Some Families of Zero-Error Block Codes for the Two-User Binary Adder Channel with Feedback

ZHEN ZHANG, TOBY BERGER, FELLOW, IEEE, AND JAMES L. MASSEY, FELLOW, IEEE

Abstract —Families of zero-error codes for the real binary adder channel with feedback that achieve high rate pairs are introduced. Two families of zero-error block codes are given for the case in which only one of the two senders receives feedback about the channel output. In the first of these families, the uninformed sender transmits at a rate of nearly one bit per symbol and the informed sender transmits slightly less that 1/2 bit per symbol. The second family is designed for the case in which the informed sender sends at or near one bit per symbol and the uninformed one sends nearly 1/2 bit per symbol. A family of zero-error codes is introduced, based on the Fibonacci recursion; these codes are readily implemented by means of a simple square-dividing strategy. The Fibonacci codes achieve $R_1 = R_2 = \log_2[(1+\sqrt{5})/2]$ in the limit of large block length. Time-sharing between members of these three code families is used to obtain an achievable rate region, or inner bound, to the zero-error capacity region for block coding. For the case in which the feedback is available to both senders, a variant of the Fibonacci difference equation is used to generate zero-error block codes with slightly higher asymptotic rate $R_1 = R_2 = 0.717$.

I. Introduction

CONSIDER the two-access communication system shown in Fig. 1. Two independent sources wish to send information to the receiver. During a message interval, the messages emanating from the sources are encoded independently with two binary block codes of the same length n. We assume that we have bit and block synchronization.

The two binary input vectors x and y are transformed by the channel into an output vector $z = x + y$, where the plus sign denotes bit-by-bit addition over the reals. This so-called "binary adder channel" is a special case of the multiple access channel. Block coding for this channel has been studied by several authors [1]–[5]. Its Shannon capacity region has been determined, and many results have been obtained about uniquely decodable (i.e., zero-error) codes for it. We shall design some uniquely decodable block codes for cases in which z is fed back either to one or to both senders.

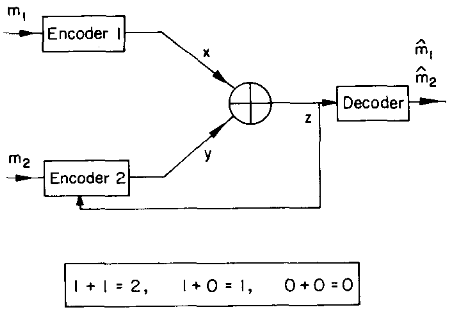

Fig. 1. 2-user binary adder channel with partial feedback.

If both encoding functions depend on the previous channel outputs, we say we have *full feedback*. If only one of them does, we say we have *partial feedback*. We concentrate mainly on the partial feedback case. One reason for this is that in the full feedback case, variable length codes perform significantly better than block codes (cf. [1]).

We now describe the encoding procedure for the two-access channel with partial feedback shown in Fig. 1. The informed encoder's encoding function f_k for the kth time slot depends both on the message m_2 it is trying to send and on the channel outputs during the first $k-1$ time slots. That is, $y_k = f_k(m_2, z_1, \cdots, z_{k-1})$. The uninformed encoder's output x_k during the kth time slot depends only on the message m_1. In the full feedback case we would have $x_k = g_k(m_1, z_1, \cdots, z_{k-1})$.

II. Two Families of Codes for Binary Adder Channel with Partial Feedback

a) The sets W(n,k): For any $x \in \{0,1\}^n$, let

$$t(x) = |\{i: x_i \neq x_{i-1}, i > 1\}|$$

denote the number of transitions in x. Define

$$W(n,k) = \{x \in \{0,1\}^n: t(x) = k\},$$

and note that $|W(n,k)| = \binom{n-1}{k}$.

Manuscript received December 16, 1985; revised September 9, 1986. This paper was presented in part at the Twenty-Third Annual Allerton Conference on Communications, Control, and Computing, Monticello, IL, October 2–4, 1985. This work was supported in part by NSF Grant ECS-8305681.

Z. Zhang is with the Mathematics Department, Bielefeld University, Bielefeld, West Germany.
T. Berger is with the School of Electrical Engineering, 308 Phillips Hall, Cornell University, Ithaca, NY 14853, USA.
J. L. Massey is with the Institute of Signal and Information Processing, Swiss Federal Institute of Technology, ETH-Zentrum, CH-8037 Zurich, Switzerland.

IEEE Log Number 8613574.

0018-9448/87/0900-0613$01.00

b) The ith run $l_i(x)$: The segment $(x_j, \cdots, x_{j+s})$ of x is denoted by $l_i(x)$ and called the ith run if

1) $x_j = \cdots = x_{j+s}$,
2) $x_{j-1} = x_{j+s+1} \neq x_j$,
3) $|\{s: 1 \le s < j,\ x_s \neq x_{s+1}\}| = i - 1$.

c) The first family of codes: The strategy of the first encoder, which receives no feedback, will be simply to transform its message m_1 into a word $x \in W(n,k)$. The second encoder, which is privy to the feedback, first maps its message m_2 into a word $v \in W(s,t)$ which it then sends in n transmissions as follows. Let

$$v = (v_1, \cdots, v_s).$$

Define $f(0) = 0, f(1) = 2$. The second encoder keeps sending v_1 until it receives a feedback $f(v_1)$. Then it keeps sending v_2 until it receives a feedback $f(v_2)$, and so on. If and when it finishes with v, the second encoder keeps sending v_s.

The decoder receives $z = (z_1, \cdots, z_n) \in \{0,1,2\}^n$. Denote the indices of the non-1 components of z by $a_1, a_2, a_3, \cdots$. If the number of the entries of this sequence is s or bigger, then define the second decoding function by

$$\hat{v}(z) = \left(f^{-1}(z_{a_1}), \cdots, f^{-1}(z_{a_s})\right).$$

From $\hat{v}(z)$, the decoder can reconstruct the sequence y transmitted by the second encoder in the manner

$$\hat{y}_j = \begin{cases} \hat{v}_l & a_{l-1} < j \le a_l \\ \hat{v}_s, & j > a_s. \end{cases}$$

The second decoding function is then defined by

$$\hat{x}(z) = z - \hat{y}(z).$$

It is easy to see that a necessary and sufficient condition for this code to be uniquely decodable is that, for any $x \in W(n,k)$ and $v \in W(s,t)$, the length of the a-sequence is at least s. This is because in this case, and only in this case, can the second encoder finish sending v within n slots. Note that the second encoder successfully sends its digit y_i only when y_i agrees with the current digit sent by the first encoder. Thus, v_1 is sent successfully at the latest at the first transition in x. After the ith successful transmission, if $v_{i+1} \neq v_i$, then the second encoder will succeed again at the next transition in the first encoder's sequence; but if $y_{i+1} = y_i$, then the second encoder next succeeds either immediately if $x_{i+1} = x_i$ or at the smallest j such that $x_j \neq x_{i+1}$ if $x_{i+1} \neq x_i$. Thus, a sufficient condition that the a-sequence has length at least s is that the number of transitions in x equals or exceeds one plus the number of transitions t in v plus twice the number $(s-t-1)$ of nontransitions in v. That is, the condition that guarantees unique decodability is

$$1 + t + 2(s - t - 1) = 2s - t - 1 \le k. \qquad (*)$$

d) Rate pairs and rate sum: The rate for the first encoder is

$$R_1 = \frac{1}{n} \log_2 \binom{n-1}{k},$$

and the rate for the second encoder is

$$R_2 = \frac{1}{n} \log_2 \binom{s-1}{t}.$$

For large n, s, and t, let $k/n = p$, $t/s = q$, and $s/n = r$. Then we have

$$R_1 \sim h(p) \qquad R_2 \sim rh(q),$$

where $h(p)$ is the binary entropy function, and (*) becomes

$$2r - rq \le p. \qquad (**)$$

Using equality in (**), we have

$$r = p/(2-q).$$

Therefore, the second rate is

$$R_2 = ph(q)/(2-q).$$

If $p = 1/2$, then $R_1 = 1$, in which case the highest rate for the second encoder under the constraint (**) is $R_2 = 0.347$. The highest rate sum reached by this family of codes is found by equating to zero the derivatives of $R_1 + R_2$ with respect to p and to q. This yields $h'(p) = h'(q)$ which implies that $p = q$. The optimizing p is then seen to satisfy $h(p) + (2-p)h'(p) = 0$, which reduces to $p^2 + p - 1 = 0$, so it is $p^* = (\sqrt{5} - 1)/2$. The resulting maximized rate sum is $-\log(1 - p^*)$, the numerical value of which is

$$\max(R_1 + R_2) = \log_2\left[2/(3-\sqrt{5})\right]$$
$$= 2\log_2\left[(1+\sqrt{5})/2\right] = 1.3885.$$

The rates of this code family are depicted in Fig. 2.

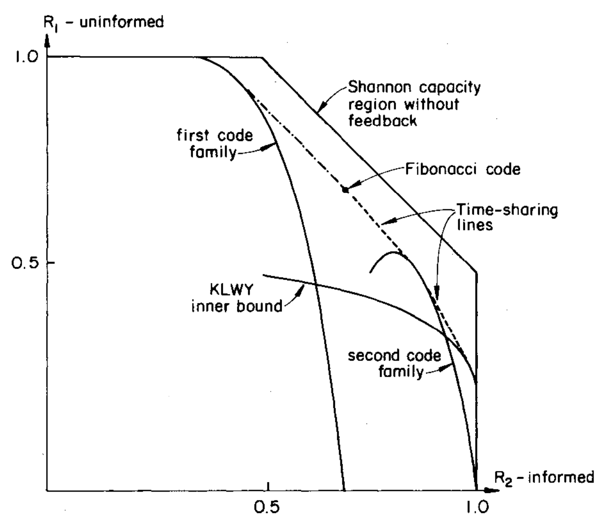

Fig. 2. Rates of code families and inner bound to zero-error capacity for partial feedback case.

e) The second family of codes: For given n and k, we construct a code from the set $W(n,k)$. For any $x \in W(n,k)$, let $|l_i(x)|$ be b_i. Define

$$v(x) = \left(1, \overset{-b_1-}{\cdots} 1,0,0,1, \overset{-b_2-}{\cdots} 1,0,0,1,\cdots\right).$$

This is a binary sequence in $W(n+2k,2k)$ which consists of k runs of 1's, whose lengths are the b_i, separated from one another by pairs of consecutive 0's. The first encoder sends the sequence $v(x)$ for some $x \in W(n,k)$. The second encoder continually uses the feedback to recover the sequence that the first encoder already has sent. The second encoder sends an arbitrary sequence in $\{0,1\}^{n+k}$ into which it inserts a 0 whenever the feedback indicates that the first encoder has just sent the first of a pair of consecutive 0's in the previous slot. The decoder is able to recover the sequences sent by both encoders because it receives 0's only either in isolation or in runs of length 2. It knows that each of the 0-pairs sent by the first encoder ends either at a received isolated 0 or at the end of a received pair of 0's. Thus, the decoder is able to recover the sequence sent by the first encoder. Using that sequence, the decoder can then recover the sequence transmitted by the second encoder and expunge from it the k extra 0's that the second encoder injected.

The rates of this code are

$$R_1 \sim \frac{1}{n+2k}\log_2\binom{n}{k}$$

and

$$R_2 = \frac{n+k}{n+2k}.$$

Letting $k/n = p$, we have for large n that

$$R_1 \sim h(p)/(1+2p)$$

and

$$R_2 = (1+p)/(1+2p).$$

This family of achievable rate pairs also is sketched in Fig. 2. Numerical results show that the best rate sum of any code in this family is 1.375, slightly smaller than the best rate sum of the first code family.

III. Codes Generated by Difference Equations

a) Square dividing strategy: Encoding for the binary adder channel with full or partial feedback can also be described by means of square dividing strategies analogous to those used by Schalkwijk [6] for binary, two-way channels. Assume that the message sets at the two encoders are C_t, $t=1,2$, where $C_1 = \{1,\cdots,a\}$ and $C_2 = \{1,\cdots,b\}$. We consider the set $C_1 \times C_2$. In the first slot for a message in a certain subset of C_1, say $C_1^1(0)$, send 0; for a message in the set $C_1^1(1) = C_1 - C_1^1(0)$, send 1. Similarly, for the second encoder, define $C_2^1(0)$ and $C_2^1(1)$, and send 0 and 1, respectively. Thus, the square $C_1 \times C_2$ is divided into four subsquares with outputs 0, 1, and 2 as shown in Fig. 3.

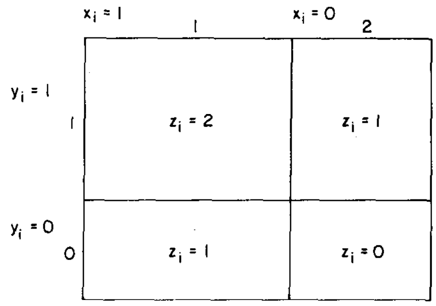

Fig. 3. Square dividing strategy.

Temporarily confine attention to the full feedback case. In this case, after receiving the first feedback, each encoder can identify which of these four subsquares was employed during the first slot. They then divide it into smaller subsquares during the second slot, and into finer and finer subsquares during subsequent slots. The decoder, on the other hand, knows the input is a message pair that belongs to one of the subsquares that is consistent with the sequence of channel outputs observed thus far. The decoder can make the correct decision provided that eventually there is only one message pair that is consistent with the channel output sequence. We shall continue to consider only cases in which no decoding error is allowed. Kasami and Lin [3] call such zero-error codes "uniquely decodable." In the square-dividing terminology, unique decodability means that eventually the square is divided into $a \cdot b$ subsquares, each of which has a unique channel output sequence.

Fig. 4 is an example of a uniquely decodable code with sizes $a=5$ and $b=3$. Note that in this example the first encoder always sends either 111 for message one, 110 for message two, 100 for message three, 001 for message four or 000 for message five. Hence, the first encoder need not be privy to the feedback. In slot 1 the second encoder sends 1 for either message one or message two and sends 0 for message three. In the second slot for either message one or message three, a 1 is sent if the larger of the two possible feedback symbols was received and a 0 is sent if the smaller one was received; the opposite is done for

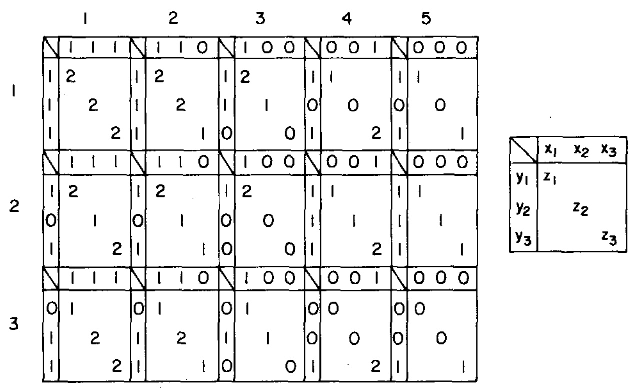

Fig. 4. (3,5) is 3-attainable: $\left(\frac{\log_2 3}{3}, \frac{\log_2 5}{3}\right) = (0.528, 0.774) \in R_0$.

message two. In the third and final slot, a 1 is sent unless the feedback pair from the first two slots indicates that the first encoder is trying to send the third of its five possible messages, in which case a 0 is sent. A unique ternary output sequence, shown on the subsquare diagonals, results for each of the 15 possible message pairs. This is but one of many different zero-error coding strategies that could have been employed in this example. The rationale underlying this particular strategy is explained in subsection b) below in conjunction with the proof of Theorem 1′.

A pair of numbers (a, b) is called *k-attainable* (for the full feedback case or the partial feedback case) if there exists a uniquely decodable code (for the corresponding case) of length k with codeword sets of sizes a and b. Clearly, any pair that is k-attainable for the partial feedback case must be k-attainable for the full feedback case. Of course, if (a, b) is k-attainable, $c \le a$, and $d \le b$, then (c, d) also is k-attainable. As shown by the example, $(5,3)$ is 3-attainable with partial feedback. For small k, it is not difficult to determine whether or not a pair of numbers is k-attainable, but as k grows this task becomes imposing.

We introduce a method to generate families of attainable pairs. We call the codes we use to reach these pairs *codes generated by difference equations*. As their name implies, they have a recursive construction that makes them easy to encode and decode. Because of their high rates and ease of implementability, they are an interesting family of codes.

b) Fibonacci codes: The simplest of our codes generated by difference equations will be called *Fibonacci codes*. Let $\{a_i\}$ be the Fibonacci numbers defined by

$$a_i = a_{i-1} + a_{i-2} \tag{1}$$

with $a_0 = a_1 = 1$. We prove below that (a_i, a_{i+1}) is i-attainable.

A pair of difference equations

$$a_i = \phi(a_j,\ j < i) \tag{2a}$$

$$b_i = \psi(b_j,\ j < i) \tag{2b}$$

will be called a pair of *code generating equations* for some initial conditions and a positive integer S if the sequences $\{a_i\}$ and $\{b_i\}$ generated by these equations and initial conditions have the property that (a_i, b_i) is $(S+i)$-attainable for all $i \ge 1$. Thus, we have claimed that the pair of Fibonacci equations $a_i = a_{i-1} + a_{i-2}$ and $b_i = b_{i-1} + b_{i-2}$ are code generating equations for $a_0 = a_1 = b_0 = 1$, $b_1 = 2$, and $S = 0$.

It is easy to find code generating equations, but at present we have no general way of finding ones that possess high rates. We shall show, however, that the aforementioned Fibonacci codes and another set of code generating equations we introduce in Section IV do indeed achieve high rates.

We now prove our claim that a pair of Fibonacci equations are code generating for the full feedback case. Subsequently, we extend this result to the partial feedback case. To facilitate the proof, we introduce the concept of an *attainable cluster*. The union of all subsquares that share the same output sequence is called a cluster. For example, in the $(5,3)$ code of Fig. 4, after the first step, the 2×2 rectangle in the upper right corner and the 1×3 rectangle in the lower left corner together constitute a cluster. A cluster is k-attainable if after k or fewer further divisions, it can be reduced to single "points" each of which has a distinct output sequence. The cluster comprised of the aforementioned 2×2 and 1×3 rectangles is 2-attainable. These two rectangles are *input-disjoint* in the sense that the user inputs can be chosen independently for these two rectangles. It should be obvious that a cluster composed of two input-disjoint rectangles of sizes 1×2 and 1×1 is 1-attainable.

Theorem 1: A pair of Fibonacci equations with $a_0 = a_1 = b_0 = 1$, $b_1 = 2$, and $S = 0$ are code generating for full feedback.

Proof: First, we define two types of parameterized clusters and prove that, by one step of square dividing, each of them can be reduced to clusters of the same two types with smaller parameter values. The first cluster type is a union of two input-disjoint rectangles with sizes $a_k \times b_{k-1}$ and $a_{k-1} \times b_k$; the second is a rectangle with size $a_k \times b_k$. We denote them, respectively, by

$$\lambda_k = a_k \times b_{k-1} \cup a_{k-1} \times b_k \tag{3}$$

and

$$\mu_k = a_k \times b_k, \tag{4}$$

where $a \times b$ denotes an a by b rectangle and $\cup$ denotes the union of input-disjoint rectangles. Note that we can choose the next input digit for the two users so as to divide λ_k into three parts,

$$\lambda_k = [\mu_{k-1}]_2 \cup [\lambda_{k-1}]_1 \cup [\mu_{k-1}]_0, \tag{5}$$

where $[\mu_{k-1}]_2$ means that the set with output 2 is μ_{k-1}, and so on.

For μ_k, we can similarly choose the next input digit so that

$$\mu_k = [\mu_{k-1}]_2 \cup [\lambda_{k-1}]_1 \cup [\mu_{k-2}]_0. \tag{6}$$

Since the 1-attainability of both $\lambda_1 = 1\times 1 \cup 1 \times 2$ and $\mu_1 = 1 \times 2$ are obvious, the theorem is proved.

The limiting rates of the Fibonacci code family are

$$R_1 = R_2 = R_f = \lim_{k\to\infty} \frac{1}{k} \log_2 a_k$$
$$= \log_2 \left[(\sqrt{5}+1)/2 \right] = 0.694.$$

This rate point is shown in Fig. 2.

Now we show that the Fibonacci codes actually are implementable in the partial feedback case.

Theorem 1′: A pair of Fibonacci equations with $a_0 = a_1 = b_0 = 1$, $b_1 = 2$, and $S = 0$ are code generating for partial feedback.

Proof: We need to prove that the Fibonacci encoding strategy can be implemented with one of the two encoders not having access to the feedback. That is, we must exhibit a technique by means of which the uninformed encoder can correctly divide each of the clusters that appears in the square dividing procedure into 1-subsets and 0-subsets. Note, as shown in Fig. 5, that the sizes of the horizontal edges of the subsquares after the successive square divisions are:

originally (Fig. 5a): a_k
after one division (Fig. 5b): a_{k-1}, a_{k-2}
after two divisions (Fig. 5c): $a_{k-2}, a_{k-3}, a_{k-2}$
after three divisions (Fig. 5d): $a_{k-3}, a_{k-4}, a_{k-3}, a_{k-3}, a_{k-4}$
after four divisions: $a_{k-4}, a_{k-5}, a_{k-4}, a_{k-4}, a_{k-5}, a_{k-4}, a_{k-5}, a_{k-4}$

and so on. Observe that, at the ith step, each of the sizes in question is either a_{k-i} or a_{k-i-1}.

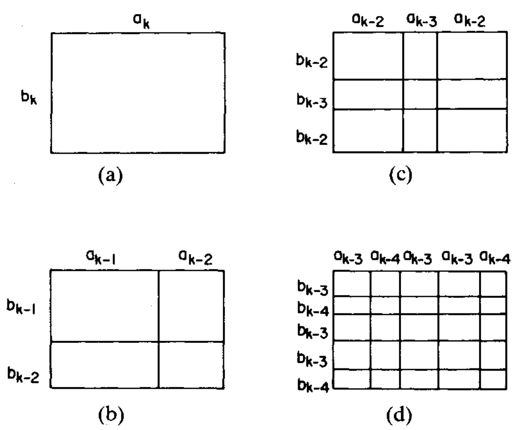

Fig. 5. (a) Original square. (b) First step of square dividing. (c) Second step of square dividing. (d) Third step of square dividing.

Define $u_i = 1$ if the message to be sent by the encoder without feedback is a member of a subset of size a_{k-i}; otherwise, define $u_i = 0$. The strategy of the encoder without feedback is to send the product $x_i = u_i u_{i-1}$ at the ith step. This strategy is depicted in Fig. 6(a) and (b), the former of which shows the subset sizes arranged on successive levels of a tree and the latter of which shows the corresponding binary transmissions.

Analogously define the binary function v_i of the sizes of the vertical squares at the ith step by $v_i = 1$ for subsets of size b_{k-i} and $v_i = 0$ for subsets of size $b_{k-i} - 1$. The strategy of the encoder with feedback is to send $y_i = v_i \oplus 1 \oplus u_{i-1}$ at the ith step, which can be done using the past feedback to deduce the value of x_{i-1}, and hence, recursively, the value of u_{i-1}.

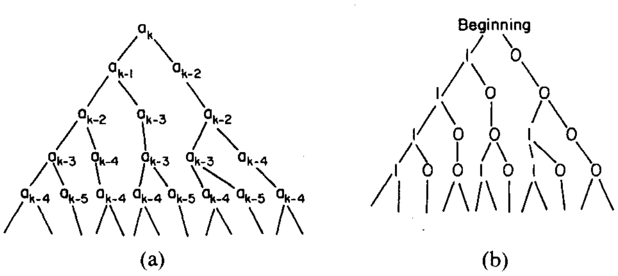

Fig. 6. (a) Square dividing tree. (b) Encoding tree of encoder without feedback.

Now we prove by induction that these two encoding algorithms achieve the same square dividing strategy we described in the full feedback case. At the first step, this is obvious. Generally, we need to prove that, for the two clusters studied in the proof of Theorem 1, the new strategies give precisely the desired dividing. In the case of the first cluster of size $a_k \times b_k$, the two encoders are both sending 1's for the bigger subsets of sizes a_{k-1} and b_{k-1}, respectively, and 0's for the smaller ones of sizes a_{k-2} and b_{k-2}, respectively. It is easy to check that the resulting outputs, shown in Fig. 7(a), are precisely the ones we need in the proof of Theorem 1. For the second cluster, $a_{k-1} \times b_{k-2} \cup a_{k-2} \times b_{k-1}$, the channel inputs calculated by the two encoders in accordance with the above prescriptions are shown in Fig. 7(b); note that the resulting outputs again exactly satisfy the requirements of the proof of Theorem 1. The next step, shown in Fig. 7(c), has the (5,3)-code of Fig. 4 embedded within it. We omit the general step in the induction argument because its validity should be apparent by now.

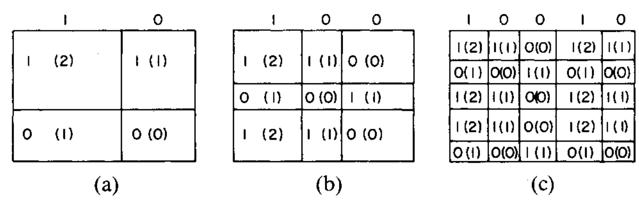

Fig. 7. Encoding strategy of encoder with feedback.

c) The inner bound to the zero-error capacity region: The three families of zero-error codes we have introduced can be combined by tangent lines representing time-sharing to produce an inner bound to the zero-error capacity region of the binary adder channel with partial feedback. This bound can be mildly improved in the low rate region for the uninformed encoder by appealing to an inner bound to the zero-error capacity region derived by Kasami *et al.* [4] for the case in which there is no feedback to either encoder; clearly, any inner bound for that case is an inner bound for the partial feedback case. That bound and a time-sharing line joining it to the second of our code families completes our overall inner bound depicted in Fig. 2. The straight-line portion of this bound, between the curve of our first code family and the Fibonacci code, has slope -1 and a rate sum of 1.3885.

IV. Codes Generated by Difference Equations for the Binary Adder Channel with Full Feedback

a) Refinement of the Fibonacci code: We call a k-attainable pair (a, b) optimal if $(a+1, b)$ and $(a, b+1)$ are no longer k-attainable. Consider the first few Fibonacci code sizes:

$$(1,2),(2,3),(3,5),(5,8),(8,13),(13,21),\cdots.$$

It is not hard to prove that the first three terms are optimal for $k=1$, 2, and 3, respectively. It turns out, however, that (5,9) is 4-attainable and (8,14) is 5-attainable. This suggests that there may exist code generating equations that generate codes with asymptotically equal rates greater than R_f. We proceed to show that this is indeed the case.

Theorem 2: With $a_0=a_1=b_0=1$, $b_1=2$, and $S=0$, the following are code generating equations:

$$a_k=a_{k-1}+a_{k-2}+5a_{k-11} \tag{7}$$

$$b_k=b_{k-1}+b_{k-2}+5b_{k-11}. \tag{8}$$

We prove this theorem in the Appendix. The $\{a_k\}$ and $\{b_k\}$ of Theorem 2 give a limiting rate pair of (0.717, 0.717) which dominates that of the Fibonacci codes. We refer to the associated codes as *refined Fibonacci codes*. We have not yet ascertained whether or not (7) or (8) are code generating for the partial feedback case as well.

We have also been able to produce code generating equations for the full feedback case with nonequal asymptotic rates. However, since their rate pairs lie strictly within the lower bound to the capacity region described in part c) of this section, we have chosen not to discuss them here.

b) Inner bound for the zero-error capacity region of a binary adder channel with full feedback: The convex hull of our first family of codes for the partial feedback case is an inner bound for the zero-error capacity region for the full feedback case. (The mirror image of the performance of the first family of codes dominates the performance of the second family of codes. Since either encoder one or encoder two could choose to ignore its feedback, we get a better bound for the full feedback case by using only the first code family.) An additional improvement is obtained by incorporating the point (0.717, 0.717), corresponding to the refined Fibonacci code, and then re-taking the convex hull. The resulting inner bound is shown in Fig. 8.

Recently, Dueck [7] has derived the exact form of the zero-error full feedback capacity region for a certain class

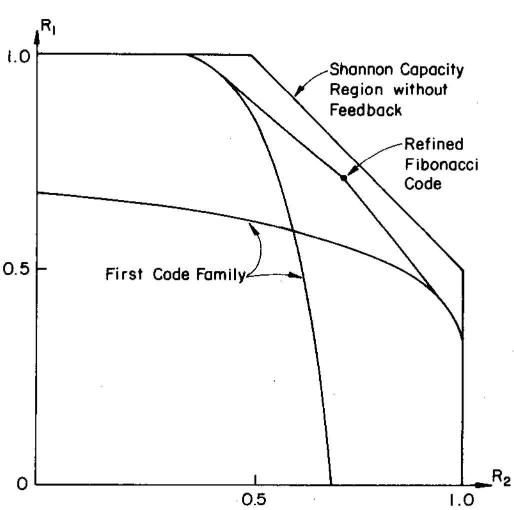

Fig. 8. Rates of first code family, extended Fibonacci code, and inner bound to zero-error capacity for full feedback case.

of multiple access channels to which our full feedback case belongs. However, numerical evaluation of his capacity region description is fraught with challenging obstacles even in this special case so that our inner bound of Fig. 8 is still of some interest.

Appendix

Proof of Theorem 2 via Three Lemmata.

Lemma 1: (5,45) is a 6-attainable pair.

To prove this lemma requires checking the square dividing procedure step by step. We omit this tedious but straightforward task.

Lemma 2: There exists K such that for $k>K$,

$$a_k/90 \le a_{k-9} \tag{9a}$$

$$b_k/90 \le b_{k-9}. \tag{9b}$$

Proof of Lemma 2: We need to prove only that, if C is the largest eigenvalue of the characteristic equation of the difference equation (7), then

$$C^9/90-1 \le 0. \tag{10}$$

This is readily verified by calculation.

Lemma 3: The following clusters are k-attainable:

$$\lambda_k=(a_{k+1}-a_k)\times b_k \cup a_k \times 5b_{k-9} \cup a_k \times (b_{k+1}-b_k) \cup 5a_{k-1}\times b_k$$

$$\beta_k=a_x \times b_k \cup a_{k+1}\times 5b_{k-8} \cup (a_{k-1}+5a_{k-10})\times 5b_{k-9}$$

$$\delta_k=(a_{k+1}-a_k)\times(b_{k+1}-b_k)\cup a_{k+2}\times 5b_{k-7}$$

$$\rho_k=5b_{k-6}\times \frac{a_{k+3}}{2} \cup 5a_{k-9}\times b_k \cup a_k \times 5b_{k-9}$$

$$\pi_k=(a_{k+1}-a_k)\times(b_{k+1}-b_k) \cup 2a_{k-2}\times b_{k-2}\cup a_{k-2}\times 2b_{k-2}$$

$$\tau_k=2a_{k-1}\times b_{k-1}\cup a_k \times 5b_{k-1}\cup 5a_{k-9}\times b_k.$$

It is obvious that Theorem 2 is a consequence of Lemma 3.

Proof of Lemma 3: We prove the following recursive inequalities, in which $'$ denotes a cluster with the roles of a and b exchanged and $\le$ means that the parts after a square dividing are subsets of the sets listed on the right side.

$$\lambda_k \le [\beta_{k-1}]_0 \cup [\lambda_{k-1}]_1 \cup [\beta'_{k-1}]_2 \tag{11}$$

$$\beta_k \le [\beta_{k-1}]_0 \cup [\lambda_{k-1}]_1 \cup [\delta_{k-1}]_2 \tag{12}$$

$$\delta_k \le [\beta_{k-1}]_0 \cup [\rho_{k-1}]_1 \cup [\rho_{k-1}]_2 \tag{13}$$

$$\tau_k \le [\beta_{k-1}]_0 \cup [\beta'_{k-1}]_1 \tag{14}$$

$$\pi_k \le [\beta_{k-1}]_0 \cup [\tau_{k-1}]_1 \cup [\tau'_{k-1}]_2 \tag{15}$$

$$\rho_k \le (5,45)\otimes\{[\beta_{k-7}]_0 \cup [\lambda_{k-7}]_1 \cup [\pi_{k-7}]_2\}, \tag{16}$$

where the operator "$(\alpha,\beta)\otimes$" multiplies the row and column cardinalities of each code in the succeeding curly bracket by α and by β, respectively. The lemma follows from these inequalities.

Proof of (11):

$$\lambda_k = [a_{k-1} \times b_{k-1} \cup a_k \times 5b_{k-9} \cup (a_k - a_{k-1}) \times 5b_{k-10}]_0$$
$$\cup [a_{k-1} \times (b_k - b_{k-1}) \cup 5a_{k-10} \times b_{k-1} \cup a_{k-1} \times 5b_{k-10} \cup (a_k - a_{k-1}) \times b_{k-1}]_1$$
$$\cup [a_{k-1} \times b_{k-1} \cup 5a_{k-9} \times b_k \cup (b_k - b_{k-1}) \times 5a_{k-10}]_2$$
$$= [\beta_{k-1}]_0 \cup [\lambda_{k-1}]_1 \cup [\beta'_{k-1}]_2.$$

Proof of (12):
Since

$$a_k - a_{k-1} > a_{k-2} \geq 2a_{k-4} \geq 4a_{k-6} \geq 8a_{k-8} > 5a_{k-10},$$

we have $a_{k-1} + 5a_{k-10} \leq a_k$, so

$$\beta = [a_{k-1} \times b_{k-1} \cup (a_k + 5a_{k-10}) \times 5b_{k-9}]_0$$
$$\cup [a_{k-1} \times (b_k - b_{k-1}) \cup (a_k - a_{k-1}) \times b_{k-1}]_1$$
$$\cup [(a_k - a_{k-1}) \times (b_k - b_{k-1}) \cup a_{k+1} \times 5b_{k-8}]_2$$
$$\leq [\beta_{k-1}]_0 \cup [\lambda_{k-1}]_1 \cup [\delta_{k-1}]_2.$$

Proof of (13):
Since

$$b_{k-1} \geq 2b_{k-3} \geq 4b_{k-5} \geq 8b_{k-7} > 5b_{k-10},$$

we have

$$\delta_k = [a_{k-1} \times b_{k-1}]_0 \cup \left[a_{k-1} \times 5b_{k-1} \cup b_{k-1} \times 5a_{k-10} \cup \frac{a_{k+2}}{2} \times 5b_{k-7}\right]_1$$
$$\cup \left[5b_{k-10} \times 5a_{k-10} \cup \frac{a_{k+2}}{2} \times 5b_{k-7}\right]_2$$
$$\leq [\beta_{k-1}]_0 \cup [\rho_{k-1}]_1 \cup [\rho_{k-1}]_2.$$

Proof of (14):

$$\tau_k = [a_{k-1} \times b_{k-1} \cup a_k \times 5b_{k-9}]_0$$
$$\cup [a_{k-1} \times b_{k-1} \cup b_k \times 5a_{k-9}]_1$$
$$\leq [\beta_{k-1}]_0 \cup [\beta_{k-1'}]_1.$$

Proof of (15):

$$\pi_k = [a_{k-1} \times b_{k-1}]_0 \cup [a_{k-1} \times 5b_{k-10} \cup 2a_{k-2} \times b_{k-2} \cup b_{k-1} \times 5a_{k-10}]_1$$
$$\cup [5a_{k-10} \times 5b_{k-10} \cup a_{k-2} \times 2b_{k-2}]_2$$
$$\leq [\beta_{k-1}]_0 \cup [\tau_{k-1}]_1 \cup [\tau_{k-1'}]_2.$$

Proof of (16):

$$\rho_k \leq (5,45) \otimes \{b_{k-6} \times a_{k-6} \cup a_{k-9} \times 2b_{k-9} \cup 2a_{k-9} \times b_{k-9}\}$$
$$\leq (5,45) \otimes \{[b_{k-7} \times a_{k-7}]_0 \cup [a_{k-7} \times (b_{k-6} - b_{k-7}) \cup (a_{k-6} - a_{k-7}) \times b_{k-7}]_1$$
$$\cup [(a_{k-6} - a_{k-7}) \times (b_{k-6} - b_{k-7}) \cup a_{k-9} \times 2b_{k-1} \cup 2a_{k-9} \times b_{k-9}]_2\}$$
$$\leq (5,45) \otimes \{[\beta_{k-7}]_0 \cup [\lambda_{k-7}]_1 \cup [\pi_{k-7}]_2\}.$$

Theorem 2 gives a limiting rate pair (0.717, 0.717), which dominates that of the Fibonacci codes.

References

[1] T. S. Han and H. Sato, "On the zero-error capacity region by variable length codes for multiple channel with feedback," preprint.

[2] R. Ahlswede, "Multi-way communication channels," in *Proc. 2nd Int. Symp. Inform. Theory*, Tsahkadsor, Armenian S. S. R., 1971, pp. 23–52, Hungarian Academy of Sciences, 1973.

[3] T. Kasami and L. Lin, "Coding for a multiple-access channel," *IEEE Trans. Inform. Theory*, vol. IT-22, pp. 129–137, Mar. 1976.

[4] T. Kasami, S. Lin, V. K. Wei, and S. Yamamura, "Graph theoretic approaches to the code construction of the two-user multiple-access binary adder channel," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 114–130, Jan. 1983.

[5] H. C. A. van Tilborg, "Upper bounds on $|C_2|$ for a uniquely decodable code pair (C_1, C_2) for a two-access binary adder channel," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 386–389, May 1983.

[6] J. P. M. Schalkwijk, "On an extension of an achievable rate region for the binary multiplying channel," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 445–448, May 1983.

[7] G. Dueck, "The zero-error feedback capacity region of a certain class of multiple access channels," *Problems of Control and Information Theory*, vol. 14, no. 2, pp. 89–103, 1985.