LUND UNIVERSITY

Optimal and near-optimal encoders for short and moderate-length tailbiting trellises

Ståhl, Per; Anderson, John B; Johannesson, Rolf

Published in:
IEEE Transactions on Information Theory

DOI:
10.1109/18.796408

1999

Link to publication

Citation for published version (APA):
Ståhl, P., Anderson, J. B., & Johannesson, R. (1999). Optimal and near-optimal encoders for short and moderate-length tailbiting trellises. *IEEE Transactions on Information Theory*, *45*(7), 2562-2571. https://doi.org/10.1109/18.796408

Total number of authors:
3

General rights
Unless other specific re-use rights are stated the following general rights apply:
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.
• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal

Read more about Creative commons licenses: https://creativecommons.org/licenses/

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

LUND UNIVERSITY

PO Box 117
221 00 Lund
+46 46-222 00 00

Download date: 09. May. 2024

[5] D. J. Muder, "Minimal trellises for block codes," *IEEE Trans. Inform. Theory*, vol. 34, pp. 1049–1053, Sept. 1988.
[6] J. B. Anderson and S. M. Hladik, "tail-biting MAP decoders," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 297–302, Feb. 1998.
[7] A. R. Calderbank, G. D. Forney, Jr., and A. Vardy, "Minimal tail-biting trellises: The Golay code and more," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1435–1455, July 1999.
[8] V. Pless, "Decoding the Golay codes," *IEEE Trans. Inform. Theory*, vol. IT-32, pp. 561–576, July 1986.
[9] A. Vardy and Y. Be'ery, "More efficient soft decoding of the Golay codes," *IEEE Trans. Inform. Theory*, vol. 37, pp. 667–672, May 1991.
[10] J. H. Conway and N. J. A. Sloane, "Soft decoding techniques for codes and lattices, including the Golay code and the Leech lattice," *IEEE Trans. Inform. Theory*, vol. IT-32, pp. 41–50, Jan. 1986.
[11] J. Snyders and Y. Be'ery, "Maximum likelihood soft decoding of binary block codes and decoders for the Golay codes," *IEEE Trans. Inform. Theory*, vol. 35, pp. 963–975, Sept. 1989.
[12] O. Amrani, Y. Be'ery, A. Vardy, F. W. Sun, and C. A. van Tilborg, "The Leech lattice and Golay code: Bounded distance decoding and multilevel construction," *IEEE Trans. Inform. Theory*, vol. 40, pp. 1030–1043, July 1994.
[13] F. M. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, vols. I/II. Amsterdam, The Netherlands: North-Holland, 1977.
[14] P. Robertson, "Illuminating the structure of code and decoder of parallel concatenated recursive systematic (turbo) codes," in *Proc. GLOBECOM'94*, Dec. 1994, pp. 1298–1303.
[15] N. Wiberg, H. A. Loeliger, and R. Kotter, "Codes and iterative decoding on general graphs," *Euro. Trans. Telecommun.*, vol. 6, pp. 513–526, Sept. 1995.
[16] Li Ping, S. Chan, and K. L. Yeung, "Max-log-MAP filtering algorithm for decoding product F_{24} code," in *Proc. IEEE 1997 Int. Conf. Communications*, June 1997, pp. 1366–1370.
[17] Li Ping and S. Chan, "Iterative decoding of concatenated Hadamard codes," in *Proc. IEEE 1998 Int. Conf. Communications*, June 1998, pp. 136–140.

Optimal and Near-Optimal Encoders for Short and Moderate-Length Tail-Biting Trellises

Per Ståhl, *Student Member, IEEE*, John B. Anderson, *Fellow, IEEE*, and Rolf Johannesson, *Fellow, IEEE*

***Abstract*—The results of an extensive search for short and moderate-length polynomial convolutional encoders for time-invariant tail-biting representations of block codes at rates $R = 1/4, 1/3, 1/2,$ and $2/3$ are reported. The tail-biting representations found are typically as good as the best known block codes.**

***Index Terms*—Convolutional codes, error-correction coding, tail-biting encoders, trellis codes.**

I. Introduction

There exist two main principles to terminate convolutional codes into block codes. Assume for simplicity that the generator matrix for a rate $R = b/c$ convolutional code of memory m is polynomial and realized in controller canonical form.

Manuscript received February 1, 1998; revised February 19, 1999. This work was supported in part by the Swedish Research Council for Engineering Sciences under the "Visiting Professors Programme" 1996/97, and in part by the Foundation for Strategic Research–Personal Computing and Communication under Grant PCC-9706–09.

The authors are with the Department of Information Technology, Information Theory Group, Lund University, S-221 00 Lund, Sweden.

Communicated by F. R. Kschischang, Associate Editor for Coding Theory. Publisher Item Identifier S 0018-9448(99)07009-1.

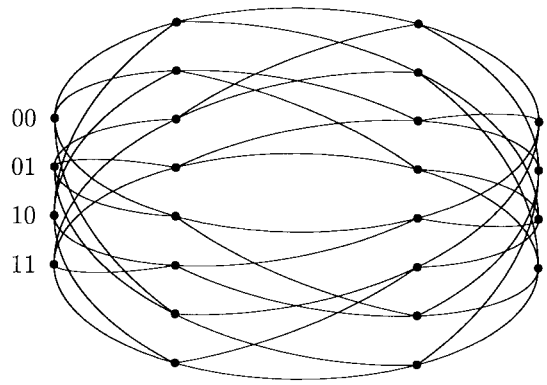

Fig. 1. Circular trellis of length $L = 6$ for a four-state convolutional encoder.

In the first method we start in the zero state and encode the K information symbols followed by m b-tuples of zeros. Hence, we reach the zero state and the convolutional code has been *terminated* into a block code by the so-called *zero-tail* (ZT) method at the cost of a rate loss by a factor $K/(K + mb)$. If the trellis is short this rate loss might not be acceptable.

A termination method that does not suffer from any rate loss is *tail-biting*, which can be used to construct very powerful regular trellis representations of block codes. Assuming a trellis of L sections (for simplicity we assume here that $m \leq L$), the *tail-biting condition* is the restriction that the convolutional encoder state $\boldsymbol{\sigma}$ at time $t = 0$ is equal to the encoder state at time L, i.e., $\boldsymbol{\sigma}_0 = \boldsymbol{\sigma}_L$. A tail-biting trellis of length L corresponds to a total of $K = bL$ information symbols, c symbols per branch, block length $N = Lc$, 2^b branches per trellis node; the number of codewords is

$$M = 2^K = 2^{bL} \tag{1}$$

and its rate is

$$R = K/N = b/c. \tag{2}$$

Let $\boldsymbol{u}_{[0,L)} = \boldsymbol{u}_0\boldsymbol{u}_1 \cdots \boldsymbol{u}_{L-1}$ denote the input (information) sequence, $\boldsymbol{v}_{[0,L)} = \boldsymbol{v}_0\boldsymbol{v}_1 \cdots \boldsymbol{v}_{L-1}$ the output sequence (codeword), and

$$G(D) = G_0 + G_1 D + \cdots + G_m D^m \tag{3}$$

the generator matrix. The codewords of the tail-biting representation of the block code $\mathcal{B}_{\rm tb}$ that is obtained from the convolutional code $\mathcal{C}$ encoded by the generator matrix $G(D)$ can be compactly written as

$$\boldsymbol{v}_{[0,L)} = \boldsymbol{u}_{[0,L)}\boldsymbol{G}_L^{\rm tb} \tag{4}$$

where

$$\boldsymbol{G}_L^{\rm tb} = \begin{pmatrix} G_0 & G_1 & \cdots & & G_m & & & \\ & G_0 & G_1 & \cdots & & G_m & & \\ & & \ddots & \ddots & & & \ddots & \\ & & & G_0 & G_1 & \cdots & & G_m \\ G_m & & & & G_0 & G_1 & \cdots & G_{m-1} \\ G_{m-1} & G_m & & & & \ddots & \ddots & \vdots \\ \vdots & & \ddots & & & & \ddots & G_1 \\ G_1 & G_2 & \cdots & G_m & & & & G_0 \end{pmatrix} \tag{5}$$

0018–9448/99$10.00 © 1999 IEEE

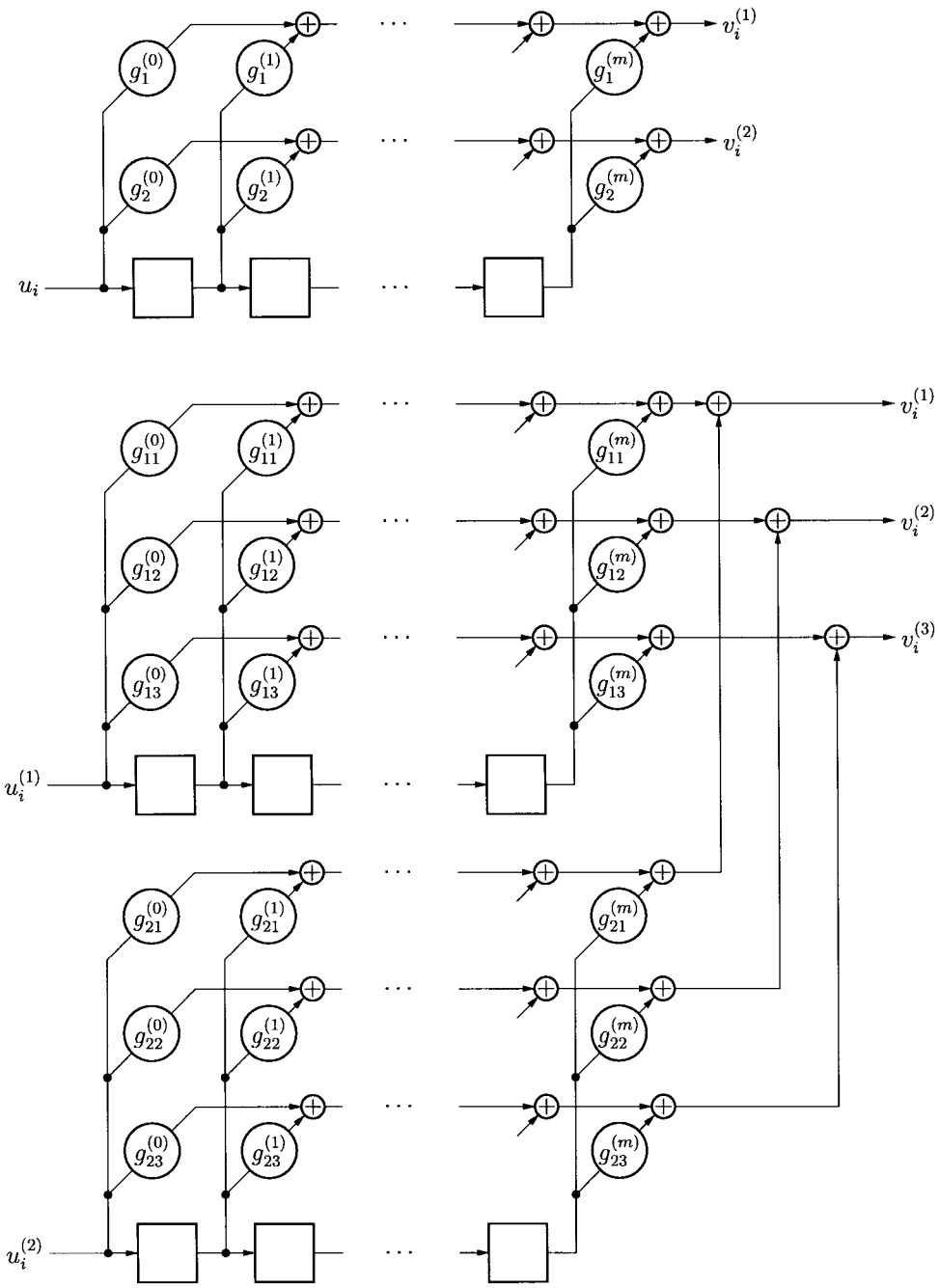

Fig. 2. Polynomial rate $R = 1/2$ and $R = 2/3$ convolutional encoders.

is the $L \times L$ generator matrix for the tail-biting representation of the block code $\mathcal{B}_{\rm tb}$. The initial state is uniquely determined by the last m input b-tuples.

Since we require that we have the same state at the beginning as at the end we can use a *circular trellis* for the tail-biting representation of a block code. In Fig. 1 we show as an example a circular trellis of $L = 6$ sections for a four-state tail-biting representation.

The block code is the union of 2^{bm} subsets corresponding to the paths that go through each of the 2^{bm} states at time 0. In fact, these are 2^{bm} cosets of the $(N, K - bm)$ zero-tail terminated convolutional code corresponding to the paths that go through the all-zero state at time 0.

Tail-biting is particularly valuable in applications such as packet transmission, where codewords are apt to be short. Being trellis methods, tail-biting decoders easily accept soft-channel output. Finally, as we show here, time-invariant tail-biting encoders can generate many of the most powerful binary block codes.

Tail-biting (TB) representations of block codes were introduced by Solomon and van Tilborg [1]. An early paper that explains tail-biting in detail is [2]. In [3], a two-way decoding algorithm for exact *a posteriori* probability (APP) decoding of TB trellises is presented. The complexity of this algorithm is of the order 2^{2bm}, and performance is governed by the true minimum distance d of the code.

More commonly, an approximate APP decoding algorithm such as that given in [4] is used (the one in [4] applies the BCJR recursions to the TB trellis). Such algorithms dramatically reduce the complexity to the order of 2^{bm}; moreover, they achieve near-

TABLE I
THE BEST RATE $R = 1/4$ CODES FOUND FOR GIVEN K, $2 \leq K \leq 50$, EXHAUSTIVE SEARCH

m	1			2			3			4			5		
K	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d
2	(4,4,4,6)	5	2	(4,4,6,7)	5	2	(44,54,64,70)	5	2	(46,52,60,70)	5	2	(46,56,67,73)	5	2
3	(4,4,6,6)	6	4	(4,4,6,7)	6	3	(40,54,60,70)	6	3	(42,54,62,70)	6	3	(43,51,63,76)	6	3
4	(4,4,6,6)	6	4	(4,5,6,7)	8	11	(40,50,60,64)	8	11	(46,56,66,76)	8	11	(45,56,66,73)	8	11
5	(4,4,6,6)	6	5	(4,5,6,7)	8	5	(40,54,60,70)	9	10	(40,52,62,74)	9	10	(45,57,64,72)	9	10
6	(4,6,6,6)	6	1	(5,6,7,7)	8	3	(40,54,64,70)	10	12	(46,56,62,76)	10	9	(45,57,64,72)	10	9
7	(4,6,6,6)	7	8	(5,6,7,7)	10	21	(54,64,70,74)	12	35	(42,52,62,76)	12	28	(40,51,63,75)	12	28
8	(4,6,6,6)	7	8	(5,5,7,7)	10	16	(54,64,70,74)	12	14	(54,56,62,76)	12	8	(45,57,61,77)	12	8
9			K	(5,5,7,7)	10	18	(54,64,70,74)	12	9	(46,54,56,74)	14	72	(45,47,63,75)	14	72
10				(5,5,7,7)	10	12	(44,54,64,74)	12	10	(46,54,66,76)	14	30	(46,55,61,77)	16	285
11				(5,5,7,7)	10	11	(54,54,64,74)	13	33	(46,52,72,76)	15	22	(43,55,71,75)	16	88
12						K	(54,54,64,74)	13	36	(52,56,66,76)	16	123	(43,53,75,76)	16	12
13							(54,54,64,74)	13	26	(52,56,66,76)	16	52	(47,55,73,75)	18	234
14									$2K$			$4K$	(47,53,67,75)	18	84
15													(47,57,65,73)	18	45
16															$3K$

TABLE II
THE BEST RATE $R = 1/3$ CODES FOUND FOR GIVEN K, $2 \leq K \leq 50$, EXHAUSTIVE SEARCH

m	1			2			3			4			5			6		
K	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d
2	(4,4,6)	4	3	(4,6,7)	4	3	(54,60,70)	4	3	(46,56,62)	4	3	(54,60,75)	4	3	(430,674,730)	4	3
3	(4,4,6)	4	3	(4,6,7)	4	3	(54,70,74)	4	3	(46,56,62)	4	3	(42,67,75)	4	3	(430,674,730)	4	3
4	(4,6,6)	4	1	(4,6,7)	6	12	(40,60,64)	6	12	(52,60,70)	6	12	(45,61,77)	6	12	(570,600,724)	6	12
5	(4,6,6)	5	6	(4,6,7)	6	10	(40,64,70)	7	15	(40,62,64)	7	15	(43,57,67)	7	15	(430,664,744)	7	15
6	(4,6,6)	5	6	(5,6,7)	6	1	(54,60,70)	8	45	(46,60,70)	8	45	(56,61,73)	8	45	(430,674,730)	8	45
7			K	(5,6,7)	7	8	(54,70,74)	8	28	(46,52,70)	8	21	(46,67,75)	8	21	(444,534,670)	8	21
8				(5,6,7)	7	8	(44,64,74)	8	13	(50,72,76)	8	1	(47,60,73)	8	1	(440,640,764)	8	1
9				(5,6,7)	7	9	(44,64,74)	8	9	(46,56,62)	10	99	(54,60,75)	10	99	(430,674,730)	10	99
10				(5,7,7)	8	45	(44,64,74)	9	20	(52,72,74)	10	46	(55,62,72)	10	21	(514,630,730)	10	21
11				(5,7,7)	8	33	(54,64,74)	10	44	(52,72,74)	10	11	(46,54,67)	11	55	(424,504,774)	11	33
12				(5,7,7)	8	27	(54,64,74)	10	36	(56,62,72)	10	6	(47,57,61)	12	200	(530,714,774)	12	178
13				(5,7,7)	8	26			$3K$	(52,66,76)	12	182	(47,55,67)	12	39	(534,674,740)	12	39
14						$2K$				(52,66,76)	12	126	(56,65,75)	12	14	(610,724,744)	13	126
15										(46,52,76)	11	15	(47,53,67)	13	75	(510,664,764)	14	390
16										(52,66,76)	12	80	(45,67,75)	13	32	(454,564,770)	14	168
17										(52,66,76)	12	85	(51,67,73)	13	34	(570,654,744)	14	51
18										(52,66,76)	12	93	(47,53,75)	13	18	(464,534,734)	15	240
19										(52,66,76)	12	95			K	(454,574,654)	15	114
20												$5K$				(564,624,754)	15	60
21																(534,664,744)	15	63
22																(564,624,754)	15	66
23																		$3K$

optimal performance provided that there are no "pseudocodewords" (multicycle paths through the trellis that are not codewords) whose "effective weight" is less than d. We have not investigated whether low effective weight pseudocodewords exist for the TB trellises presented in this correspondence.

II. SEARCHING FOR GOOD ENCODERS

Since the encoder is used for tail-biting trellis representations of block codes, the codeword set is limited to those register output sequences which end in the same register states at which they began. The full set of codewords comprises a linear block code and d, the weight of the least weight codeword in the set, is the minimum distance of the code. We search for the polynomial encoders of memory m that produce the largest d for each pair (N, K). In keeping with the usual notation, the code is called an (N, K, d) code.

For rates of the form $1/c$, c an integer, c sets of $(m + 1)$ taps, called the *generators* $\boldsymbol{g}_1, \boldsymbol{g}_2, \cdots, \boldsymbol{g}_c$, perform the encoding, as shown at the top of Fig. 2 for the case $c = 2$ (rate $R = 1/2$); m is the memory of the encoder and it is assumed that at least one generator has $g_j^{(m)} = 1$. Generator $\boldsymbol{g}_j$ describes the effect of the input on the jth bit of each group of c outputs. Similarly, the rate $R = 2/3$ encoder at the bottom in the figure is defined by the six generators $\boldsymbol{g}_{ij}, i = 1, 2; j = 1, 2, 3$, where $\boldsymbol{g}_{ij}$ describes the effect of input stream i on output stream j.

An effective strategy for finding good generators for TB trellis representations is to search exhaustively for the best noncatastrophic generators. (For TB lengths equal to a multiple of the length of the nontrivial all-zero cycle in the state transition graph of a catastrophic generator we have two all-zero trellis paths and, hence, the catastrophic generator cannot generate a TB trellis.) An early such study is [5]; we search to much higher m than they do and correct some apparent errors. For TB trellises out to $K = 50$ information bits, we have found the optimal generators by exhaustive search out to memory 5 at rate $R = 1/4$, $m = 6$ at rate $R = 1/3$, $m = 8$ at rate $R = 1/2$, and overall constraint length $\nu = 7$ at rate $R = 2/3$.

Although these codes suffice for many applications, good codes with longer memory are sometimes needed, and an exhaustive search

TABLE III
THE BEST RATE $R = 1/2$ CODES FOUND FOR GIVEN K, $2 \leq K \leq 50$, EXHAUSTIVE SEARCH

m	1			2			3			4			5			6			7			8		
K	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d
2	(4,6)	2	1	(6,7)	2	1	(54,60)	2	1	(44,76)	2	1	(53,70)	2	1	(434,730)	2	1	(556,750)	2	1	(511,733)	2	1
3	(4,6)	3	4	(4,5)	3	4	(54,74)	3	4	(46,60)	3	4	(50,67)	3	4	(434,730)	3	4	(556,750)	3	4	(461,750)	3	4
4	(4,6)	3	4	(4,7)	4	14	(40,64)	4	14	(54,62)	4	14	(46,75)	4	14	(444,610)	4	14	(570,766)	4	14	(537,705)	4	14
5			K	(6,7)	4	10	(50,64)	4	10	(46,60)	4	10	(51,66)	4	10	(434,730)	4	10	(524,712)	4	10	(461,750)	4	10
6				(6,7)	4	9	(54,60)	4	6	(46,60)	4	6	(56,75)	4	6	(410,664)	4	6	(556,750)	4	6	(461,750)	4	6
7				(5,7)	4	7	(54,74)	4	7	(44,76)	4	7	(53,70)	4	7	(434,730)	4	7	(556,750)	4	7	(423,722)	4	7
8				(5,7)	4	2	(54,60)	5	24	(46,60)	5	24	(53,70)	5	24	(530,654)	5	24	(556,750)	5	24	(423,722)	5	24
9				(5,7)	5	18	(54,70)	6	102	(52,64)	6	102	(43,73)	6	102	(564,774)	6	102	(570,766)	6	102	(513,662)	6	102
10				(5,7)	5	12	(54,70)	6	90	(54,62)	6	90	(57,60)	6	40	(554,754)	6	40	(414,642)	6	40	(537,704)	6	40
11				(5,7)	5	11	(54,74)	6	44	(56,76)	6	44	(53,70)	7	176	(444,624)	7	176	(540,606)	7	176	(472,745)	7	176
12						K	(54,74)	6	48	(62,72)	6	30	(50,67)	6	20	(414,730)	8	759	(406,670)	8	759	(411,664)	8	759
13							(54,74)	6	13	(56,76)	6	13	(56,67)	7	117	(444,614)	7	117	(476,634)	7	117	(400,653)	7	117
14									K	(46,66)	7	72	(44,73)	7	56	(654,760)	8	546	(442,632)	8	546	(576,763)	8	546
15										(56,62)	7	75	(54,75)	8	450	(414,724)	8	450	(442,766)	8	450	(453,771)	8	450
16										(56,62)	7	64	(54,75)	8	348	(564,740)	8	238	(512,654)	8	238	(437,670)	8	238
17										(46,66)	7	34	(53,76)	8	170	(534,630)	8	153	(532,674)	8	153	(422,665)	8	153
18										(56,62)	7	54	(57,63)	8	117	(564,644)	8	72	(406,732)	8	54	(530,727)	8	54
19										(56,62)	7	38	(56,75)	8	76	(550,634)	8	19	(540,762)	8	19	(537,656)	8	19
20												$2K$	(53,75)	8	60	(464,744)	9	280	(552,702)	9	240	(573,606)	9	240
21													(53,75)	8	42	(434,554)	9	189	(436,554)	10	1701	(427,667)	10	1680
22													(45,77)	8	44	(564,634)	10	1364	(562,616)	10	1298	(562,721)	10	1144
23													(55,75)	8	46	(564,634)	10	989	(572,712)	10	690	(345,572)	10	690
24													(53,75)	8	24	(454,634)	9	96	(466,772)	10	696	(561,745)	10	372
25															K	(554,744)	10	575	(554,762)	10	225	(535,616)	10	125
26																(564,634)	10	494	(436,566)	10	130	(527,703)	10	52
27																(554,744)	10	378	(552,636)	10	108	(436,665)	10	27
28																(564,634)	10	364	(522,676)	10	56	(573,621)	11	588
29																(554,744)	10	319	(446,756)	10	29	(573,635)	11	319
30																(564,634)	10	390	(522,676)	10	30	(463,751)	11	150
31																(554,744)	10	341	(572,626)	10	31	(453,755)	12	1736
32																(564,634)	10	384			K	(465,771)	12	1376
33																(554,744)	10	363				(557,751)	12	726
34																		$11K$				(515,677)	12	578
35																						(515,677)	12	490
36																						(515,677)	12	396
37																						(515,677)	12	407
38																						(515,677)	12	380
39																						(515,677)	12	403
40																						(515,677)	12	360
41																								$9K$

is now too time consuming. There are several other ways to gain an idea of the capabilities of long-memory TB encoders. One is to search for good TB generators among those similar to the best known generators for convolutional codes. Another is to choose longer generators at random and keep the best found after, say, several days of computer search. We found this second method quite effective. It probably works because most long random codes are good.

For a given set of generators and TB length L, it remains to find the minimum Hamming distance d for the code. Since the code is linear, it is enough to find the minimum weight of any codeword other than the all-zero one. The nonzero block code trellis paths fall into two cases, which are illustrated in Fig. 3.

Case (i): (Distance within one code subset, d_{intra}.) The neighbor path touches the all-zero path at least once; all paths considered are within the subset of paths leaving state 0. Finding the minimum weight among such paths is the same as finding the minimum weight path in a convolutional code. By the symmetry of the circular trellis, the behavior of paths out of one trellis stage is the same as out of all the others, so we can place the touch point at the left of the trellis. We start the usual dynamic program from this point and solve for the minimum-weight descendant in the trellis that splits from this point and merges again later to the all-zero path. Note that this can happen before L stages, but must happen at least by the Lth stage, by the tail-biting condition. We call this the *intra minimum distance* of the TB trellis representation and denote it d_{intra}.

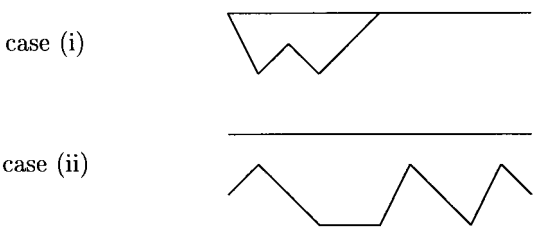

Fig. 3. Two different types of paths.

Case (ii): (Distance between code subsets, d_{inter}.) The neighbor path never touches the all-zero path. This case is unique to tail-biting. The worst case is found by executing a dynamic program starting from some state $\boldsymbol{\sigma}'$ not equal to state $\mathbf{0}$. The usual dynamic program in case (i) may be used, but with some modified search conditions. The program must delete any paths merging to the all-zero path; second,

TABLE IV
THE BEST RATE $R = 2/3$ CODES FOUND FOR GIVEN K, $4 \leq K \leq 50$, EXHAUSTIVE SEARCH

ν	1			2			3			4			5			6			7		
K	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d	G	d	n_d
4	(0,4,4)	2	3	(0,4,6)	2	3	(2,4,6)	2	3	(0,6,7)	2	3	(10,40,70)	2	3	(04,54,74)	2	3	(34,40,70)	2	3
	(4,2,4)			(4,2,4)			(7,7,2)			(4,1,7)			(74,70,20)			(70,34,40)			(74,06,52)		
6	(0,4,4)	2	3	(0,4,6)	2	3	(2,4,6)	2	3	(1,5,5)	2	3	(20,50,50)	2	3	(10,60,64)	2	3	(14,50,70)	2	3
	(6,2,6)			(4,2,4)			(7,1,4)			(6,3,7)			(74,34,60)			(60,74,34)			(74,52,26)		
8			$K/2$	(0,4,6)	3	16	(0,4,6)	3	16	(0,6,7)	3	16	(10,40,70)	3	16	(04,54,74)	3	16	(30,50,64)	3	16
				(4,2,4)			(4,7,1)			(4,1,7)			(74,70,20)			(70,34,40)			(76,64,20)		
10				(0,4,6)	3	15	(2,4,6)	4	105	(2,4,6)	4	105	(20,60,70)	4	105	(04,50,64)	4	105	(34,44,54)	4	105
				(4,2,4)			(7,6,2)			(4,0,7)			(74,04,40)			(54,64,30)			(70,02,60)		
12				(0,4,6)	3	8	(2,4,6)	4	102	(0,6,7)	4	78	(00,60,70)	4	78	(04,54,74)	4	78	(24,70,70)	4	78
				(6,2,4)			(7,6,2)			(4,3,5)			(40,74,04)			(40,24,60)			(72,20,56)		
14				(0,4,6)	3	7	(2,6,6)	4	56	(0,6,7)	4	56	(10,50,50)	4	56	(04,54,64)	4	56	(34,40,70)	4	56
				(6,2,4)			(7,2,7)			(4,1,6)			(54,34,40)			(50,70,14)			(72,26,70)		
16						$K/2$	(2,4,6)	4	46	(0,5,7)	4	26	(30,70,70)	4	26	(00,44,64)	4	26	(34,44,54)	4	26
							(7,5,0)			(7,3,4)			(74,34,60)			(60,10,64)			(76,60,36)		
18							(2,6,6)	4	27	(1,6,6)	4	18	(20,40,50)	4	18	(14,50,70)	4	18	(10,60,64)	4	18
							(7,2,7)			(6,2,7)			(70,24,60)			(74,34,40)			(76,56,00)		
20							(2,4,6)	4	15	(1,6,6)	5	172	(30,40,50)	5	162	(10,60,64)	5	162	(24,44,64)	5	162
							(7,1,4)			(6,2,7)			(70,30,44)			(40,14,74)			(70,66,30)		
22							(2,4,6)	4	11	(1,5,5)	5	132	(20,50,70)	6	1287	(00,54,64)	6	1276	(34,70,74)	6	1276
							(7,1,4)			(6,3,7)			(70,40,14)			(54,30,64)			(74,16,70)		
24									$K/2$	(1,5,5)	5	96	(20,50,70)	6	1048	(00,64,70)	6	1036	(24,70,70)	6	1024
										(6,3,7)			(70,40,14)			(54,20,50)			(70,26,50)		
26										(1,4,7)	5	52	(30,40,70)	6	793	(04,44,54)	6	767	(04,44,54)	6	767
										(4,7,3)			(70,64,24)			(60,54,10)			(46,10,74)		
28										(0,5,7)	5	42	(10,50,70)	6	574	(04,44,54)	6	329	(30,40,74)	6	259
										(7,3,4)			(74,64,20)			(50,34,50)			(72,26,76)		
30										(1,5,6)	5	45	(20,50,70)	6	315	(04,50,54)	6	195	(24,60,70)	6	145
										(6,7,3)			(74,40,14)			(50,74,10)			(70,06,40)		
32										(2,7,7)	5	32	(30,60,70)	6	160	(04,60,64)	6	104	(34,40,54)	6	72
										(5,3,4)			(70,44,04)			(60,34,74)			(72,16,50)		
34												K	(30,60,70)	6	170	(04,64,70)	6	51	(20,54,70)	6	17
													(70,44,04)			(50,74,04)			(76,66,32)		
36													(30,60,70)	6	147	(10,54,64)	6	36	(10,44,74)	7	774
													(74,64,00)			(44,74,10)			(76,20,72)		
38													(30,60,70)	6	133	(00,54,74)	6	38	(34,50,60)	7	437
													(70,04,54)			(54,34,44)			(70,32,76)		
40													(30,60,70)	6	140	(04,54,70)	6	20	(14,54,74)	7	300
													(74,64,00)			(50,20,54)			(74,76,32)		
42													(30,60,70)	6	147	(10,54,54)	7	528	(20,54,74)	8	3780
													(70,04,54)			(64,30,64)			(76,06,54)		
44													(30,60,70)	6	132	(10,54,54)	7	418	(14,54,70)	8	2860
													(74,64,00)			(64,30,64)			(76,34,42)		
46															$6K$	(10,54,54)	7	414	(20,54,74)	8	2162
																(64,30,64)			(76,06,54)		
48																(10,54,54)	7	432	(20,54,74)	8	1842
																(64,30,64)			(76,06,54)		
50																(10,54,54)	7	425	(30,64,64)	8	1400
																(64,30,64)			(76,04,42)		

at the Lth stage only paths entering state $\boldsymbol{\sigma}'$ count. The procedure must be repeated for each nonzero starting state, and the minimum taken over all the outcomes. We call this the *inter minimum distance* of the TB trellis representation and denote it d_{inter}.

The minimum distance for the tail-biting representation is $d = \min\{d_{\text{intra}}, d_{\text{inter}}\}$. If the tail-biting trellis is long enough, case (i) paths lead to the minimum distance but for short trellises case (ii) codewords may lead to the minimum weight path, i.e., d_{inter} might be less than d_{intra}. Hence, short TB trellis representations of block codes could have quite different optimal generators than terminated convolutional codes.

The minimum distance discussed above is the principal determinant of the merit of the resulting code when maximum-likelihood decoding or APP decoding is used. However, for certain suboptimal decoding algorithms it has recently been observed by Frey, Kötter, and Vardy [6] and Forney, Kschischang, and Marcus [7] that the decoding performance may be affected by so-called "pseudocodewords." A pseudocodeword is a trellis path of one or more cycles that starts and ends in the same state at some integer multiple of the cycle length. The codewords are precisely the length-L pseudocodewords. See also [8], [9]. We have not checked whether our tail-biting representations have pseudocodewords with low effective weight.

III. RESULTS AND OBSERVATIONS

Tables I–IV list the best encoders found by exhaustive search for rates $R = 1/4, 1/3, 1/2$, and $2/3$, respectively. For each number of information bits K up to 50, the best time-invariant generators are listed at each encoder memory m or overall constraint length ν. Among the generators leading to the same distance d, the set shown leads to the fewest codeword neighbors at d, denoted n_d. The generator notation is octal, with each octal symbol representing groups of three generator bits beginning at the left; e.g., $g_j^{(0)}, \cdots, g_j^{(3)} = 1101$ base $2 = 64$ base 8.

Encoders of fixed memory m achieve higher distances as the tail-biting trellis lengthens. At a given distance, a longer trellis reduces the number of nearest neighbors at d. Eventually, neither d nor n_d

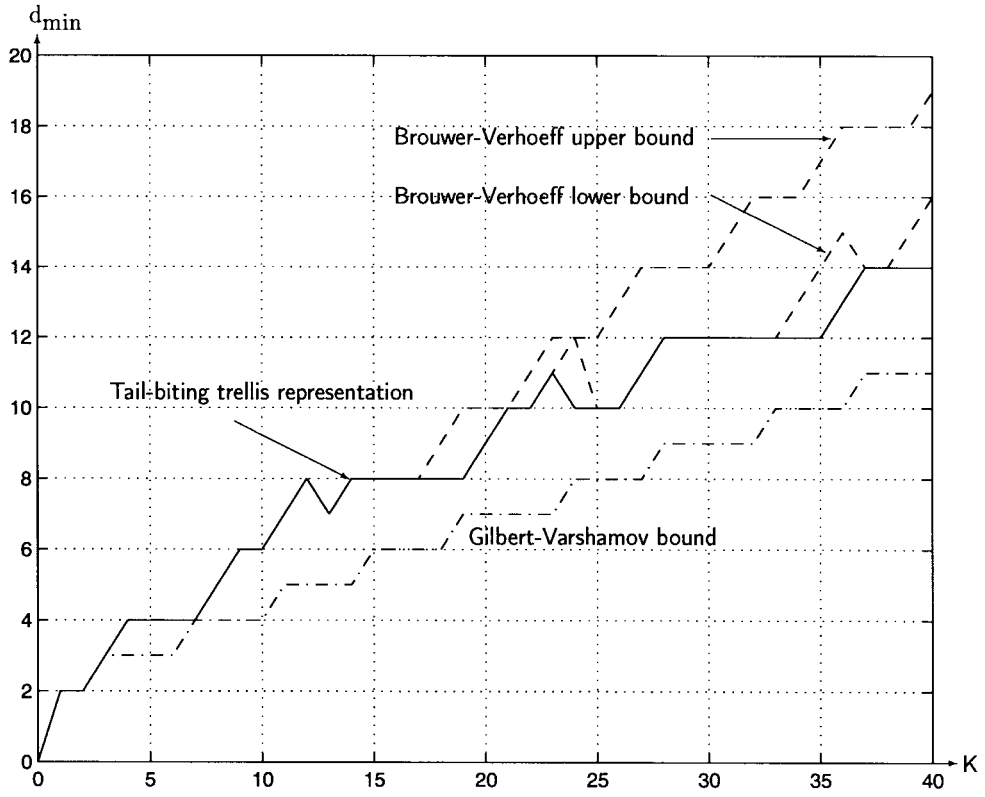

Fig. 4. Best minimum distance found for $R = 1/2$.

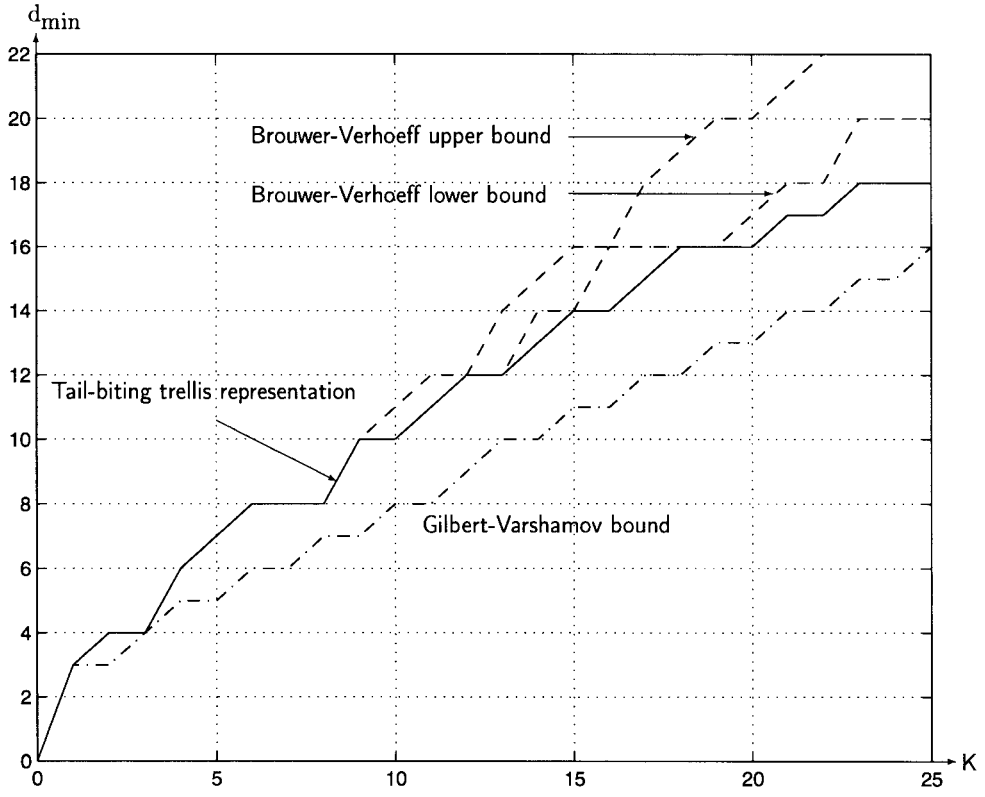

Fig. 5. Best minimum distance found for $R = 1/3$.

improves further with tail-biting length; in fact, the best generators are now those of the memory m convolutional code with best free distance and best $n_{d_{\text{free}}}$. When the columns of a table reach this point, no further entries are shown. The neighbor number continues to grow with K, according to the linear rate indicated. An explanation of the K at which distance saturation occurs is given in the next section.

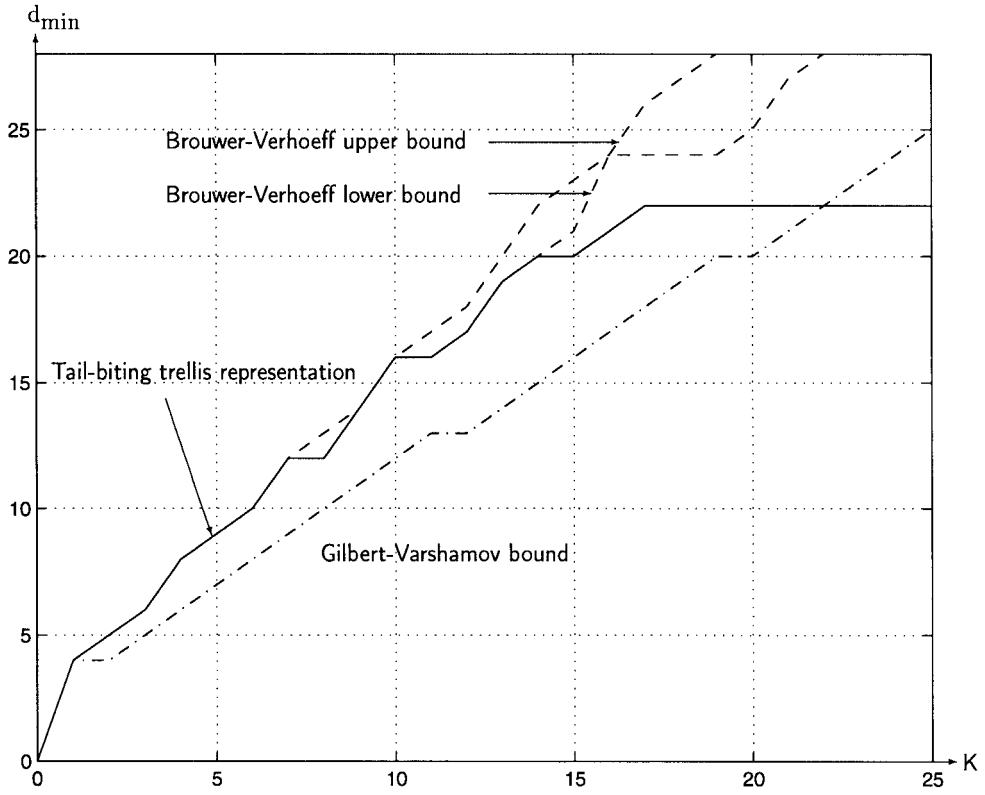

Fig. 6. Best minimum distance found for $R = 1/4$.

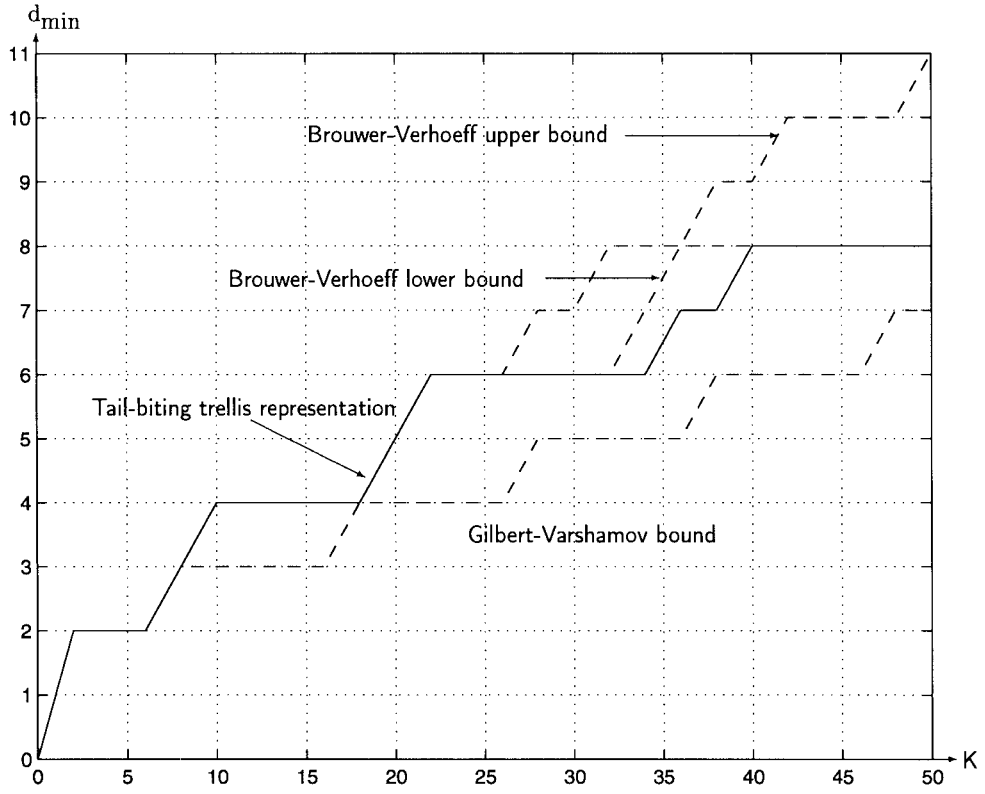

Fig. 7. Best minimum distance found for $R = 2/3$.

TABLE V
THE BEST RATE $R = 1/4$ CODES FOUND FOR GIVEN K, $2 \le K \le 50$. THE NONSYSTEMATIC ENCODERS ARE FOUND BY EXHAUSTIVE SEARCH FOR $1 \le m \le 5$ AND BY RANDOM SEARCH FOR $m \ge 6$. THE SYSTEMATIC ENCODERS ARE ALL FOUND BY EXHAUSTIVE SEARCH. THE BEST $K = 23$ NONSYSTEMATIC ENCODER IS BEST ALSO FOR $24 \le K \le 50$ WITH $n_d = K$. THE BEST $K = 15$ SYSTEMATIC ENCODER IS BEST ALSO FOR $16 \le K \le 50$ WITH $n_d = K$

	nonsystematic				systematic			
K	G	m	d	n_d	G	m	d	n_d
2	(4,4,4,6)	1	5	2	(4,4,4,6)	1	5	2
3	(4,4,6,7)	2	6	3	(4,4,6,7)	2	6	3
4	(4,5,6,7)	2	8	11	(4,5,6,7)	2	8	11
5	(40,54,60,70)	3	9	10	(40,54,60,70)	3	9	10
6	(46,56,62,76)	4	10	9	(40,54,64,72)	4	10	9
7	(42,52,62,76)	4	12	28	(40,51,63,75)	5	12	28
8	(54,56,62,76)	4	12	8	(40,53,61,75)	5	12	8
9	(46,54,56,74)	4	14	72	(400,510,630,764)	6	14	72
10	(46,55,61,77)	5	16	285	(400,464,734,744)	6	16	285
11	(470,274,714,610)	6	16	66	(400,412,576,674)	7	16	66
12	(424,474,704,724)	6	17	60	(400,532,574,716)	7	17	72
13	(564,164,534,714)	6	19	325	(400,516,732,746)	7	18	156
14	(646,536,526,720)	7	20	805	(400,546,572,736)	7	18	14
15	(536,616,474,266)	7	20	198	(400,456,672,736)	7	18	15
16	(456,466,552,636)	7	21	128				K
17	(466,616,652,772)	7	22	340				
18	(452,616,536,766)	7	22	234				
19	(662,746,576,532)	7	22	76				
20	(452,616,536,766)	7	22	80				
21	(662,746,576,532)	7	22	42				
22	(736,636,452,726)	7	22	23				
23	(472,556,726,762)	7	22	23				
24				K				

TABLE VI
THE BEST RATE $R = 1/3$ CODES FOUND FOR GIVEN K, $2 \le K \le 50$. THE NONSYSTEMATIC ENCODERS ARE FOUND BY EXHAUSTIVE SEARCH FOR $1 \le m \le 6$ AND BY RANDOM SEARCH FOR $m \ge 7$. THE SYSTEMATIC ENCODERS ARE ALL FOUND BY EXHAUSTIVE SEARCH. THE BEST $K = 28$ NONSYSTEMATIC ENCODER IS BEST ALSO FOR $29 \le K \le 50$ WITH $n_d = 4K$. THE BEST $K = 21$ SYSTEMATIC ENCODER IS BEST ALSO FOR $22 \le K \le 50$ WITH $n_d = 3K$

	nonsystematic				systematic			
K	G	m	d	n_d	G	m	d	n_d
2	(4,4,6)	1	4	3	(4,4,6)	1	4	3
3	(4,4,6)	1	4	3	(4,4,6)	1	4	3
4	(4,6,7)	2	6	12	(4,6,7)	2	6	12
5	(40,64,70)	3	7	15	(40,64,70)	3	7	15
6	(54,60,70)	3	8	45	(40,56,70)	4	8	45
7	(46,52,70)	4	8	21	(40,46,76)	4	8	21
8	(50,72,76)	4	8	1	(40,56,66)	4	8	1
9	(46,56,62)	4	10	99	(40,53,67)	5	10	99
10	(55,62,72)	5	10	21	(400,672,704)	7	10	21
11	(424,504,774)	6	11	33	(400,456,722)	7	11	33
12	(530,714,774)	6	12	178	(400,543,767)	8	12	178
13	(47,55,67)	5	12	39	(400,417,573)	8	12	39
14	(762,442,644)	7	13	112	(400,463,537)	8	13	154
15	(332,670,452)	7	14	330	(400,475,733)	8	14	390
16	(712,476,650)	7	14	80	(400,455,737)	8	14	240
17	(442,075,567)	8	15	204	(400,455,737)	8	14	102
18	(731,271,375)	8	16	648	(400,453,767)	8	14	90
19	(767,323,247)	8	16	323	(400,557,723)	8	14	57
20	(517,415,675)	8	16	100	(400,455,737)	8	14	80
21	(157,467,625)	8	17	130	(400,557,737)	8	14	63
22	(423,563,675)	8	17	110				$3K$
23	(575,323,517)	8	18	460				
24	(733,753,511)	8	18	360				
25	(437,673,625)	8	18	175				
26	(437,673,625)	8	18	130				
27	(465,537,671)	8	18	111				
28	(465,537,671)	8	18	112				
29				$4K$				

For a fixed K, there is also an encoder memory m or overall constraint length ν, going across a table row, at which no further distance growth occurs, although sometimes a larger m or ν allow an encoder with fewer nearest neighbors. There is thus a saturation as m or ν grow for a given K, as well as *vice versa*. As a rule, a code with a given d should be as short as possible, in order to obtain a good decoded data error rate.

Tables V–VIII show the best time-invariant tail-biting generators found at each K, regardless of encoder memory or overall constraint length. These tables also list the best feedforward systematic encoders found by exhaustive searching.

Figs. 4–7 compare d and K for the best TB encoders to the best bounds given in Brouwer and Verhoeff [10], for all four rates. Note that the TB encoders in the right half of the figure were found by random search and are not optimal; wherever the TB encoders are known to be optimal, they essentially match the B-V table. It is likely that better TB encoders exist where L is large.

Several especially interesting codes were found during the searches. A number of rate $R = 1/3$ $(15, 5, 7)$ TB trellis representations with memory 3 were found which are equivalent to the $(15, 5, 7)$ Bose–Chaudhuri–Hocquenghem (BCH) code (i.e., by permuting codeword bit positions, the TB codeword set can be made identical to the BCH code). This eight-state TB trellis is the trellis description of this BCH code with the fewest known states (cf. [11]). The square-root lower bound combined with Schuurman's 16-state conventional trellis for the $(15, 5, 7)$ BCH code [12] suggests that there might be a four-state time-varying TB trellis for this code. By computer search we showed that, somewhat surprisingly, no such generator exists.

Among the optimal rate $R = 1/2$ codes, a $(24, 12, 8)$ code with memory 6 was found. Since the extended Golay code has the same (N, K, d) and it can be shown that there is only one such code, we have found a 64-state TB trellis representation for the extended Golay code, i.e., a TB trellis representation with the same number of states as the trellis given in [13]; such a trellis was previously reported in [1] and [5]. (These three are nonequivalent as convolutional encoders.) Recently, Calderbank, Forney, and Vardy [14] reported a 16-state TB trellis for this code, but the trellis is time-varying with period 4, whereas our 64-state TB trellis is time-invariant. The existence of the 16-state, rate $R = 1/2$ TB trellis for the $(24, 12, 8)$ Golay code shows that nonnegligible savings in complexity might be obtained by considering time-varying generators.

Some of our TB trellis representations beat those given in [5]. For example, our 64-state, rate $R = 1/4$ $(48, 12, 17)$ trellis beats the $(48, 12, 16)$ code given there; this code ties the shortening of the $(49, 13, 17)$ code in [10]. Since a 16-state TB trellis for a $(48, 12, 16)$ code can be obtained just by doubling the bits in the Golay code, it seems probable that there is a simpler time-varying TB trellis for the $(48, 12, 17)$ code as well.

Three of our interesting encoders are given in Table IX.

IV. DISTANCE BEHAVIOR AS THE TAIL-BITING TRELLIS GROWS

As noted in Section II, optimal generators for short TB trellises are very different from those of convolutional codes. Consider a fixed memory m. Long TB trellis representations have both the same generators and the same distance as the best memory m convolutional codes. For TB lengths L in between (about $4m$ in the rate $R = 1/2$ case), the generators become the same as or at least roughly similar to

TABLE VII
THE BEST RATE $R = 1/2$ CODES FOUND FOR GIVEN K, $2 \leq K \leq 50$. THE NONSYSTEMATIC ENCODERS ARE FOUND BY EXHAUSTIVE SEARCH FOR $1 \leq m \leq 8$ AND BY RANDOM SEARCH FOR $m \geq 9$. THE SYSTEMATIC ENCODERS ARE ALL FOUND BY EXHAUSTIVE SEARCH. THE BEST $K = 33$ SYSTEMATIC ENCODER IS BEST ALSO FOR $34 \leq K \leq 50$ WITH $n_d = 2K$

	nonsystematic				systematic			
K	G	m	d	n_d	G	m	d	n_d
2	(4,6)	1	2	1	(4,6)	1	2	1
3	(4,6)	1	3	4	(4,6)	1	3	4
4	(4,7)	2	4	14	(4,7)	2	4	14
5	(6,7)	2	4	10	(40,74)	3	4	10
6	(54,60)	3	4	6	(40,72)	4	4	6
7	(5,7)	2	4	7	(40,74)	3	4	7
8	(54,60)	3	5	24	(40,72)	4	5	24
9	(54,70)	3	6	102	(400,654)	6	6	102
10	(57,60)	5	6	40	(400,762)	7	6	40
11	(53,70)	5	7	176	(400,732)	7	7	176
12	(414,730)	6	8	759	(400,675)	8	8	759
13	(56,67)	5	7	117	(400,653)	8	7	117
14	(654,760)	6	8	546	(400,727)	8	8	546
15	(414,724)	6	8	450	(4000,7152)	10	8	450
16	(564,740)	6	8	238	(4000,7147)	11	8	238
17	(534,630)	6	8	153	(4000,6762)	10	8	170
18	(406,732)	7	8	54	(4000,6572)	10	8	72
19	(540,762)	7	8	19	(4000,6571)	11	8	19
20	(552,702)	7	9	240	(4000,6275)	11	9	320
21	(427,667)	8	10	1680	(4000,6571)	11	9	189
22	(562,721)	8	10	1144	(4000,6457)	11	9	110
23	(1757,5207)	11	11	3312	(4000,6457)	11	9	69
24	(4744,2132)	10	10	252	(4000,7345)	11	9	72
25	(535,616)	8	10	125	(4000,7345)	11	9	75
26	(6445,2053)	11	10	39	(4000,6457)	11	9	52
27	(5640,6542)	10	11	675	(4000,6457)	11	9	57
28	(6551,4055)	11	12	4690	(4000,7153)	11	9	84
29	(6351,4524)	11	12	3161	(4000,6457)	11	9	58
30	(5534,6417)	11	12	2285	(4000,6457)	11	9	60
31	(3356,4245)	11	12	868	(4000,6457)	11	9	62
32	(4763,6235)	11	12	632	(4000,7153)	11	9	96
33	(6467,5501)	11	12	275	(4000,6457)	11	9	66
34	(7204,4667)	11	12	136				2K
35	(3531,6407)	11	12	35				
36	(5355,7167)	11	13	936				
37	(7715,6501)	11	14	6401				
38	(5133,7317)	11	14	4674				
39	(5765,7633)	11	14	2964				
40	(5755,6141)	11	14	1880				
41	(5437,6235)	11	14	738				
42	(6773,6235)	11	14	378				
43	(4637,7521)	11	14	258				
44	(4373,6523)	11	14	286				
45	(5175,6643)	11	14	90				
46	(4325,6747)	11	15	1474				
47	(5537,6131)	11	14	47				
48	(5147,6671)	11	14	48				
49	(5175,6643)	11	14	49				
50	(5537,6131)	11	14	50				

TABLE VIII
THE BEST RATE $R = 2/3$ CODES FOUND FOR GIVEN K, $4 \leq K \leq 50$. THE NONSYSTEMATIC ENCODERS ARE FOUND BY EXHAUSTIVE SEARCH FOR $1 \leq \nu \leq 6$ AND BY RANDOM SEARCH FOR $\nu \geq 7$. THE SYSTEMATIC ENCODERS ARE ALL FOUND BY EXHAUSTIVE SEARCH

	nonsystemaic				systematic			
K	G	ν	d	n_d	G	ν	d	n_d
4	(0,4,4)	1	2	3	(4,0,4)	1	2	3
	(4,2,4)				(0,4,6)			
6	(0,4,4)	1	2	3	(4,0,4)	1	2	3
	(6,2,6)				(0,4,6)			
8	(0,4,6)	2	3	16	(4,0,6)	3	3	16
	(4,2,4)				(0,4,7)			
10	(2,4,6)	3	4	105	(40,00,70)	5	4	105
	(7,6,2)				(00,40,54)			
12	(0,6,7)	4	4	78	(40,00,70)	5	4	102
	(4,3,5)				(00,40,54)			
14	(2,6,6)	3	4	56	(40,00,74)	7	4	56
	(7,2,7)				(00,40,62)			
16	(0,5,7)	4	4	26	(40,00,72)	8	4	26
	(7,3,4)				(00,40,66)			
18	(1,6,6)	4	4	18	(40,00,54)	8	4	18
	(6,2,7)				(00,40,71)			
20	(30,40,50)	5	5	162	(400,000,522)	14	5	162
	(70,30,44)				(000,400,372)			
22	(00,54,64)	6	6	1276	(400,000,722)	14	6	1276
	(54,30,64)				(000,400,456)			
24	(76,54,22)	8	6	1024	(400,000,744)	14	6	1024
	(10,42,54)				(000,400,537)			
26	(04,44,54)	6	6	767	(400,000,766)	15	6	767
	(50,74,10)				(000,400,543)			
28	(04,66,42)	8	6	259	(4000,0000,6314)	18	6	308
	(70,46,56)				(0000,4000,5654)			
30	(14,70,46)	8	6	150	(4000,0000,3750)	17	6	180
	(64,74,36)				(0000,4000,5154)			
32	(52,12,54)	8	6	80	(4000,0000,5570)	17	6	88
	(26,52,40)				(0000,4000,6514)			
34	(52,64,16)	8	6	34	(4000,0000,6710)	17	6	34
	(10,66,62)				(0000,4000,5174)			
36	(64,67,77)	10	7	792	(4000,0000,6454)	18	6	18
	(22,37,41)				(0000,4000,5714)			
38	(15,50,22)	10	7	513	(4000,0000,6454)	18	7	513
	(75,01,63)				(0000,4000,5714)			
40	(57,15,17)	10	8	4875	(4000,0000,7444)	18	6	10
	(53,23,51)				(0000,4000,5174)			
42	(04,63,75)	10	8	3801	(4000,0000,7464)	18	7	318
	(62,75,33)				(0000,4000,6134)			
44	(34,21,45)	10	8	2904	(4000,0000,7654)	18	7	242
	(72,57,01)				(0000,4000,5164)			
46	(34,65,40)	10	8	1886	(4000,0000,7654)	18	7	276
	(60,13,32)				(0000,4000,5164)			
48	(04,63,75)	10	8	1350	(4000,0000,7654)	18	7	264
	(62,75,33)				(0000,4000,5164)			
50	(64,41,17)	10	8	350	(4000,0000,7654)	18	7	275
	(43,13,65)				(0000,4000,6514)			

TABLE IX
THREE INTERESTING TB ENCODERS

R	(N, K, d)	Generator (octal notation)	m	n_d
1/2	(24,12,8)	(414 730)	6	759
1/3	(15,5,7)	(40 64 70)	3	15
1/4	(48,12,17)	(470 224 354 744)	6	108

the best free-distance generators of memory m convolutional codes, but the achieved distance falls short.

Thinking conversely, we can fix the TB sections at L and ask what is the best distance obtainable by any code found at any memory. These distances can be seen in Fig. 4. They are obtained from generators whose m is considerably shorter than the tail-biting trellis. The relationship of K to d and m seems to be similar to that of the decision depth parameter for convolutional codes to d and m. We now discuss this relationship.

Roughly speaking, the decision depth parameter is the depth needed in convolutional decoding so that distance properties are not dominated by unmerged trellis paths. A precise definition of this depth can be given with reference to the case (i) paths in Fig. 3. Consider all trellis paths stemming from a root node as in the figure; without loss of generality, the all-zero path is one of these and was transmitted. Then the decision depth function $L_D(d)$ at distance d is the first trellis depth at which every nonzero path, merged with the all-zero path or not, is d away from the all-zero path. In ordinary trellis decoding, the decision on the first branch cannot be made until at least $L_D(d)$ stages have been explored, if the code is to act as if it had minimum distance d. The decoder observation window is thus L_D trellis stages. In a typical code, the paths merging to the all-zero

path that leads to the free distance occurs well before $L_D(d_{\text{free}})$, since the many close unmerged paths are what dominate L_D. The L_D function is tabulated in [15] for most convolutional codes. These tabulations show that $L_D(d)$ rapidly assumes the asymptotic rule

$$L_D(d) \approx d/ch^{-1}(1-R) \tag{6}$$

as d grows, where $h^{-1}(\ )$ is the inverse binary entropy function. That the right-hand side is an overbound can be shown for several theoretical classes of codes, including time-invariant convolutional codes for $L_D(d) \leq m+1$ and random convolutional codes [16], [3].

Turning now to tail-biting trellis representations of block codes, we modify the discussion, but not by much. Case (ii) path pairs, for which merger to all-zero is forbidden, must be considered, and paths must start and end in the same state. In addition, TB trellis representations are block codes, so the Gilbert–Varshamov bound applies; it states that a linear (N, K, d) block code exists whenever

$$\sum_{i=0}^{d-2} \binom{N-1}{i} < 2^{N-K}. \tag{7}$$

As N, K, d grow, (7) takes the asymptotic form $d/N \gtrapprox h^{-1}(1-R)$, which may be rewritten for our code parameters as

$$L = K/b \lessapprox d/ch^{-1}(1-R) \tag{8}$$

or, equivalently,

$$K \lessapprox dR/h^{-1}(1-R). \tag{9}$$

If we assume that the Gilbert–Varshamov bound is tight, then (8) is the same as (6). The precise trends here for block codes are shown in Figs. 4–7. None of this guarantees that an (N, K, d) TB representation exists with size L as small as (8). But if the TB trellis representations have performance equal to that of the best block codes, then their length L and distance d will be related as in (8). Figs. 4–7 show that this is apparently the case.

To summarize these remarks, consider a TB trellis representation with growing length, derived from generators of fixed memory m. At small L, case (ii) unmerged path pairs determine which generators have the best distance. As L grows, the best selection changes unpredictably, but our results show that the codes are excellent and that (9) approximately holds. At large L, the case (i) d_{intra} must eventually dominate d, since the minimum distance among case (ii) path pairs clearly grows linearly with L and exceeds any limit. This statement applies even though the TB condition applies: If a given distance d' is reached by L', descendants can reach any specified state by $L' + m$, and their distance cannot be smaller. This being the case, the entire distance structure must come from case (i). The best TB generators of memory m become identical to the best convolutional code generators. So also must all the distance properties, and thus (6) and (7) apply to tail-biting trellis representations of block codes as they do to convolutional codes.

V. Conclusion

We have constructed a new list of optimal short and moderate-length encoders for tail-biting trellis representations of block codes. These are as good as the best known block codes at their rate and block length, and at the same time they have important advantages over block and convolutional codes. Soft channel outputs yield a 2–3-dB signal-to-noise ratio (SNR) improvement, and are easily obtained by trellis decoders. The trellis structure itself seems to lead to simpler decoding, compared to block decoders with the same (N, K, d). Finally, the rate loss from the terminating bits in a convolutional encoder, which can be significant at moderate block lengths, is eliminated by tail-biting representations. The advantages summarized here are particularly important in packet and other short-block transmission systems.

Acknowledgment

The suggestions made by the reviewers helped improve this correspondence.

References

[1] G. Solomon and H. C. A. van Tilborg, "A connection between block and convolutional codes," *SIAM J. Appl. Math.*, vol. 37, pp. 358–369.
[2] H. H. Ma and J. K. Wolf, "On tailbiting codes," *IEEE Trans. Commun.*, vol. COM-34, pp. 104–111, Feb. 1986.
[3] R. Johannesson and K. Sh. Zigangirov, *Fundamentals of Convolutional Coding*. Piscataway, NJ: IEEE Press, 1999.
[4] J. B. Anderson and S. M. Hladik, "Tailbiting BCJR decoders," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 297–302, Feb. 1998.
[5] B. D. Kudryashov and T. G. Zakharova, "Block codes from convolutional codes," *Probl. Pered. Inform.*, vol. 25, pp. 98–102, Oct.–Dec. 1989.
[6] B. J. Frey, R. Kötter, and A. Vardy, "Skewness and pseudocodewords in iterative decoding," in *Proc. 1998 IEEE Int. Symp. Information Theory* (Cambridge, MA: MIT, Aug. 16–21, 1998), p. 148.
[7] G. D. Forney, Jr., F. R. Kschischang, and B. Marcus, "Iterative decoding of tail-biting trellises," in *Proc. 1998 IEEE Information Theory Workshop* (San Diego, CA, Feb. 9–11, 1998).
[8] N. Wiberg, "Codes and decoding on general graphs," Ph.D. dissertation, Dept. Elec. Eng., Univ. Linköping, Sweden, Apr. 1996.
[9] N. Wiberg, H.-A. Loeliger, and R. Kötter, "Codes and iterative decoding on general graphs," *Euro. Trans. Telecomm.*, vol. 6, pp. 513–525, 1995.
[10] A. E. Brouwer and T. Verhoeff, "An updated table of minimum distance bounds for linear codes," *IEEE Trans. Inform. Theory*, vol. 39, pp. 662–677, 1993. [Online]. Available WWW: http://www.win.tue.nl/math/dw/voorlincod.html.
[11] A. Lafourcade and A. Vardy, "Lower bounds on trellis complexity of block codes," *IEEE Trans. Inform. Theory*, vol. 41, pp. 1938–1954, 1995.
[12] P. Schuurman, "A table of state complexity bounds for binary linear codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 2034–2042, 1996.
[13] G. D. Forney, Jr., "Coset codes—Part II: Binary lattices and related codes," *IEEE Trans. Inform. Theory*, vol. 34, pp. 1152–1187, 1988.
[14] A. R. Calderbank, G. D. Forney, Jr., and A. Vardy, "Minimal tail-biting trellises: The Golay code and more," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1435–1455, July 1999.
[15] J. B. Anderson and K. Balachandran, "Decision depths of convolutional codes," *IEEE Trans. Inform. Theory*, vol. 35, pp. 455–459, Mar. 1989.
[16] R. Johannesson and K. Sh. Zigangirov, "Distances and distance bounds for convolutional codes," in *Topics in Coding Theory—In honor of Lars H. Zetterberg*. Berlin, Germany: Springer-Verlag, 1989, pp. 109–136.