constrained system of *infinite* sequences $s_0 s_1 s_2 \ldots$, $s_i \in \Sigma$, of edge labels, or the constrained system of *bi-infinite* sequences $\ldots s_{-2} s_{-1} s_0 s_1 s_2 \ldots$, $s_i \in \Sigma$. In the terminology of symbolic dynamics (cf. [2]), a constrained system of bi-infinite sequences is called a *shift space*, and a constrained system of infinite sequences is called a *one-sided shift*.

Given a (d,k)-constrained system, $\mathcal{S}(d,k)$, let $q_{d,k}(n)$ be the number of length-n sequences in $\mathcal{S}(d,k)$. The *Shannon capacity*, or simply *capacity*, of $\mathcal{S}(d,k)$ is defined as

$$C(d,k) = \lim_{n\to\infty} \frac{1}{n} \log_2 q_{d,k}(n). \tag{1}$$

It is well known (see, e.g., [3]) that $C(d,k) = \log_2 \rho_{d,k}$, where $\rho_{d,k}$ is the unique largest magnitude root of a certain polynomial, $\chi_{d,k}(z)$, called the *characteristic polynomial* of the constraint. $\chi_{d,k}(z)$ is, in fact, the characteristic polynomial of the adjacency matrix of the corresponding labeled graph $\mathcal{G}_{d,k}$. When k is finite, $\chi_{d,k}(z)$ takes the form

$$\chi_{d,k}(z) = z^{k+1} - \sum_{j=0}^{k-d} z^j \tag{2}$$

and when $k = \infty$

$$\chi_{d,\infty}(z) = z^{d+1} - z^d - 1. \tag{3}$$

The root $\rho_{d,k}$ is always real and lies in the interval $(1,2]$, so that $0 < C(d,k) \leq 1$. In fact, $C(d,k) = 1$ if and only if $(d,k) = (0,\infty)$.

It is easily verified that certain pairs of (d,k)-constrained systems have the same capacity. For example, we have the identities

$$C(d,2d) = C(d+1,3d+1) \tag{4}$$

$$C(d,\infty) = C(d-1,2d-1) \tag{5}$$

true for all $d \geq 1$. The first equality is a consequence of the fact that $\chi_{d+1,3d+1}(z)$ can be factorized as $(z^{d+1}+1)\,\chi_{d,2d}(z)$. Since all the roots of $z^{d+1}+1$ lie on the unit circle, while the largest roots of the χ-polynomials lie outside the unit circle, we must have $\rho_{d,2d} = \rho_{d+1,3d+1}$. Similarly, the factorization

$$\chi_{d-1,2d-1}(z) = \chi_{d,\infty}(z) \sum_{i=0}^{d-1} z^i$$

yields (5), since

$$\sum_{i=0}^{d-1} z^i = (z^d - 1)/(z-1)$$

has all its roots on the unit circle as well.

Repeatedly applying the two identities above also yields the chain of equalities

$$C(1,2) = C(2,4) = C(3,7) = C(4,\infty). \tag{6}$$

In [5], [6], it was shown that no equalities other than those listed in (4)–(6) are possible among the capacities $C(d,k)$.

Given a pair of (d,k)-constrained systems, $\mathcal{S}(d,k)$ and $\mathcal{S}(\hat{d},\hat{k})$, with the same capacity, a question that naturally arises in the context of constrained coding is whether or not there exists a rate 1:1, finite-state encoder from $\mathcal{S}(d,k)$ to $\mathcal{S}(\hat{d},\hat{k})$ that is sliding-block decodable. We provide a brief review of the terminology used here. The reader is referred to [1] for a thorough discussion of encoders and decoders for constrained systems.

A rate p:q finite-state encoder is a finite-state machine, shown schematically in the top half of Fig. 3, that accepts a block of p bits as input and generates a binary codeword of length q as output. The actual output codeword depends on the input block and the current internal state of the encoder. Upon generating an output, the internal state of the encoder may change depending on the output and the current state. A rate p:q encoder from $\mathcal{S}(d,k)$ to $\mathcal{S}(\hat{d},\hat{k})$ must satisfy the following requirement: in response to a sequence of p-bit input blocks whose concatenation is in $\mathcal{S}(d,k)$, the encoder must produce a sequence of q-bit codewords whose concatenation lies in $\mathcal{S}(\hat{d},\hat{k})$. In particular, a rate 1:1 encoder from $\mathcal{S}(d,k)$ to $\mathcal{S}(\hat{d},\hat{k})$, in response to an input sequence from $\mathcal{S}(d,k)$, produces an output sequence belonging to $\mathcal{S}(\hat{d},\hat{k})$ by sequentially replacing each input bit by an output bit.

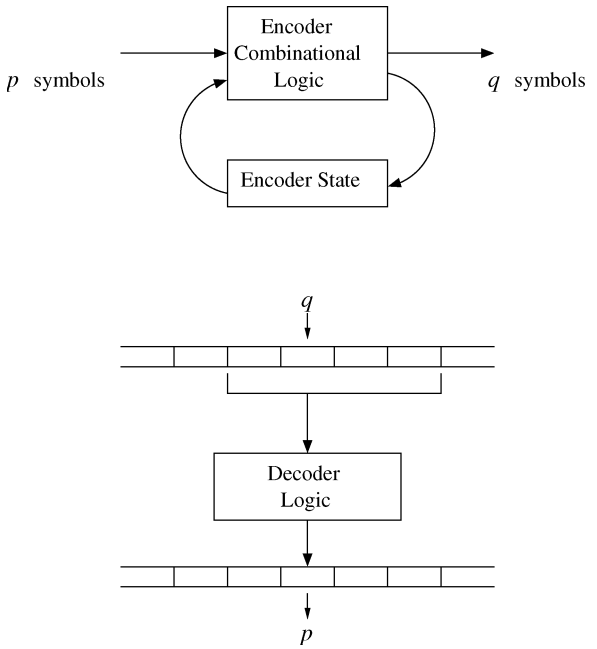

Fig. 3. Schematics of a rate p:q finite-state encoder and a sliding-block decoder.

A *sliding-block decoder* for a rate 1:1 finite-state encoder is a mapping

$$\mathcal{D} : \{0,1\}^{m+a+1} \to \{0,1\}$$

for some integers m, a satisfying $m + a \geq 0$, such that if $\boldsymbol{c} = c_0c_1c_2\ldots$, $c_i \in \{0,1\}$, is any sequence of bits produced by the encoder in response to the input sequence of bits $\boldsymbol{b} = b_0b_1b_2\ldots$, $b_i \in \{0,1\}$, then for $i \geq m$

$$b_i = \mathcal{D}(c_{i-m}c_{i-m+1}\ldots c_i\ldots c_{i+a}).$$

In other words, given a sequence $\boldsymbol{c} = c_0c_1c_2\ldots$ of encoded bits, a sliding-block decoder makes a decision on c_i on the basis of its local context in the sequence $\boldsymbol{c}$. The local context of c_i in this case is a "decoding window" consisting of the bit c_i itself, along with a fixed number m of bits preceding c_i and a fixed number a of bits following c_i.

More generally, a sliding-block decoder for a rate p:q finite-state encoder is a mapping

$$\mathcal{D} : (\{0,1\}^q)^{m+a+1} \to \{0,1\}^p$$

for some integers m, a satisfying $m + a \geq 0$, such that if $\boldsymbol{c} = \boldsymbol{c}_0\boldsymbol{c}_1\boldsymbol{c}_2\ldots$, $\boldsymbol{c}_i \in \{0,1\}^q$, is any sequence of q-bit codewords produced by the encoder in response to the input sequence of p-bit blocks $\boldsymbol{b} = \boldsymbol{b}_0\boldsymbol{b}_1\boldsymbol{b}_2\ldots$, $\boldsymbol{b}_i \in \{0,1\}^p$, then for $i \geq m$

$$\boldsymbol{b}_i = \mathcal{D}(\boldsymbol{c}_{i-m}\boldsymbol{c}_{i-m+1}\ldots\boldsymbol{c}_i\ldots\boldsymbol{c}_{i+a}).$$

The integer m is referred to as the *memory* of the sliding-block decoder, and the integer a is called the *anticipation* of the decoder. A schematic representation of a sliding-block decoder is provided in Fig. 3. Naturally, a finite-state encoder is called *sliding-block decodable* if there exists a sliding-block decoder for the encoder. Not every finite-state encoder may be sliding-block decodable.

In this correspondence, we completely resolve the question of when there exists a rate 1:1, sliding-block decodable, finite-state encoder between (d,k)-constrained systems having the same capacity. Moreover, for each case where such an encoder exists, we explicitly describe the

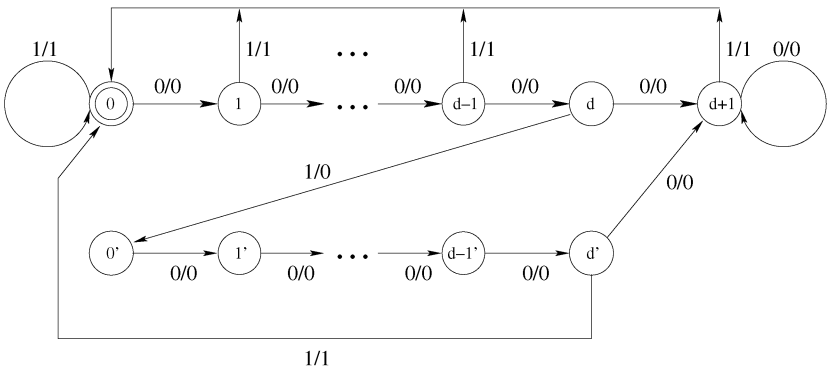

Fig. 4. Encoder from $\mathcal{S}(d, 2d)$ to $\mathcal{S}(d+1, 3d+1)$.

encoder and its sliding-block decoder. Specifically, we prove the following theorem.

Theorem I.1: Let $\mathcal{S}(d,k)$ and $\mathcal{S}(\hat{d},\hat{k})$ be such that $C(d,k) = C(\hat{d},\hat{k})$. Then, there exists a rate 1:1, sliding-block decodable, finite-state encoder from $\mathcal{S}(d,k)$ to $\mathcal{S}(\hat{d},\hat{k})$ if and only if one of the following conditions holds:

1. $(d,k) = (0,1)$ and $(\hat{d},\hat{k}) = (1,\infty)$;
2. $(d,k) = (d,2d)$ and $(\hat{d},\hat{k}) = (d+1,3d+1)$, $d \geq 1$;
3. $(d,k) = (d,\infty)$ and $(\hat{d},\hat{k}) = (d-1,2d-1)$, $d \geq 1$;
4. $(d,k) = (1,2)$ and $(\hat{d},\hat{k}) = (3,7)$.

Note that Conditions 1–4 of the theorem are *not* symmetric with respect to (d,k) and $(\hat{d},\hat{k})$. In fact, the only case where there exists a rate 1:1, sliding-block decodable, finite-state encoder from $\mathcal{S}(d,k)$ to $\mathcal{S}(\hat{d},\hat{k})$, as well as one in the reverse direction from $\mathcal{S}(\hat{d},\hat{k})$ to $\mathcal{S}(d,k)$, is when $\{(d,k),(\hat{d},\hat{k})\} = \{(0,1),(1,\infty)\}$. The existence of these encoders is covered by Conditions 1 and 3, the latter with $d = 1$, in the statement of the theorem.

The proof of the theorem is presented in two parts. In Section II, we explicitly demonstrate rate 1:1 finite-state encoders and sliding-block decoders for all of the indicated cases. In Section III, we use results from the symbolic dynamics literature to prove that no such encoder can be found for any other pair $\mathcal{S}(d,k)$ and $\mathcal{S}(\hat{d},\hat{k})$ with equal capacity.

We would like to remark that as a corollary of a theorem of Ashley [4, Theorem 1.1], one can deduce necessary and sufficient conditions for the existence of a rate 1:1 sliding-block decodable finite-state encoder from one (d,k)-constrained system to another with equal capacity. In principle, these conditions could be used to prove Theorem I.1. Moreover, Ashley's proof of sufficiency of the conditions is constructive and could be used to define a rate 1:1 encoder with the desired properties. However, due to the generality of Ashley's construction, the resulting encoders will be substantially more complex than those presented in Section II. Also, as shown in Section III, the nonexistence results implicit in Theorem I.1 can be proved using more easily verifiable conditions necessary for the existence of the sliding-block mappings.

II. Existence of Encoders

A rate 1:1 finite-state encoder from $\mathcal{S}(0,1)$ onto $\mathcal{S}(1,\infty)$ that is trivially sliding-block decodable is obtained by mapping the symbols 0 and 1 to their respective complements. The same rule also defines a similar encoder from $\mathcal{S}(1,\infty)$ onto $\mathcal{S}(0,1)$.

The graph in Fig. 4 depicts a rate 1:1 finite-state encoder from $\mathcal{S}(d,2d)$ to $\mathcal{S}(d+1,3d+1)$, for $d \geq 1$. The vertices of the graph represent the states of the encoder, and the directed edges represent the state transitions. The vertex labeled 0 (drawn as a double circle in Fig. 4) will be referred to as the *initial state* of the encoder. Note that each edge of the graph is tagged by x/y, where x is the *input label* for that edge, and y is the corresponding *output label*. The encoder transforms (finite or infinite) sequences from $\mathcal{S}(d,2d)$ to sequences in $\mathcal{S}(d+1,3d+1)$ as follows. Suppose that $\boldsymbol{x} = x_0x_1x_2\ldots$ is a sequence in $\mathcal{S}(d,2d)$ that is to be encoded. Starting at the initial state (vertex 0), there is precisely one path whose sequence of input labels is $x_0x_1x_2\ldots$. The sequence $\boldsymbol{y} = y_0y_1y_2\ldots$ of output labels along this path is the output of the encoder corresponding to the input sequence $\boldsymbol{x}$.

We now provide an explanation for why this encoder produces $(d+1,3d+1)$-constrained sequences as its output in response to $(d,2d)$-constrained input sequences. For the sake of notational convenience, we shall use the shorthand "$(1)\,0^j$" to either denote a block of the form 10^j within some sequence, or a block of the form 0^j at the *beginning* of a sequence. Let $\boldsymbol{x}$ be a sequence in $\mathcal{S}(d,2d)$ that is to be encoded. Note that $\boldsymbol{x}$ also belongs to $\mathcal{S}(d+1,3d+1)$ if and only if it does not contain the block 10^d1. From the encoder graph in Fig. 4, it can be seen that the only edge in the graph whose output label is different from its input label is the edge $e_{d0'}$, from state d to state $0'$. This edge is used in the encoding process precisely when the input sequence $\boldsymbol{x}$ contains the block $(1)\,0^d1$ (i.e., $\boldsymbol{x}$ contains 10^d1 or starts with 0^d1). Thus, when the edge $e_{d0'}$ is not used by the encoder, which happens precisely when $\boldsymbol{x}$ does not contain $(1)\,0^d1$, the input sequence is passed through unchanged. But, in this case, since $\boldsymbol{x}$ does not contain 10^d1, it belongs to $\mathcal{S}(d+1,3d+1)$ as well, so that the output of the encoder is indeed a $(d+1,3d+1)$-constrained sequence. On the other hand, when the edge $e_{d0'}$ is used by the encoder, a block of the form $(1)\,0^d1$ is converted to $(1)\,0^{d+1}$. Note that if the block (in $\boldsymbol{x}$) following the transformed block is 0^j1, $d \leq j \leq 2d$, then the segment of the output sequence corresponding to the input segment $(1)\,0^d1\,0^j1$ is $(1)\,0^{d+j+1}1$, which satisfies the $(d+1,3d+1)$-constraint. Thus, if the input to the encoder is a $(d,2d)$-constrained sequence, then the corresponding output is always a $(d+1,3d+1)$-constrained sequence.

It must be pointed out that if the input to the encoder is *not* a $(d,2d)$-constrained sequence, then the output of the encoder may not be a $(d+1,3d+1)$-constrained sequence. For example, the all-zeros input sequence of any length is passed through unchanged by the encoder.

A sliding-block decoder for the above encoder, with memory $d+1$ and anticipation d, is defined via the mapping $\mathcal{D} : \{0,1\}^{2d+2} \to \{0,1\}$ specified by the following rule:

$$\mathcal{D}(y_{i-(d+1)},\ldots,y_i,\ldots,y_{i+d}) = \begin{cases} 1, & \text{if } y_{i-(d+1)} = 1 \text{ and} \\ & y_j = 0, j = i-d,\ldots,i+d \\ y_i, & \text{otherwise.} \end{cases}$$

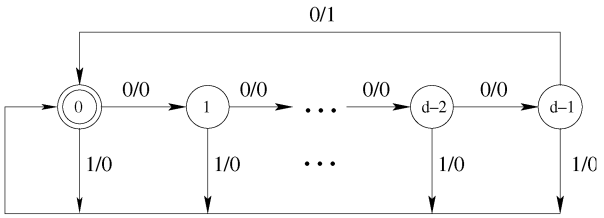

Fig. 5. Encoder from $\mathcal{S}(d,\infty)$ to $\mathcal{S}(d-1,2d-1)$.

Let $y_0y_1y_2\ldots$ be the sequence generated by the encoder in response to the $(d,2d)$-constrained input sequence $x_0x_1x_2\ldots$. To ensure proper retrieval of the entire sequence $x_0x_1x_2\ldots$, the decoder is actually run on the extended sequence $y_{-(d+1)}y_{-d}\ldots y_{-1}y_0y_1y_2\ldots$, where $y_{-(d+1)}y_{-d}\ldots y_{-1}=0^d1$. It is easily verified that the sliding-block decoder defined above reverses any changes made by the encoder, since the decoding rule converts blocks of the form $(1)0^{d+j+1}1$, $d\le j\le 2d$, to $(1)\,0^d\,1\,0^j1$.

Moving on to the next case, a rate 1:1 finite-state encoder from $\mathcal{S}(d,\infty)$ to $\mathcal{S}(d-1,2d-1)$, for $d\ge 1$, is graphically depicted in Fig. 5. The initial state of the encoder is the vertex labeled 0, drawn as a double circle in the figure. The encoding procedure is similar to that described previously for the encoder in Fig. 4.

We now describe how the encoder converts infinite (d,∞)-constrained input sequences to infinite $(d-1,2d-1)$-constrained output sequences. The encoding process for finite sequences is just a finite termination of the process for infinite sequences. Given a binary sequence $\boldsymbol{x}=x_0x_1x_2\ldots$, we use $x_{[i,j)}$, $i<j$, to denote the finite block $x_ix_{i+1}\ldots x_{j-1}$. Also, $x_{[i,\infty)}$ will be used to denote the infinite sequence $x_ix_{i+1}x_{i+2}\ldots$. Using this notation, we define a *maximal* run of zeros in a binary sequence $\boldsymbol{x}$ to be a block $x_{[i,j)}$ (j may be ∞), that consists of zeros alone, such that $x_{i-1}=1$ and, if j is finite, $x_j=1$ as well. To extend this definition naturally to the case of an initial run of zeros, we take x_{-1} to be 1. Thus, a maximal run of zeros is a run of zeros that is not a subblock of a larger run of zeros. Note that in a (d,∞)-constrained sequence, every maximal run of zeros after the first 1 is of length d or more.

From the encoder graph in Fig. 5, it can be seen that the encoder converts each 1 in the input sequence to a 0, and the process of encoding a maximal run of zeros always begins at the initial state (vertex 0). Following paths on the graph from the initial state, we see that in response to a maximal zero run of finite length ℓ, the encoder produces the output sequence $(0^{d-1}1)^r0^s$, where r, s are the unique integers such that $\ell=rd+s$, with $0\le s\le d-1$. Also, the output of the encoder corresponding to a maximal zero run of infinite length is the infinite sequence $(0^{d-1}1)^\infty=0^{d-1}1\,0^{d-1}1\,0^{d-1}1\ldots$. Now, let the input to the encoder be a (d,∞)-constrained sequence of the form $0^{\ell_0}1\,0^{\ell_1}1\,0^{\ell_2}1\ldots$, with $\ell_0\ge 0$ and $d\le\ell_i<\infty$, $i\ge 1$. The corresponding encoder output is the sequence

$$(0^{d-1}1)^{r_0}0^{s_0}\,0\,(0^{d-1}1)^{r_1}0^{s_1}\,0\,(0^{d-1}1)^{r_2}0^{s_2}\ldots$$

where for each $i\ge 0$, r_i, s_i are the unique integers such that $\ell_i=r_id+s_i$, with $0\le s_i\le d-1$. It is easily verified that this output sequence is a $(d-1,2d-1)$-constrained sequence. Finally, if the input (d,∞)-constrained sequence is of the form $0^{\ell_0}1\,0^{\ell_1}1\ldots 0^{\ell_{j-1}}1\,0^{\ell_j}$, for some integer $j\ge 1$, with $\ell_0\ge 0$, $d\le\ell_i<\infty$, $1\le i\le j-1$, and $\ell_j=\infty$, then the encoder output is the $(d-1,2d-1)$-constrained sequence

$$(0^{d-1}1)^{r_0}0^{s_0}\,0\,(0^{d-1}1)^{r_1}0^{s_1}\,0\,\ldots\,(0^{d-1}1)^{r_{j-1}}0^{s_{j-1}}\,(0^{d-1}1)^\infty$$

where r_i,s_i, $0\le i\le j-1$, are defined as before.

The mapping $\mathcal{D}:\{0,1\}^{2d}\to\{0,1\}$ specified by the rule given in what follows, defines a sliding-block decoder for the above encoder, with memory d and anticipation $d-1$

$$\mathcal{D}(y_{i-d},\ldots,y_i,\ldots,y_{i+d-1})=\begin{cases}0, & \text{if } y_{i-d}=y_i=1 \text{ and}\\ & y_j=0,\ i-d+1\le j\le i+d-1, j\ne i\\ y_i, & \text{otherwise.}\end{cases}$$

Let $\boldsymbol{y}=y_0y_1y_2\ldots$ be the sequence generated by the encoder in response to the (d,∞)-constrained input sequence $\boldsymbol{x}=x_0x_1x_2\ldots$. To start decoding, the decoder is given as input the extended sequence $y_{-d}y_{-(d-1)}\ldots y_{-1}y_0y_1y_2\ldots$, where $y_{[-d,0)}=0^{d-1}1$. Let $\hat{\boldsymbol{x}}=\hat{x}_0\hat{x}_1\hat{x}_2\ldots$ be the output produced by the decoder in response to $\boldsymbol{y}$. When $\boldsymbol{x}$ begins with a run of d or more zeros, then it may be verified that $\hat{\boldsymbol{x}}=\boldsymbol{x}$. However, when $\boldsymbol{x}$ starts with 0^j1, with $0\le j\le d-1$, then we indeed have $\hat{x}_i=x_i$ for all $i\ge 0$, *except* for $i=j$, as $x_j=1$, but the decoder produces $\hat{x}_j=0$. Therefore, since the decoder has no prior knowledge of the length of the initial zero run in $\boldsymbol{x}$, it can only be guaranteed that $\hat{x}_i=x_i$ for all $i>d$.

To finish the proof of the sufficiency of Conditions 1–4 of Theorem I.1, it only remains to show the existence of a rate 1:1 finite-state encoder from $\mathcal{S}(1,2)$ to $\mathcal{S}(3,7)$ and the corresponding sliding-block decoder. Let $\mathcal{E}_1$ be the encoder from $\mathcal{S}(1,2)$ to $\mathcal{S}(2,4)$ guaranteed by Condition 2 of the theorem applied with $d=1$, and let $\mathcal{D}_1$ be its sliding-block decoder. Similarly, Condition 2 applied with $d=2$ yields an encoder $\mathcal{E}_2$ from $\mathcal{S}(2,4)$ to $\mathcal{S}(3,7)$ with sliding-block decoder $\mathcal{D}_2$. But now, the composition $\mathcal{E}_2\circ\mathcal{E}_1$ is a rate 1:1, finite-state encoder from $\mathcal{S}(1,2)$ to $\mathcal{S}(3,7)$, which is sliding-block decodable by the decoder defined by $\mathcal{D}_1\circ\mathcal{D}_2$. This completes the proof of the sufficiency of Conditions 1–4 of Theorem I.1.

III. Nonexistence of Encoders

In this section, we prove the converse part of Theorem I.1, i.e., we show that there do not exist rate 1:1, sliding-block decodable, finite-state encoders between $\mathcal{S}(d,k)$ and $\mathcal{S}(\hat{d},\hat{k})$ that have the same capacity but are not covered by Conditions 1–4 of the theorem. The proof relies upon results from symbolic dynamics; the reader is referred to [2] for the relevant background.

As noted earlier, all the equalities possible among the capacities $C(d,k)$ are listed in (4)–(6). Thus, we need to show the nonexistence of rate 1:1, sliding-block decodable encoders in the following cases:

i) from $\mathcal{S}(d+1,3d+1)$ to $\mathcal{S}(d,2d)$, $d\ge 1$;

ii) from $\mathcal{S}(d-1,2d-1)$ to $\mathcal{S}(d,\infty)$, $d>1$;

iii) from $\mathcal{S}(4,\infty)$ to $\mathcal{S}(1,2)$ or $\mathcal{S}(2,4)$.

Observe that a sliding-block decoder for a rate 1:1 finite-state encoder from $\mathcal{S}(d,k)$ to $\mathcal{S}(\hat{d},\hat{k})$ (viewed as constrained systems of finite sequences) can be extended to a sliding-block map from the shift space $\mathcal{S}(\hat{d},\hat{k})$ *onto* the shift space $\mathcal{S}(d,k)$. Now, from an elementary result in symbolic dynamics [2, p. 18, Proposition 1.5.11], we know that, under a sliding-block mapping from $\mathcal{S}(\hat{d},\hat{k})$ to $\mathcal{S}(d,k)$, the image of a bi-infinite sequence $\boldsymbol{y}$ in $\mathcal{S}(\hat{d},\hat{k})$ of period n must be a sequence $\boldsymbol{x}$ in $\mathcal{S}(d,k)$ whose least period divides n. Moreover, from [2, p. 414, Proposition 12.2.3], we can deduce that whenever the shift spaces $\mathcal{S}(d,k)$ and $\mathcal{S}(\hat{d},\hat{k})$ have the same capacity, the existence of a sliding-block mapping from $\mathcal{S}(\hat{d},\hat{k})$ onto $\mathcal{S}(d,k)$ implies that the characteristic polynomial $\chi_{d,k}(z)$ is a divisor of $\chi_{\hat{d},\hat{k}}(z)$ in the ring of integer polynomials. Consequently, there exists a sliding-block decoder for a rate 1:1 encoder from $\mathcal{S}(d,k)$ to $\mathcal{S}(\hat{d},\hat{k})$ only if the periodic sequence condition and the characteristic polynomial condition simultaneously hold.

The existence of such an encoder for the constrained systems listed in cases i) and ii) above is ruled out by the application of the condition

on periodic sequences. In particular, for case i), note that the condition is violated because for any $d \geq 1$, the shift space $\mathcal{S}(d, 2d)$ contains a sequence of period $d+1$, namely, the sequence $(0^d1)^\infty = \cdots 0^d1\,0^d1\,0^d1\ldots$, while $\mathcal{S}(d+1, 3d+1)$ does not contain any sequence of period $d+1$ or less. The periodic sequence condition is violated in case ii) as well, since $\mathcal{S}(d, \infty)$ contains the all-zeros sequence 0^∞, which is periodic with period 1, while $\mathcal{S}(d-1, 2d-1)$, $d > 1$, does not contain the all-zeros sequence or the all-ones sequence, which are the only sequences of period 1. We would like to remark that when $d = 1$, the shift space $\mathcal{S}(d-1, 2d-1)$ does in fact contain the all-ones sequence.

The condition on periodic sequences is not strong enough to handle case iii), since $\mathcal{S}(4, \infty)$ contains the period-1 sequence consisting of all zeros. Instead, we show that there cannot exist a rate 1:1 sliding-block decodable encoder from $\mathcal{S}(4, \infty)$ to either $\mathcal{S}(1, 2)$ or $\mathcal{S}(2, 4)$ by appealing to the characteristic polynomial condition. From (3), we see that

$$\chi_{4,\infty}(z) = z^5 - z^4 - 1$$

and from (2) we find that

$$\chi_{1,2}(z) = z^3 - z - 1$$

and

$$\chi_{2,4}(z) = z^5 - z^2 - z - 1.$$

By inspection, it follows that $\chi_{4,\infty}(z)$ is not a divisor of either $\chi_{1,2}(z)$ or $\chi_{2,4}(z)$ in the ring of integer polynomials. Therefore, there does not exist a sliding-block mapping from either $\mathcal{S}(1, 2)$ or $\mathcal{S}(2, 4)$ onto $\mathcal{S}(4, \infty)$. This completes the proof of Theorem I.1.

We would like to make one final observation regarding the (d, k)-constrained shift spaces $\mathcal{S}(d, k)$. Our proof of Theorem I.1 in fact shows that given distinct shift spaces $\mathcal{S}(d, k)$ and $\mathcal{S}(\hat{d}, \hat{k})$ of equal capacity, there exists a sliding-block map from $S(\hat{d}, \hat{k})$ onto $S(d, k)$ if and only if one of Conditions 1–4 in the statement of the theorem holds. It follows that the only case where there exists a sliding-block map from $\mathcal{S}(\hat{d}, \hat{k})$ onto $\mathcal{S}(d, k)$, and from $\mathcal{S}(d, k)$ onto $\mathcal{S}(\hat{d}, \hat{k})$ as well, is when $\{(d, k), (\hat{d}, \hat{k})\} = \{(0, 1), (1, \infty)\}$. Therefore, aside from $\mathcal{S}(0, 1)$ and $\mathcal{S}(1, \infty)$, no pair of distinct (d, k)-constrained shift spaces can be conjugate. It is a well-known and trivial fact that $\mathcal{S}(0, 1)$ and $\mathcal{S}(1, \infty)$ are indeed conjugate as shift spaces, the required conjugacy being obtained by mapping 0's and 1's to their respective complements. Thus, the only pair of (d, k)-constrained shift spaces that are conjugate are $\mathcal{S}(0, 1)$ and $\mathcal{S}(1, \infty)$.

ACKNOWLEDGMENT

The authors wish to thank Brian Marcus for an especially thorough review of the correspondence, and for suggesting various changes to the presentation in the interests of simplicity and correctness.

REFERENCES

[1] B. H. Marcus, R. M. Roth, and P. H. Siegel, "Constrained systems and coding for recording channels," in *Handbook of Coding Theory*, R. Brualdi, C. Huffman, and V. Pless, Eds. Amsterdam, The Netherlands: Elsevier, 1998.

[2] D. Lind and B. Marcus, *An Introduction to Symbolic Dynamics and Coding*. Cambridge, U.K.: Cambridge Univ. Press, 1995.

[3] K. A. S. Immink, P. H. Siegel, and J. K. Wolf, "Codes for digital recorders," *IEEE Trans. Inform. Theory*, vol. 44, pp. 2260–2299, Oct. 1998.

[4] J. Ashley, "Resolving factor maps for shifts of finite type with equal entropy," *Ergod. Theory Dynam. Syst.*, vol. 11, pp. 219–240, 1991.

[5] N. Kashyap and P. H. Siegel, "Capacity equalities in 1-dimensional (d, k)-constrained systems," in *Proc. IEEE Int. Symp. Information Theory*, Yokohama, Japan, June/July 2003, p. 105.

[6] ——, "Equalities among capacities of (d, k)-constrained systems," *SIAM J. Discr. Math.*, vol. 17, no. 2, pp. 276–297, 2003.

Binary Construction of Quantum Codes of Minimum Distance Three and Four

Ruihu Li and Xueliang Li

***Abstract*—We give elementary recursive constructions of binary self-orthogonal codes with dual distance four for all even lengths $n \geq 12$ and $n = 8$. Consequently, good quantum codes of minimum distance three and four for such length n are obtained via Steane's construction and the CSS construction. Previously, such quantum codes were explicitly constructed only for a sparse set of lengths. Almost all of our quantum codes of minimum distance three are optimal or near optimal, and some of our minimum-distance four quantum codes are better than or comparable with those known before.**

***Index Terms*—Binary code, quantum error correcting code, self-orthogonal code.**

I. INTRODUCTION

Since the initial discovery of quantum error-correcting codes [8], researchers have made great progress in developing quantum codes. Many code constructions are given in [2], [8], [9], [11]. Reference [2] gives a thorough discussion of the principles of quantum coding theory, many example codes, and a tabulation of codes and bounds on the minimum distance for codeword length n up to 30 quantum bits. For larger n there has been less progress, and only a few general code constructions are known, see [1], [2], [4], [5], [10], [11].

In [2, Theorems 10 and 11], Calderbank *et al.* proved that when n is a power of 2 or sums of odd power of 2, there exists a quantum code with parameters $[[n, n-m-2, 3]]$ for certain m, see Theorem 1.2 below. An $[[n, k, d]]$ code is an additive minimum-distance d quantum code of length n encoding k quantum bits [2]. In this correspondence, we use elementary recursive constructions to generalize their result to all even $n \geq 12$ and $n = 8$. Our quantum codes are *additive* and *pure* in the nomenclature of [2], [11]. A pure additive code is *nondegenerate* in the nomenclature of [2], [6]. Using the sphere-packing bound, we show that almost all of our quantum codes of minimum distance three are optimal or near optimal, and some of our quantum codes of minimum distance four are better than or comparable to previously known codes in [1], [2], [5], [11].

Manuscript received May 1, 2003; revised August 16, 2003.

R. Li is with the Department of Applied Mathematics and Physics, College of Art and Science, Air Force Engineering University, Xi'an, Shaanxi 710051, China, and the Department of Computer Science and Engineering, Northwestern Polytechnical University, Xi'an, Shaanxi 710016, China (e-mail: liruihu@yahoo.com.cn).

X. Li is with the Center for Combinatorics, Nankai University, Tianjin 300071, China (e-mail: x.li@eyou.com).

Communicated by E. H. Knill, Associate Editor for Quantum Information Theory.

Digital Object Identifier 10.1109/TIT.2004.828149