Titre: Title:	High rate punctured convolutional codes: structure properties and construction technique
Auteurs: Authors:	Guy Bégin, & David Haccoun
Date:	1987
Type:	Rapport / Report
Référence: Citation:	Bégin, G., & Haccoun, D. (1987). High rate punctured convolutional codes: structure properties and construction technique. (Rapport technique n° EPM-RT-87-45). https://publications.polymtl.ca/9846/

EPM/RT-87/45

HIGH RATE PUNCTURED CONVOLUTIONAL CODES: STRUCTURE PROPERTIES AND CONSTRUCTION TECHNIQUE

GUY BEGIN, Ing.
DAVID HACCOUN, Ing. Ph.D.

Ecole Polytechnique de Montréal

Novembre 1987

Dépôt légal, 4e trimestre 1987
Bibliothèque nationale du Québec
Bibliothèque nationale du Canada

Pour se procurer une copie de ce document, s'adresser au:

Editions de l'Ecole Polytechnique de Montréal
Ecole Polytechnique de Montréal
Case postale 6079, Succursale A
Montréal (Québec) H3C 3A7
(514) 340-4000

Compter 0,10$ par page (arrondir au dollar le plus près) et ajouter 3,00$ (Canada) pour la couverture, les frais de poste et la manutention. Régler en dollars canadiens par chèque ou mandat-poste au nom de l'Ecole Polytechnique de Montréal. Nous n'honorerons que les commandes accompagnées d'un paiement, sauf s'il y a eu entente préalable dans le cas d'établissements d'enseignement, de sociétés ou d'organismes canadiens.

HIGH RATE PUNCTURED CONVOLUTIONAL CODES:

STRUCTURE PROPERTIES AND CONSTRUCTION TECHNIQUE*

by

Guy BEGIN and David HACCOUN

Department of Electrical Engineering
Ecole Polytechnique de Montreal
C. P. 6079, station A,
Montreal, Quebec, Canada.
H3C 3A7

* This research has been supported in part by the Natural Sciences and Engineering Research Council of Canada and by the Fonds FCAR of Quebec.

HIGH RATE PUNCTURED CONVOLUTIONAL CODES: STRUCTURE PROPERTIES AND CONSTRUCTION TECHNIQUE

by

Guy BEGIN and David HACCOUN

ABSTRACT

This paper presents some properties of punctured convolutional codes and provides a construction method and a list of new good high rate long memory punctured codes. The structure of punctured codes is examined and an upper bound on the free distance of punctured codes is derived, indicating that punctured codes are good codes. A construction method that generates the low rate original codes which duplicate given known high rate codes through perforation is proposed. Tables of punctured codes that duplicate the best known non-systematic codes of rate 2/3 and 3/4 with memories ranging from 3 to 23 and from 3 to 9 respectively are given, together with the best known systematic codes for rates ranging from 2/3 to 7/8 with very long memory (44-48).

I. Introduction

The use of convolutional coding and probabilistic decoding offers some very attractive solutions to the problems of channel error correction and detection. Much effort has been devoted to finding efficient schemes using convolutional codes, for a wide range of applications [1]-[4]. However, most of this work has been devoted to low rate convolutional codes. Relatively little work has been done to promote the use of high rate convolutional codes, because in general, the usual decoding techniques that are suitable for low rate convolutional codes become rapidly cumbersome when used for high rate codes.

To alleviate this problem, a special class of high rate convolutional codes called punctured convolutional codes has been proposed [5]. These high rate punctured codes are derived from low rate original codes and thus maintain the simple underlying structure of low rate codes. They may therefore easily be decoded by traditional techniques such as Viterbi decoding, as was originally proposed [5]-[7], or more recently, by sequential decoding [8]-[10]. An interesting feature of these codes is that they allow the easy implementation of variable rate coding-decoding.

This paper presents some properties of punctured codes and provides a construction method and a list of new good long memory punctured codes. After presenting preliminary concepts in section II, we examine in section III the structure of punctured codes by relating the paths of the low rate original code to those of the resulting punctured code. This approach yields useful properties that may guide the search for good punctured codes. Section IV pertains to the relationship that exists between the Hamming weights of punctured and non punctured paths. Using a special class of perforation patterns an upper bound on the free distance of punctured codes is derived. In section V, the problem of constructing good long memory punctured codes suitable for sequential decoding is considered. A construction method that generates the low rate original codes which duplicate given known high rate codes through perforation is proposed. Using this construction procedure, original low rate codes which duplicate the best known high rate codes are found. Tables of punctured codes that are equivalent to the best known non-systematic codes of rate 2/3 and 3/4 with memories ranging from 3 to 23 and from 3 to 9 respectively, are given in section VI. Finally, punctured codes that duplicate the best known systematic codes for rates ranging from 2/3 to 7/8 with memories about 50 are also given.

II. Basic concepts

A punctured code is a high rate convolutional code obtained by periodically deleting (i.e. puncturing) certain symbols from the output of a low rate encoder. The resulting high rate code depends on the low rate code produced by the low rate encoder, called the original code, and on the number and specific location of the deleted symbols. The pattern of occurrence of the punctured symbols is called the perforation pattern and it is conveniently expressed in matrix form.

Throughout this paper, we will use the notation (V,B) to denote a convolutional code produced by a B-input/V-output encoder. Whenever the memory of the encoder is to be specified, the code will denoted as a (V,B) code of memory M. The coding rate R of a (V,B) code is obviously B/V. The specific convolutional code is specified by its generating matrix G(D) of dimension B x V whose elements are the generator polynomials:

$$g_{ij}(D) = \sum_{k=0}^{m_i} g_{ij}^{k} D^{k} = g_{ij}^{0} + g_{ij}^{1} D + \ldots + g_{ij}^{m_i} D^{m_i}; \qquad (1)$$

$$i = 1, \ldots, B; \; j = 1, \ldots, V.$$

Whenever B = 1, i is dropped in this notation.

For a (V,B) convolutional code obtained from a (Vo,1) original code, the perforation pattern is expressed as a B rows and Vo columns binary matrix P with elements

$$[p_{ij}] = \begin{cases} 0 \text{ if symbol } j \text{ of every } i\text{-th branch is punctured} \\ 1 \text{ if symbol } j \text{ of every } i\text{-th branch is not punctured} \end{cases} \qquad (2)$$

To illustrate consider the rate 1/2 code whose trellis is shown in figure 1. If this code is punctured according to the perforation pattern

$$P = \begin{Vmatrix} 1 & 1 \\ 1 & 0 \end{Vmatrix} \qquad (3)$$

then we obtain a rate 2/3 code for which the trellis is shown in figure 2.

A convolutional encoder for a punctured code may be viewed as the combination of the convolutional encoder for the original code and a sampler that punctures the output sequence according to the perforation pattern. By simply changing the perforation pattern, one readily can change the coding rate of the resulting code.

The decoding of high rate punctured codes using either Viterbi decoding [5]-[7] or sequential decoding [8]-[10] is made easier because of the fact that the high rate code can be considered as if it were a low rate code. The perforation is taken into account by discarding or inhibiting the metric evaluation of the punctured symbols. Decoding on the low rate structure is much simpler since it involves only two new nodes at each step instead of 2^B as is normally the case for a (V,B) code.

III. Structure of punctured convolutional codes

In this section, we examine the structure of punctured convolutional codes. Specifically, we are interested in the structure of paths of the original and punctured codes and in the relationships that exist between them.

To investigate the structure of punctured codes, one has to go back to the process of generating such a code. Conceptually, a punctured code is obtained through two different operations: a) grouping B consecutive branches of a low rate code into "super branches" and b) deleting some code symbols from these super branches. The first operation, the grouping of branches, affects the way we look at the original code. The grouped super branches may be viewed as the output of the low rate (BV_0,B) encoder corresponding to the low rate original $(V_0,1)$ code, where B is the number of branches that are grouped together. Hence the paths of the original code go through three different stages leading to the generation of a high rate punctured code, namely: 1) original low rate $(V_0,1)$ code, 2) low rate

(BV_0, B) code and 3) punctured high rate (V,B) code. In order to distinguish between these stages, we introduce the following definitions.

Definitions:

Elementary code:	The low-rate $(V_0,1)$ original code.
Elementary branch:	A branch of the elementary code, consisting of V_0 symbols.
Elementary path:	A path of the elementary code, consisting elementary branches.
B-code:	The low-rate (BV_0,B) code obtained by grouping every B consecutive elementary branches of the elementary code in a "super branch".
B-branch:	A "super branch" of the B-code, consisting of BV_0 symbols.
B-path:	A path of the B-code, consisting of B-branches.
Punctured code:	The (V,B) code obtained following the perforation of symbols of the B-branches.
Punctured branch:	A branch of the punctured code, consisting of V symbols, $V < BV_0$.

To illustrate these definitions, the state diagram of the elementary code of figure 3 is shown in figure 4 and the state diagram of the corresponding 2-code is shown in figure 5. The diagram of figure 4 depicts elementary branches, elementary paths, etc., whereas the diagram of figure 5 depicts B-branches, B-paths, etc.

We now turn to the analysis of the structure of the paths of punctured codes. Consider the state diagram of a general convolutional code of memory M. A state in this diagram will be denoted by

$$S_i \; ; \quad i \in \{0,\dots,2^M-1\}.$$

A single transition from state S_i to state S_j will be denoted

$$S_{ij} \; ; \quad i,\ j \in \{0,\dots,2^M-1\}$$

and any path going through n transitions from state S_i to state S_j but without passing on state S_0 will be denoted

$$\underline{S}_{ij}^{(n)} \; ; \quad i,\ j \in \{0,\dots,2^M-1\}.$$

Finally, a path consisting of n consecutive identical single transitions S_{ij} will be denoted

$$(S_{ij})^n \; ; \quad i,\ j \in \{0,\dots,2^M-1\}.$$

In the state diagram a path that leaves the zero state (S_0) at some time t and remerges to this state for the first time at a later time (t + L) is said to be a remerging path of length L of the code. These paths play an important role in the determining of various Hamming distances of the code. We now state a theorem on the remerging paths of a B-code.

<u>Theorem 1</u>: To each remerging path in the trellis of an elementary code, there correspond at least B remerging paths in the trellis of the B-code, all comprising the elementary code sequence and possibly some additional "0" symbols.

PROOF: Consider the state diagram of a B-code, as illustrated in figure 5. A remerging path of length (n + 2) may be expressed as:

$$s_{0i}, \underline{S}_{ij}^{(n)}, s_{j0}, \quad \text{where } i, j \neq 0 \tag{4}$$

Observe that this path is described in terms of the states S_i of the B-code. The same path may also be expressed in terms of the underlying transitions between the states S_i of the elementary code. A single transition leaving state S_i and reaching state S_p translates as the sequence:

$$s_{ij}, \underbrace{s_{jk}, \ldots, s_{mn}}_{(B\,-\,2)\text{ transitions}}, s_{np} \tag{5}$$

where j, k, m and n $\in \{0, \ldots, 2^M-1\}$. Hence the remerging path can be translated as:

$$\underbrace{s_{0j}, \ldots,}_{B} s_{km}\underbrace{, \ldots,}_{B} s_{np}, \ldots, s_{qr}\underbrace{, \ldots,}_{B} s_{t0} \tag{6}$$

where each brace spans B state transitions. Notice that in this sequence, some of the states S_j may be allowed to be S_0, as long as none of them occurs on a boundary between the groups of B state transitions, since this would translate for the B-code as a transition to state S_0.

Let $s_{0i}, \underline{S}_{ij}^{(n)}, s_{j0}$ be a reconverging path of the elementary code. Then all the following paths translate as valid reconverging B-paths:

$$s_{0i}, \underline{s}_{ij}^{(n)}, s_{j0}, [s_{00}, \ldots, s_{00}]$$
$$s_{00}, s_{0i}, \underline{s}_{ij}^{(n)}, s_{j0}, [s_{00}, \ldots, s_{00}]$$
$$(s_{00})^2, s_{0i}, \underline{s}_{ij}^{(n)}, s_{j0}, [s_{00}, \ldots, s_{00}] \tag{7}$$
$$\vdots$$
$$(s_{00})^{B-1}, s_{0i}, \underline{s}_{ij}^{(n)}, s_{j0}, [s_{00}, \ldots, s_{00}]$$

In these expressions, the s_{00} transitions enclosed in brackets bring the total number of transitions to a multiple of B. The actual number of these transitions is different for every path and varies from 0 to B-1. For example, for the 2-code already considered, there are 2 2-paths corresponding to a given elementary path. The relationship between these 2 2-paths is illustrated in figure 6. Note the presence of s_{00} transitions at the beginning and at the end of the state transition sequence.

Since no s_{00} transition occurs on a boundary between the B-branches, all the B-paths are valid remerging paths, and clearly there are B distinct such B-paths. They all include the sequence $s_{0i}, \underline{s}_{ij}^{(n)}, s_{j0}$, which gives the original code sequence, and since the additional s_{00} transitions result in branches of V_0 "0" symbols, then all the B-paths consist of the original code sequence and some (maybe none) "0" symbols added at the beginning and at the end. Q.E.D

The lengths of the B-paths depend on the actual number of s_{00} transitions appended to the original sequence, and hence not all the B-paths have the same length. However, the lengths do not differ by more than 1, since no more than 2(B - 1) s_{00} transitions may be appended to a path.

We now examine certain relationships that exist between the coding rate of the original code and that of the resulting punctured code.

Theorem 2: A high-rate (V,B) punctured code may be obtained by puncturing an original low rate $(V_0,1)$ code. However, a punctured code obtained from a $(V_0,1)$ original code cannot, in general, be obtained from a $(V_0',1)$ code if $V_0' < V_0$.

PROOF: This can be demonstrated with a simple example. Without loss of generality, assume that a punctured rate 2/3 code is obtained from an original (3,1), memory 2 code, with generator polynomials $g_1(D)$, $g_2(D)$, $g_3(D)$, and using the perforation pattern

$$P = \begin{Vmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{Vmatrix}$$

Let the impulse response of the original (3,1) code be

$$g_1^0 \; g_2^0 \; g_3^0 \; g_1^1 \; g_2^1 \; g_3^1 \; g_1^2 \; g_2^2 \; g_3^2 \qquad (8)$$

The impulse response of the corresponding 2-code is written as

$$\begin{matrix} g_1^0 & g_2^0 & g_3^0 & g_1^1 & g_2^1 & g_3^1 & g_1^2 & g_2^2 & g_3^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & g_1^0 & g_2^0 & g_3^0 & g_1^1 & g_2^1 & g_3^1 & g_1^2 & g_2^2 & g_3^2 \end{matrix} \qquad (9)$$

and that of the resulting punctured code becomes

$$\begin{array}{cccccccc} g_1^0 & g_2^0 & g_3^1 & g_1^2 & g_2^2 & 0 & & \\ 0 & 0 & g_3^0 & g_1^1 & g_2^1 & g_3^2 & & \end{array} \tag{10}$$

Observe that all the non-zero entries in this impulse response are independent of each other. This is true in general of any (3,1) code used in this situation.

Now consider a (2,1) memory 2 code with impulse response:

$$g_1^0 \; g_2^0 \; g_1^1 \; g_2^1 \; g_1^2 \; g_2^2 \tag{11}$$

The impulse response of the corresponding 2-code would then be

$$\begin{array}{cccccccccc} g_1^0 & g_2^0 & g_1^1 & g_2^1 & g^{12} & g_2^2 & 0 & 0 & & \\ 0 & 0 & g_1^0 & g_2^0 & g_1^1 & g_2^1 & g_1^2 & g_2^2 & & \end{array} \tag{12}$$

Now in order to obtain a rate 2/3 code from this 2-code, one symbol must be punctured on every 2-branch. But regardless of the way this perforation is performed, the non-zero entries of the resulting impulse response will never be independent of each other. Therefore it is not possible in general to obtain the same punctured code as obtained from the (3,1) code. Observe that this property is due solely to the rates of the original codes, and does not depend at all on the memories of the original codes. Q.E.D.

Intuitively, we see that with a large V_0, there is more freedom in choosing the generators that will yield the particular punctured code, so that some punctured codes that are possible from a low rate $1/V_0$ code could not otherwise be obtained from a a code with a smaller V_0. This result suggests that a very large V_0 should be used to generate the best punctured code possible. However, there is a limit above which it becomes useless to increase the V_0 of the original code, as we shall see in the next theorem.

<u>Theorem 3</u>: To generate <u>any</u> (V,B) punctured code, the rate of the original code is no lower than that of a (V,1) code.

PROOF: Assume that some original $(V_0,1)$ code is punctured to make a (V,B) punctured code, with $V_0 > V$. The total number of symbols to be punctured in every B-branch of the B-code is $W = BV_0-V$. The possible perforation patterns are given by $B \times V_0$ binary matrices that have V "1" entries and W "0" entries. Therefore there will be at least $(V_0 - V)$ rows of the perforation matrix that contain no entry "1". These rows, and the corresponding generators of the original code, may be eliminated from the description of the punctured code without any effect on the resulting code, bringing the original code to a (V,1) code. Therefore, V generators are sufficient to obtain any possible (V,B) punctured code. Q.E.D.

We now introduce a class of perforation patterns called <u>orthogonal perforation patterns</u>. These patterns play an important role in the establishing of further results concerning punctured codes.

<u>Definition</u>: A perforation pattern in which the symbols that are not punctured on one elementary branch of the B-branch are punctured on every other elementary branch of the B-branch is called an <u>orthogonal perforation pattern</u>. For example, the following perforation pattern is orthogonal:

$$P_o = \begin{Vmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{Vmatrix} \tag{13}$$

Conversely, a non-orthogonal perforation pattern is a pattern in which the symbols of at least one modulo-2 adder are used in more than one of the B elementary branches of the B-branch. An orthogonal perforation pattern is an "efficient" pattern in the sense that it uses all the potential (different generators) of the original code. The following theorem establishes the theoretical importance of the orthogonal perforation patterns for determining any punctured code.

Theorem 4: Any punctured code can be obtained by means of an orthogonal perforation pattern.

PROOF: Assume, without loss of generality (see Theorem 3) that a (V,1) original code is used to obtain a (V,B) punctured code. Let $g_j(D)$, $j = 1, 2, \ldots, V$ be the generator polynomials of the original code. Assume further that the perforation pattern P_{no} used is non-orthogonal. Then, one or more of the columns of the perforation matrix contain more than one entry "1". But P_{no} is a V columns matrix with a total of V entries "1". Thus, at least one of the columns of P_{no} that has no entry "1".

Assume for the moment that only column k contains no entry "1" and that column m contains 2 entries "1". (This will be generalized later.) The (V,B) punctured code obtained from P_{no} and this original code may be obtained with an orthogonal perforation pattern provided that we select another original code.

The new original code to use is identical to the one above except for generator $g_k(D)$. Since P_{no} never uses the symbols from $g_k(D)$ we can change $g_k(D)$ without changing the resulting punctured code. Suppose we substitute a copy of $g_m(D)$ for $g_k(D)$. The new original code, together with P_{no}, would yield the same punctured code as before.

We now transform P_{no} into an orthogonal perforation pattern, P_o. P_o will be identical to P_{no} except for columns k and m. Column k of P_o will have one entry "1" and column m will have only one entry "1", the second entry "1" of column m being "replaced" by the new entry in column k. Hence, instead of using $g_m(D)$ twice, the new pattern will use it only once, using the new generator $g_k(D)$ for the second symbol. After a possible shuffling of the columns of P_o and of the generators of the new original code to preserve the order of the code symbols, the resulting punctured code will be exactly the same as before.

This result may be generalized easily if we consider the following extensions. First, if more than one column has 2 entries "1", the construction given above may be applied separately to each column. Second, if some columns have more than 2 entries "1", then the construction can also be applied for each one of the "1" entries in excess of one, hence proving the theorem. Q.E.D.

Observe that the converse is generally not true: one cannot obtain all the possible (B,V) punctured codes with non-orthogonal perforation patterns. The class of orthogonal perforation patterns is thus complete in the sense that all the punctured codes may be obtained by using members of this class.

IV. Distance properties of the punctured paths

We have already seen that every elementary remerging path of the original code gives rise to B distinct B-paths for the B-code. Let us now see how puncturing the code affects the Hamming weights of these remerging paths. Since we are dealing with linear convolutional codes, these Hamming weights will yield the different Hamming distances of the punctured codes.

Consider a remerging B-path of L B-branches of the B-code corresponding to a $(V_0,1)$ original code. In order to determine the weight of this B-path after perforation, it is useful to consider separately the contributions to the total weight of symbols occupying each one of the BV_0 positions of the B-branches.

Let us form a vector $\underline{D}$ of distance contributions of a remerging B-path. Each of the BV_0 components of this vector will be the sum of the weights of the symbols occupying the corresponding position in the B-branches. Formally, let x_i be the i-th element of the encoded sequence $\underline{X}$ of the B-path of length L B-branches, $i = 0, 1, \ldots, LBV_0 - 1$. The components d_K of $\underline{D}$ are then given by

$$d_K = \sum_{i \equiv k \pmod{BV_0}} x_i \quad ; \; k = 0, \ldots, BV_0 - 1. \tag{14}$$

We may also represent the perforation pattern by a vector $\underline{P}$ of BV_0 binary components p_i given by

$$p_i = \begin{cases} 0 & \text{if the i-th symbol of the B-branch is punctured} \\ 1 & \text{if the i-th symbol of the B-branch is not punctured} \end{cases} \tag{15}$$

$\underline{P}$ is in fact the concatenation of all the rows of the perforation matrix.

The total Hamming distance D resulting from the perforation of a B-path $\underline{X}$ with distance contribution vector $\underline{D}$ by the perforation pattern $\underline{P}$ is then given by the scalar product:

$$D = \underline{D}\cdot\underline{P} \tag{16}$$

Using this formulation, we consider once again the B B-paths corresponding to an elementary remerging path $\underline{X}$ according to Theorem 1. Let us denote the B-paths $\underline{X}^r$, where r is the number of additional S_{00} transitions at the beginning of the code sequence (r = 0, 1, ...,B - 1). Due to these S_{00} transitions, the code symbols of the elementary code sequence $\underline{X}$ appear in different positions within the B-paths. For these symbols, we have:

$$x^r_{j+rVo} = x^t_{j+tVo} \quad ; \quad r,\ t = 0,\ \ldots,\ B-1. \tag{17}$$

Since these are the only symbols that contribute to the Hamming distances, we can relate the components of vector $\underline{D}^0$ and $\underline{D}^r$, corresponding respectively to the B-paths $\underline{X}^0$ and $\underline{X}^r$ as follows

$$\begin{aligned} d^0_k &= \sum_{i\equiv k \pmod{BVo}} x^0_i \\ &= \sum_{i\equiv k \pmod{BVo}} x^r_{i+rVo} \\ &= \sum_{i\equiv k+rVo \pmod{BVo}} x^r_i \\ &= d^r_{k+rVo \pmod{BVo}} \end{aligned} \tag{18}$$

and therefore

$$d^0_k = d^r_{k+rVo \pmod{BVo}} = d^t_{k+tVo \pmod{BVo}} \tag{19}$$

Equation (19) means that all the B distance contribution vectors of the B-paths corresponding to a given elementary path have the same components, these components being cyclically shifted by groups of Vo from one vector to the next.

The first consequence of this observation is that perforation patterns which are likewise shifted versions of each other will yield the same ensemble of B total distances for the B punctured paths corresponding to a given elementary path. Hence, all the punctured codes obtained from these different perforation patterns with the same original code have the same distance properties: free distance, distance profile, etc. The perforation patterns are thus said to be equivalent.

The second consequence of this observation will be the formulation of an upper bound on the free distance of punctured codes. Assume $\underline{D}^0$, $\underline{D}^1, \ldots, \underline{D}^{B-1}$ are the B distance contribution vectors of the B B-paths corresponding to an elementary remerging path of Hamming distance D_{org} of an original (V,1) code. P_o is an orthogonal perforation pattern used to obtain a (V,B) punctured code.

Since Vo = V, each column of P_o contains exactly one entry "1". Let $\underline{P}_o$ be the vector form of P_o, and let $\underline{P}_o{}^0$, $\underline{P}_o{}^1, \ldots, \underline{P}_o{}^{B-1}$ be the B cyclic shifts of this perforation pattern (but shifted in the opposite direction from the distance contribution vectors). We can write

$$\sum_{k=0}^{B-1} \underline{P}_o{}^k = \underline{P}_o{}^0 + \underline{P}_o{}^1 + \ldots + \underline{P}_o{}^{B-1} = (11\ldots1) \qquad (20)$$

where (11...1) is the all "1" vector. The total distances of the B punctured paths are

$$D^0 = \underline{P}_o \cdot \underline{D}^0$$
$$D^1 = \underline{P}_o \cdot \underline{D}^1$$
$$\vdots$$
$$D^{B-1} = \underline{P}_o \cdot \underline{D}^{B-1} \tag{21}$$

or, equivalently,

$$D^0 = \underline{P}_o{}^0 \cdot \underline{D}^0$$
$$D^1 = \underline{P}_o{}^1 \cdot \underline{D}^0$$
$$\vdots$$
$$D^{B-1} = \underline{P}_o{}^{B-1} \cdot \underline{D}^0 \tag{22}$$

then, for the sum of the B punctured distances,

$$\begin{aligned} D^0 + D^1 + ..+ D^{B-1} &= \underline{P}_o{}^0 \cdot \underline{D}^0 + \underline{P}_o{}^1 \cdot \underline{D}^0 + \ldots + \underline{P}_o{}^{B-1} \cdot \underline{D}^0 \\ &= (\underline{P}_o{}^0 + \underline{P}_o{}^1 + \ldots + \underline{P}_o{}^{B-1}) \cdot \underline{D}^0 \\ &= (11\ldots1) \cdot \underline{D}^0 \\ &= D_{org} \end{aligned} \tag{23}$$

We can therefore conclude that the largest punctured distances satisfy the bound

$$D \leq \frac{D_{org}}{B} \tag{24}$$

Now if the original path is such that $D_{org} = D_{fo}$, that is, the original path is at free distance D_{fo}, then we obtain the upper bound

$$D_{fp} \leq \frac{D_{fo}}{B} \tag{25}$$

where D_{fp} is the free distance of the resulting punctured code.

To generalize this result, let us recall the hypothesis that lead to the formulation of this upper bound. First, the original code has to be a

(V,1) code for a (V,B) punctured code. This condition is easily relaxed by virtue of Theorem 3 which states that any (V,B) punctured code may be obtained from a (V,1) original code. The second hypothesis is that the perforation pattern is orthogonal. This condition may also be relaxed if we recall that any punctured code may be obtained using an orthogonal perforation pattern, provided the original code is changed (see theorem 4). Since changing the original code could change the value of D_{fo}, in order to extend the validity of (25), we replace D_{fo} by a tight upper bound on the free distance of an original code of the same memory and coding rate. We thus obtain the following

<u>Theorem 5:</u>

The free distance of a (V,B) punctured convolutional code is upper bounded by

$$D_{fp} \leq \frac{\langle D_{fo} \rangle}{B} \tag{26}$$

where $\langle D_{fo} \rangle$ is an upper bound on the free distance of any convolutional code of the same rate and memory as the original code.

As a final remark, this upper bound agrees with bounds obtained without the hypothesis of a punctured code structure [11]. Therefore, we may conclude that punctured codes are not necessarily worse than normal high rate codes, at least not in the sense of having a lower free distance.

V. Finding good long memory high rate punctured codes

In a separate paper [10], we have considered the search for good high rate punctured codes with long memory for use with sequential decoding. The basic approach consists of selecting the best known rate 1/2 code of a given memory as original code and deriving several punctured codes for different coding rates from this same original code. This approach is very attractive, especially for variable rate coding-decoding, and it has indeed yielded a number of good punctured codes suitable for these applications [6], [7], [10]. However, these codes are not in general optimal: they do not meet or come close to the upper bound on the free distance, and furthermore they are not as good as the best known high rate codes. This observation is further supported by the theoretical results obtained here concerning the incidence of the rate of the original code and of the type of perforation pattern to be used on the resulting punctured code. Intuitively, a (V,1) original code should be used in conjunction with an orthogonal perforation pattern as to provide as much freedom as possible in the specification of a given (V,B) punctured code.

A natural question arises: can optimal (or nearly optimal) high rate codes be obtained by perforation? This question has lead us to investigate the conditions under which a given high rate code can be obtained by perforating a low rate code. In order to establish these conditions, we first recall the difference between an arbitrary high rate code and a punctured high rate code.

An encoder for an arbitrary (V,B) code is a B-input/V-output machine: at each encoding cycle, B information bits enter the machine and V code symbols are delivered to the output. Any one of the V output symbols may thus depend on any of the B input bits, as well as on the state of the machine.

With a punctured code, the information bits actually enter the encoder one at a time and the output symbols are produced by small groups of Vo. Since an encoder must be a causal machine, certain dependencies are thus forbidden between the input and the output streams. No output symbol at time t_K may depend on an input bit at time $t_{K+\partial}$, $\partial > 0$, that is, no output symbol can depend on an input bit that has not yet entered the encoder.

This constraint may be translated in terms of the impulse response of the high rate code as follows. Let g_{ij} be the components of the impulse response of the encoder. Then

$$g_{ij} = 0 \text{ for } i > j \;;\; i = 1, \ldots, B;\; j = 0, 1, \ldots, m_i \qquad (27)$$

insures that the corresponding high rate code respects the causality constraint, and thus, it may be obtained by perforation of a low rate code.

It is therefore an easy task to verify that a given high rate code respects this condition, by the simple observation of its impulse response. Once this verification is performed, an original code that will yield this particular high rate code may be constructed. In order to get all the possible freedom in the choice of the generators, the original code that must be used to obtain a (V,B) code will be a (V,1) code, punctured by an orthogonal perforation pattern.

The choice of the specific perforation pattern to use is dictated by the impulse response of the target high rate code. If, for example, B = V-1, as is frequently the case for a great number of high rate codes, then the perforation matrix has V - 1 rows and V columns, so one of the rows will contain 2 entries "1". The specification of which rows has 2 entries "1" specifies the perforation to use. For sequential decoding purposes, perforation patterns that have 2 entries "1" on the first row are desirable, since they will tend to yield codes with a rapidly increasing column distance function [12]. Such a perforation pattern may be selected if the impulse response satisfies the stronger condition:

$$g_{ij} = 0 \text{ for } i \geq j \text{ ; } i = 2, \ldots, B; \; j = 0, 1, \ldots, m_i \tag{28}$$

Otherwise, a perforation pattern with only entry "1" on the first row must be selected.

Construction of the original code

The construction of the original (V,1) code that will yield a specific (V,B) code through perforation is easy to perform. Let $\underline{L}_1$ through $\underline{L}_B$ denote the B rows of the impulse response of the target code. Let K_B be the length of this code:

$$K_B = 1 + \max \{m_i\} \text{ ; } i = 1, 2, \ldots B \tag{29}$$

where m_i is the memory of the i-th shift register of the (V,B) encoder. The B rows of the impulse response are vectors of K_BV components. The construction algorithm involves the following steps:

1. Select the perforation pattern P according to equation (28).

2. Form vector $\underline{P}_r$ by repeating the vector form of the perforation pattern $\underline{P}$ K_B times. For example let

$\underline{P}$ = 1 0 0 0 1 1 and $K_B = 2$
then $\underline{P}_r$ = 1 0 0 0 1 1 1 0 0 0 1 1

3. "Expand" the rows $\underline{L}_j$, j= 1, 2, ... B, into rows $\underline{L}_{je}$ according to $\underline{P}_r$. The "expansion" is done through the following algorithmic procedure:

   For each component i of $\underline{P}_r$ do:

   Begin

   a) If component i of $\underline{P}_r$ = "0", component i of $\underline{L}_{je}$ = "0".

   b) If component i of $\underline{P}_r$ = "1", component i of $\underline{L}_{je}$ = next unused component of $\underline{L}_j$. Mark this component of $\underline{L}_j$ as used.

   End

5. For all rows j, row $\underline{L}_{je}$ is shifted (j-1)V positions to the left. The first row is not shifted, the second row is shifted V positions to the left, etc. Components are lost to the left and "0" are inserted to the right.

6. Sum the B resulting rows together (component-wise).

7. The resulting vector is the impulse response of the desired original (V,1) code.

VI. Good long punctured codes

The above procedure has been used to derive the original codes which yield through perforation the best known rate 2/3 and 3/4 codes of memory $3 \leq M \leq 23$ and $3 \leq M \leq 9$ respectively.

Non-systematic codes:

We have found that all the best known high rate non-systematic codes for rate 2/3 and 3/4 ([13], [14]) satisfy condition (27). We have thus constructed the low rate original codes corresponding to all of these high rate codes. They are listed in Table 1. As expected, the (3,2) codes are obtained by puncturing (3,1) codes and the (4,3) codes are obtained from (4,1) codes. Lack of knowledge of other good long memory high rate codes for other coding rate and memories has prevented us from expanding this list further.

It is interesting to observe that, for most of the cases, the memory of the required original code is larger than that of the resulting punctured code. This is illustrated in figure 7 where the relationship between original and punctured memories for the punctured (3,2) codes of Table 1 is plotted. The memory difference is quite small, usually one or two, and it is independent of the overall memory of the code. Therefore, its importance decreases as the memory of the code increases. For large memory codes, this memory increment is of no consequence whatsoever since these codes are to be decoded by sequential decoding methods.

Although the punctured codes found here are not suitable for variable rate decoding, they do provide a practical method of using the best known codes of rate 2/3 and 3/4 with sequential decoding. The decoding of these codes by the normal (non punctured) approach is difficult because of the large number (2^B) of nodes stemming from a single node in the tree structure of a high rate (V,B) code. However, with the punctured codes approach, the decoding proceeds on the low rate structure, so the number of nodes stemming from a single node is always 2, regardless of the actual coding rate. This regularity of the decoding operation is a very desirable feature in decoder implementations.

The existence of punctured codes that duplicate the best known high rate codes is a good indication of the validity of the punctured approach to the generation of high rate codes. It certainly invalidates the claim that punctured codes are necessarily sub-optimal. Figures 8 and 9 plot the free distances of some of the rate 2/3 and 3/4 codes found in comparison with the upper bound of (26), as a function of the memory length. It clearly shows that these codes do achieve a good free distance.

original R = 1/3				punctured R = 2/3			
M o	G1	G2	G3	M	G11 G12	G21 G22	G31 G32
4	26	22	35	3	6 1	2 4	4 7
5	54	47	67	4	6 1	3 5	7 5
6	172	137	152	5	14 07	06 17	16 10
7	314	271	317	6	12 05	05 16	13 13
8	424	455	747	7	26 00	14 23	32 33
9	1634	1233	1431	8	32 13	05 33	25 22
10	3162	2553	3612	9	54 25	16 71	66 60
11	6732	4617	7153	10	53 36	23 53	51 67
12	17444	11051	17457	11	162 064	054 101	156 163

Perforation pattern $P = \begin{Vmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{Vmatrix}$

Table 1. Original codes that yield the (3,2) codes of Johannesson and Paaske [13] (in octal).

original R = 1/3				punctured R = 2/3			
M_0	G1	G2	G3	M	G11 G12	G21 G22	G31 G32
16	377052	221320	314321	12	740 367	260 414	520 515
16	274100	233221	331745	13	710 140	260 545	670 533
16	163376	101657	153006	14	337 127	023 237	342 221
16	370414	203175	321523	15	722 302	054 457	642 435
18	1277142	1144571	1526370	16	1750 0165	0514 1235	1734 1054
18	1066424	1373146	1471265	17	1266 0140	0652 1752	1270 1307
19	2667576	2153625	3502436	18	1567 0337	0367 1230	1066 1603
20	4600614	4773271	6275153	19	2422 0412	1674 2745	2356 2711
20	12400344	13365473	15646505	20	3414 0005	1625 3367	3673 2440
22	24613606	22226172	35045621	21	6562 0431	2316 4454	4160 7225
24	117356622	126100341	151373474	22	13764 03251	02430 16011	14654 11766
24	106172264	130463065	141102467	23	12346 01314	05250 14247	10412 11067

Perforation pattern $P = \begin{Vmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{Vmatrix}$

Table 1.(cont.) Original codes that yield the (3,2) codes of Johannesson and Paaske [13] (in octal).

original R = 1/3				punctured R = 2/3			
M_0	G1	G2	G3	M	G11 G12	G21 G22	G31 G32
3	16	11	15	2	3 1	1 2	3 2
4	22	22	37	3	4 1	2 4	6 7
6	72	43	72	4	7 2	1 5	4 7
7	132	112	177	5	14 03	06 10	16 17
8	362	266	373	6	15 06	06 15	15 17
9	552	457	736	7	30 07	16 23	26 36
10	2146	2512	3355	9	52 05	06 70	74 53
11	7432	5163	7026	10	63 32	15 65	46 61

Perforation pattern $P = \begin{Vmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{Vmatrix}$

Table 2. Original codes that yield the (3,2) codes of Paaske [14] (in octal).

original	R = 1/4				punctured R = 3/4				
M_0	G_1	G_2	G_3	G_4	M	G G11 G12 13	G G21 G22 23	G G31 G32 33	G G41 G42 43
6	100	170	125	161	3	4 0 0	4 6 2	4 2 5	4 4 5
7	224	270	206	357	5	6 1 0	2 6 2	2 0 5	6 7 5
8	750	512	446	731	6	6 3 2	1 4 3	0 1 7	7 6 4
10	2274	2170	3262	3411	8	16 03 01	06 12 02	04 00 17	10 13 10
11	6230	4426	4711	7724	9	10 01 07	03 15 00	07 04 14	14 16 15

Perforation pattern $P = \begin{Vmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{Vmatrix}$

Table 3. Original codes that yield the (4,3) codes of Paaske [14] (in octal).

Systematic codes:

It is quite clear from the very definition of systematic codes that they all meet condition (27). Thus, all systematic codes, whether known or to be discovered, may be obtained by puncturing a low rate original (systematic) code. Furthermore, for any (V,B) target code, the original code need only be a (2,1) systematic code since the information is conveyed by B code symbols, and any parity symbol may be obtained from the B remaining code symbols of the B-branches.

The construction technique for deriving the original codes from the target codes is a simple adaptation of the one presented for non-systematic codes. We have thus obtained original codes for all the systematic codes of Hagenauer [15]. These codes are listed in Table 4.

The possibility of generating these codes by perforation allows once again their easy and practical decoding by sequential decoding. For instance, it would be quite impractical to decode a rate 7/8 code by the straightforward sequential decoding approach. In contrast, by the punctured approach, this code may be decoded as simply as a rate 1/2 code.

Just like the non-systematic codes, the systematic punctured codes found here do not readily lend themselves to variable rate decoding. However, families of good punctured systematic codes with different coding rate could be obtained from single original systematic codes.

R	P^T	G_2
2/3	1 1 0 1	33275606556377737
3/4	1 1 1 0 0 1	7567302467170307 74725
4/5	1 1 1 1 0 0 0 1	74754644665211334567 25475223
5/6	1 1 1 1 1 0 0 0 0 1	1717511311712277223 3670106777
7/8	1 1 1 1 1 1 1 0 0 0 0 0 0 1	17736344537740145413754 37553121

All G_1 = 100...0 and G_2 is given in octal (right justification).

Table 4. Systematic punctured codes obtained from original (2,1) codes.

VII. Conclusion

In this paper, a number of properties of high rate punctured convolutional codes have been presented. These properties provide a better understanding of this special class of high rate codes and give useful indications for guiding the search for good punctured codes. A Relationship between the paths of the original code and those of the resulting punctured code have been established. An upper bound on the free distance of punctured codes has been derived, indicating that punctured codes are indeed good codes.

Furthermore, conditions that insure that a given high rate code may be obtained by perforating a low rate code have been formulated. A construction procedure has been established for deriving the low rate original code corresponding to a target high rate code. Using this procedure, a number of good long punctured codes that are suitable for sequential decoding have been found. These codes duplicate the best known usual high rate codes of same memory length.

VIII. References

[1] V. Bhargava, D. Haccoun, R. Matyas, P. Nuspl, *Digital communications by satellite*, Wiley, New York, 1981.

[2] W. W. Wu, *Elements of Digital satellite Communication*, Computer Science Press, Rockville, 1984, vol. I and II.

[3] J. H. Yuen, *Deep Space Telecommunications Systems Engineering*, Plenum Press, New York, 1983.

[4] W. Wu, D.Haccoun, Y. Hirata and R. Peile, "Coding for Satellite Communication," *IEEE Journal on Selected Areas in Communications*, vol. SAC-5, No. 4, pp. 724-748, May 1987.

[5] J. B. Cain, G. C. Clark and J. Geist, "Punctured Convolutional Codes of Rate (n-1)/n and Simplified Maximum Likelihood Decoding," *IEEE Trans. Inform. Theory*, vol. IT-25, pp. 97-100, January 1979.

[6] Y. Yasuda, Y. Hirata, K. Nakamura, S. Otani, "Development of a Variable-Rate Viterbi Decoder and its Performance Characteristics," *6th International Conference on Digital Satellite Communications*, Phoenix, September 1983.

[7] Y. Yasuda, K. Kashiki, Y. Hirata, "High-Rate Punctured Convolutional Codes for Soft Decision Viterbi Decoding," *IEEE Trans, Commun.*, vol. COM-32, pp. 315-319, March 1984.

[8] G. Bégin and D. Haccoun, "Sequential Decoding of Punctured Convolutional Codes," *Proceedings of the 13th Biennial Symposium on Communications*, Kingston, Ontario, June 1986, pp. A.3.5-A.3.8.

[9] G. Bégin and D. Haccoun, "Decoding of Punctured Convolutional Codes by the Stack Algorithm," *Abstracts of Papers, 1986 IEEE International Symposium on Information Theory*, Ann Arbor, Michigan, October 1986, p. 159.

[10] D. Haccoun and G. Bégin, "Punctured Codes for Sequential Decoding, " to be published.

[11] J. A. Heller, "Sequential Decoding: Short Constraint Length Convolutional Codes," in *Jet Propul. Lab. Calif. Inst. Tech. Pasadena, Space Programs Summary 37-54*, vol. 3, pp.171-174, Dec. 1968.

[12] J. L. Massey and D. J. Costello, "Nonsystematic Convolutional Codes for Sequential Decoding in Space Applications," *IEEE Trans. Commun. Technol.*, vol COM-19, pp. 806-813, Oct. 1971.

[13] R. Johannesson and E. Paaske, "Further Results on Binary Convolutional Codes with an Optimum Distance Profile," *IEEE Trans. Inform. Theory*, vol. IT-24, pp. 264 268, March 1978.

[14] E. Paaske, "Short Binary Convolutional Codes with Maximum Free Distance for Rate 2/3 and 3/4", *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 683-689, September 1969.

[15] J. Hagenauer, "High Rate Convolutional Codes with Good Profiles," *IEEE Trans. Inform. Theory*, vol. IT-23, pp. 615-618, September 1977.

LIST OF FIGURE CAPTIONS

LIST OF TABLES

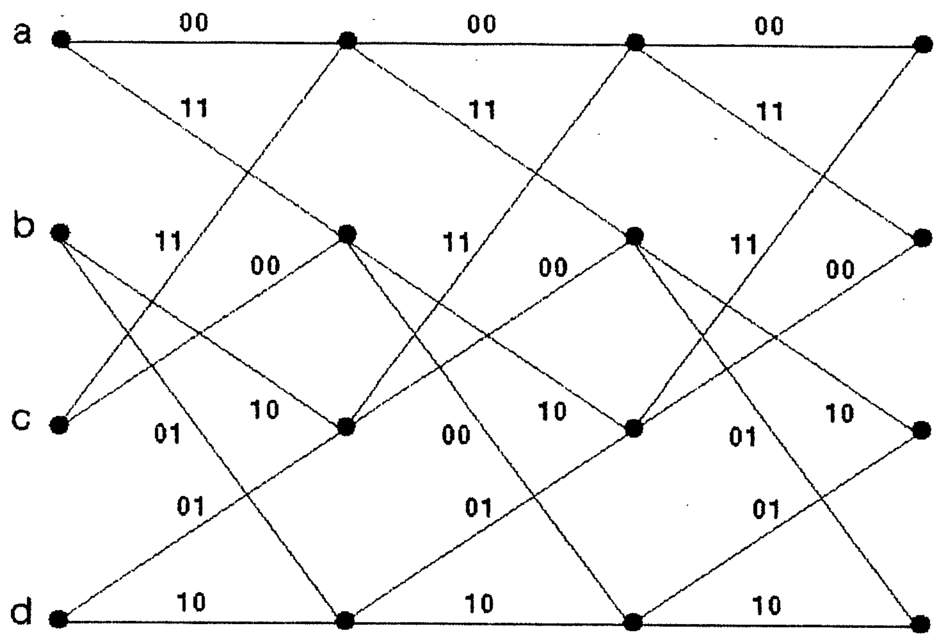

Figure 1. Trellis of a rate 1/2 convolutional code.

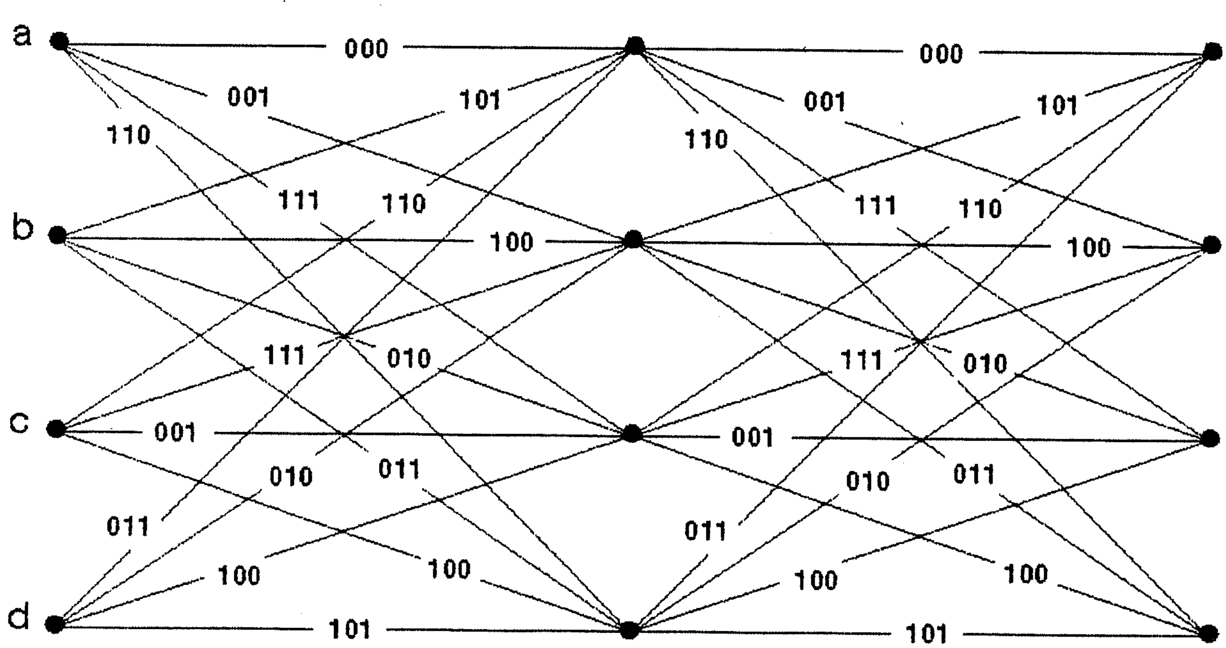

Figure 2. Trellis of the rate 2/3 punctured convolutional code.

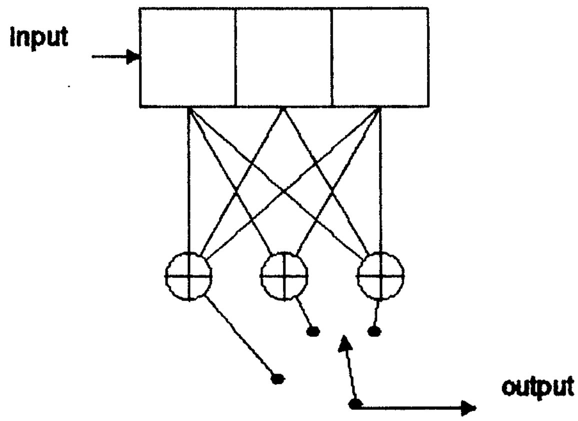

Figure 3. A encoder for a (3,1) memory 2 original code.

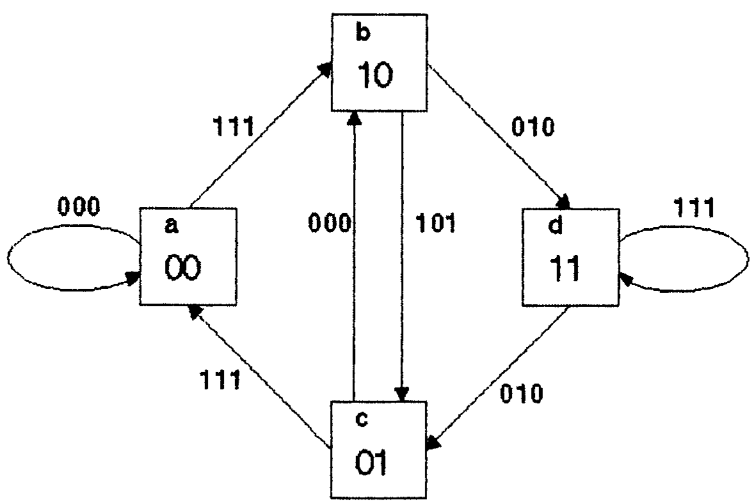

Figure 4. State diagram of the elementary code produced by the encoder of figure 3.

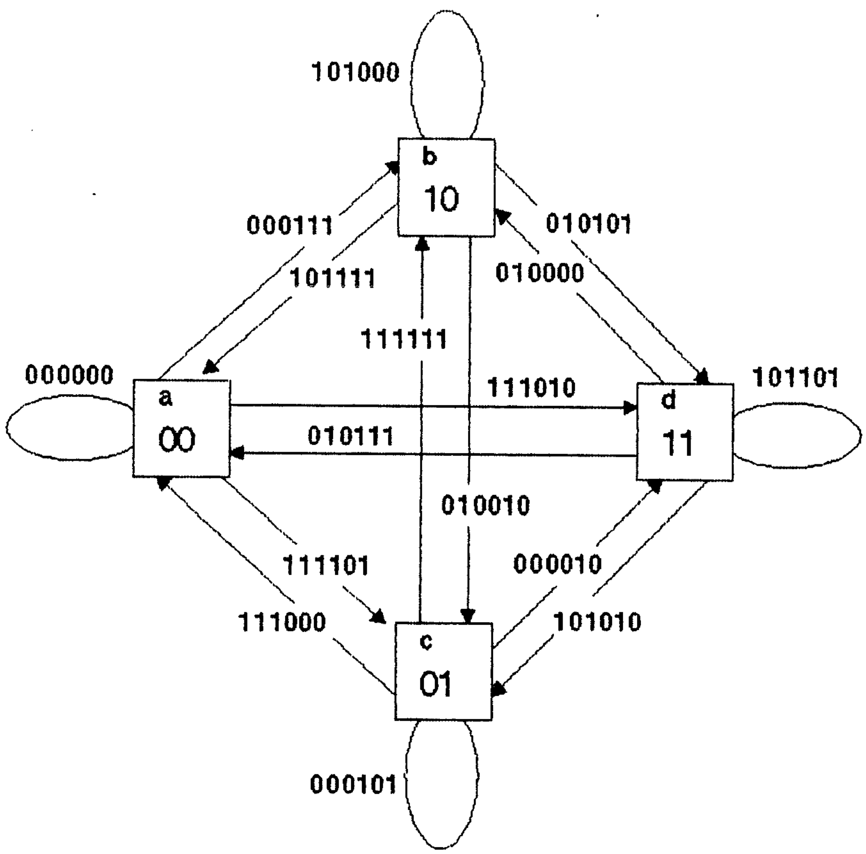

Figure 5. State diagram of the 2-code produced by the encoder of figure 3.

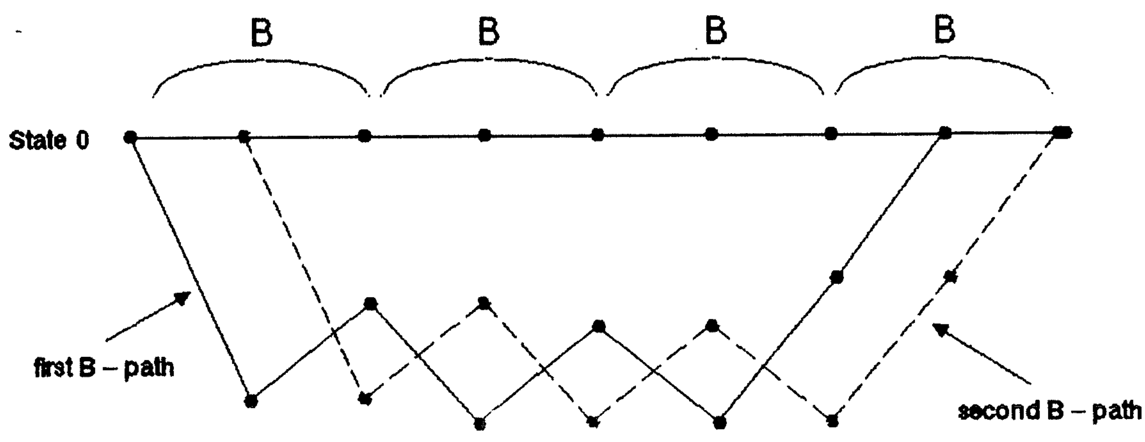

Figure 6. Relationship between 2-paths corresponding to the same elementary remerging path.

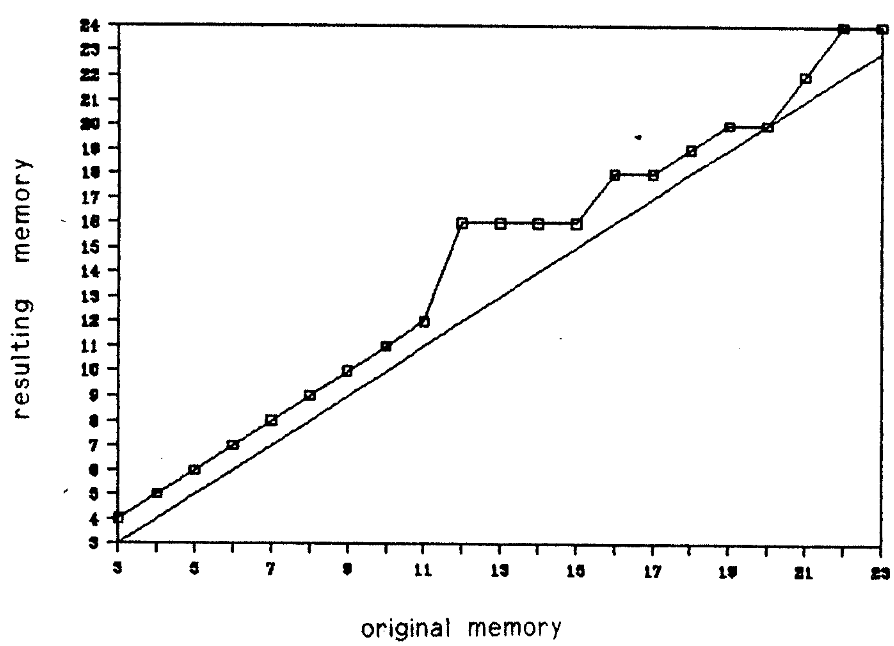

Figure 7. Relationship between original and punctured memories for the punctured (3,2) codes of Table 1.

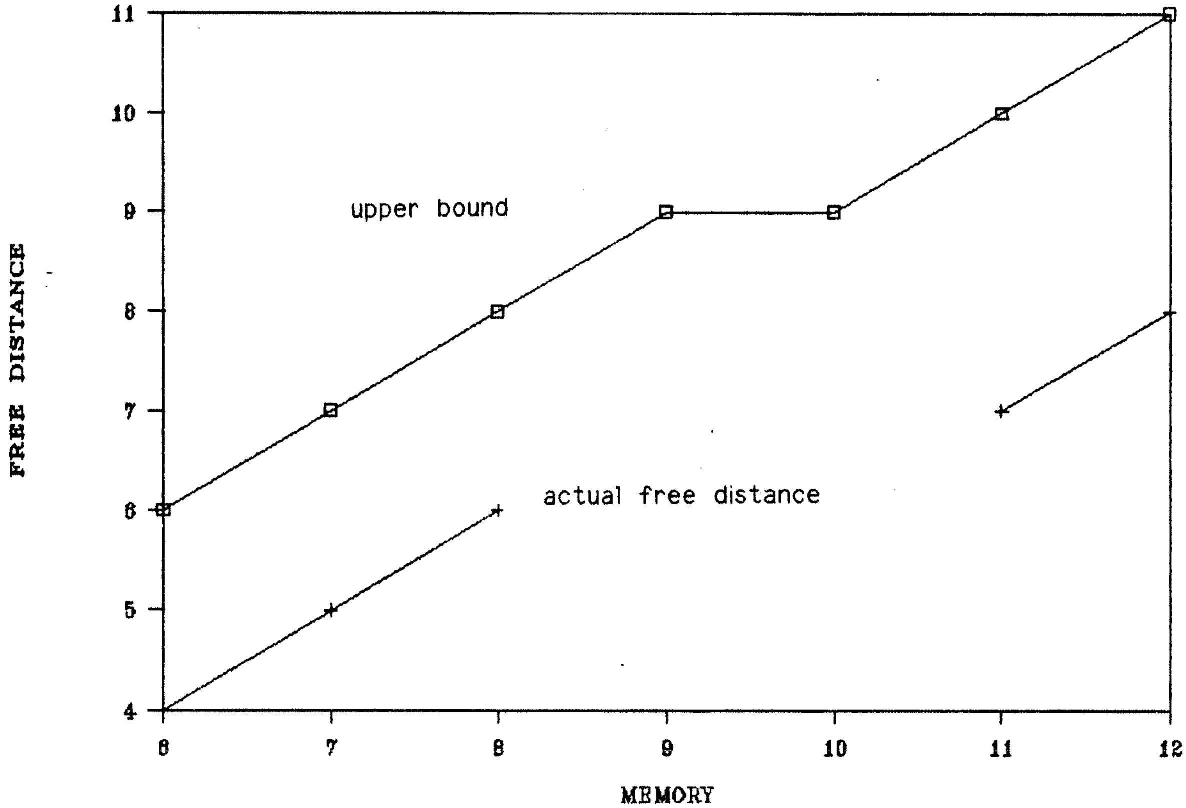

Figure 8. Free distances of some of the rate 2/3 codes found in comparison with the upper bound of (26), as a function of the memory.

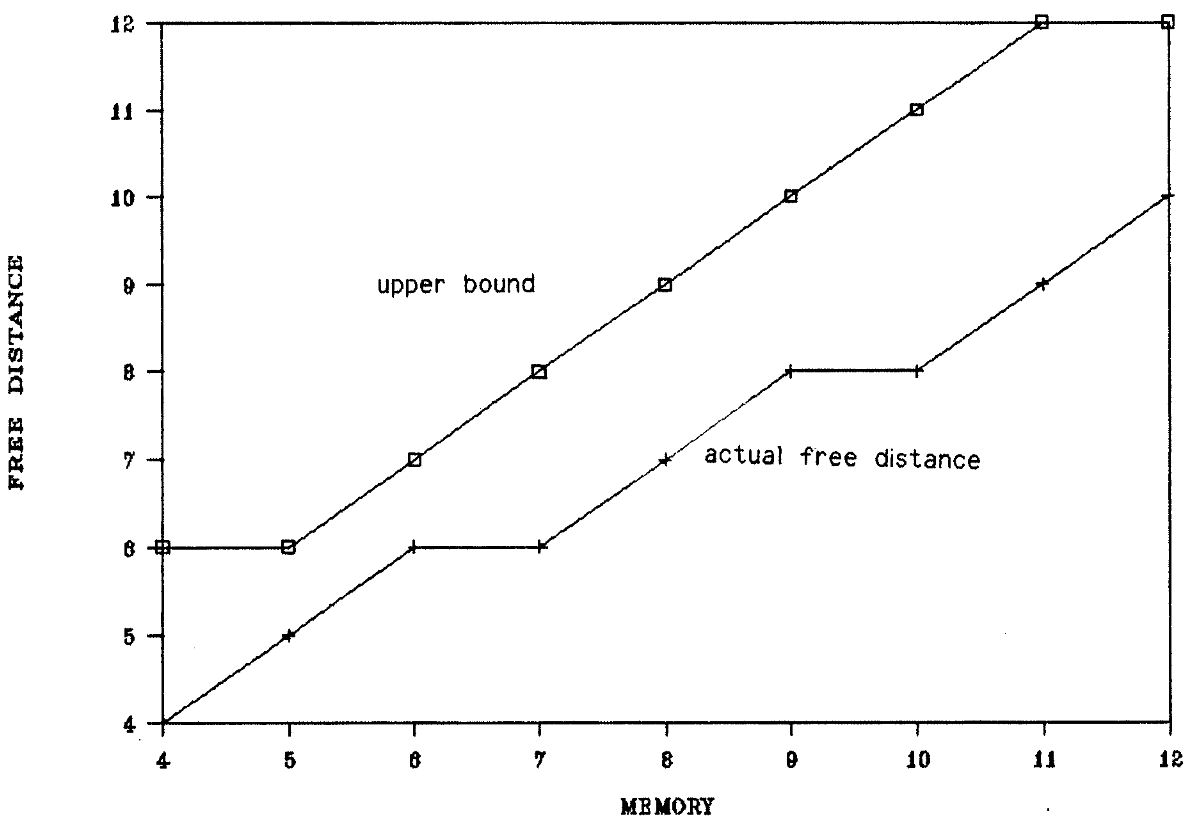

Figure 9. Free distances of some of the rate 3/4 codes found in comparison with the upper bound of (26), as a function of the memory.

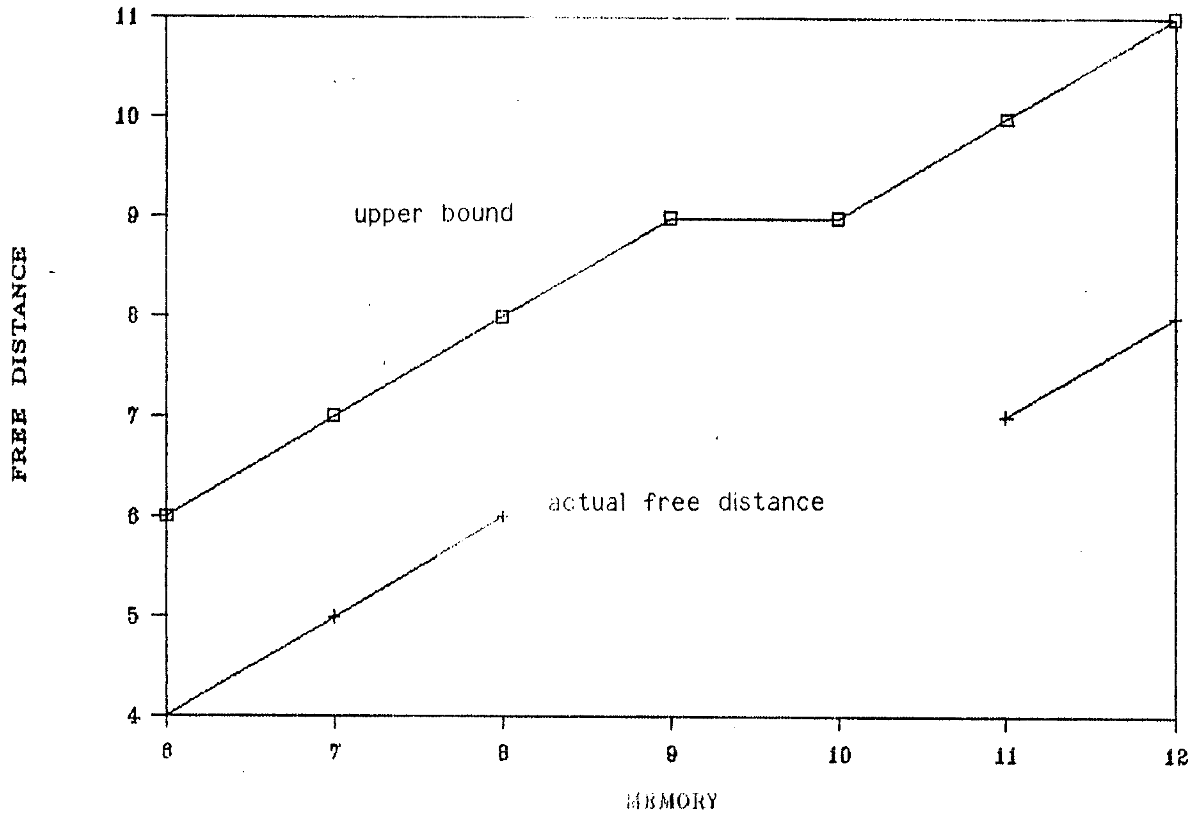

Figure 8. Free distances of some of the rate 2/3 codes found in comparison with the upper bound of (26), as a function of the memory.

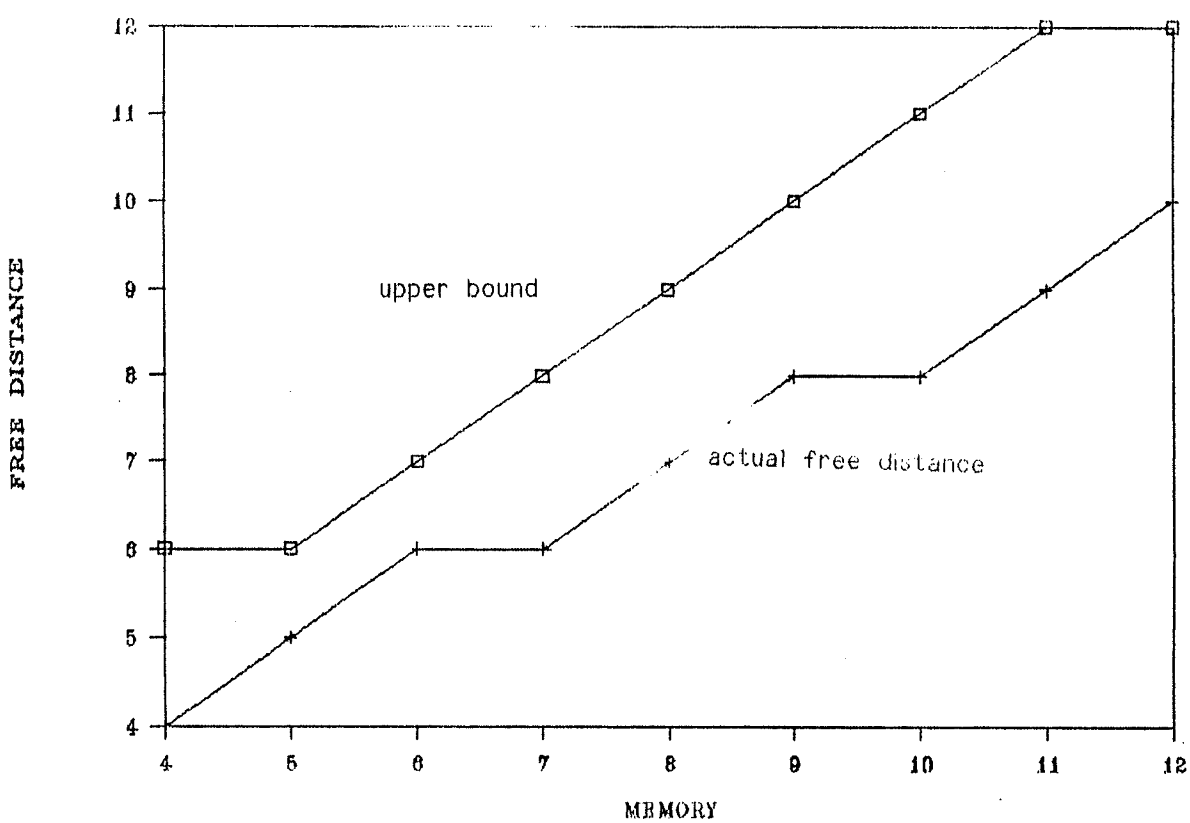

Figure 9. Free distances of some of the rate 3/4 codes found in comparison with the upper bound of (26), as a function of the memory.

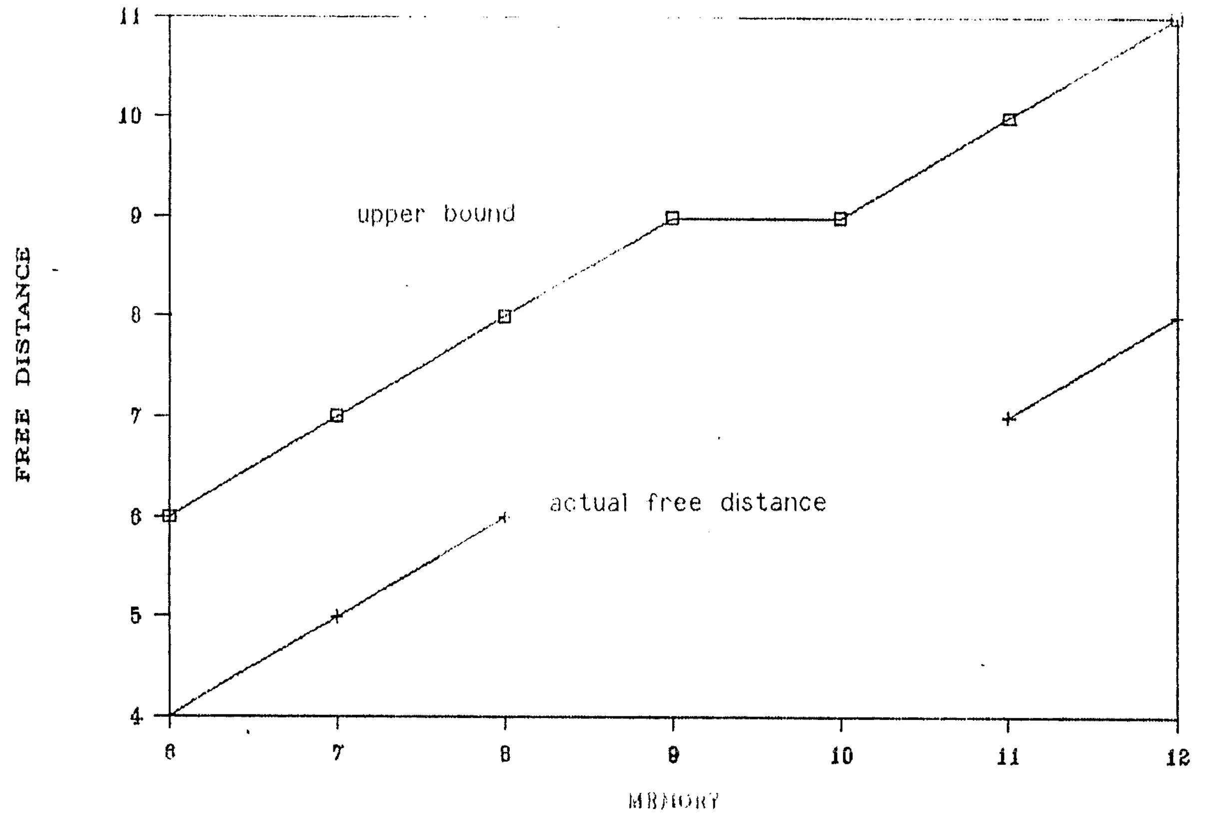

Figure 8. Free distances of some of the rate 2/3 codes found in comparison with the upper bound of (26), as a function of the memory.

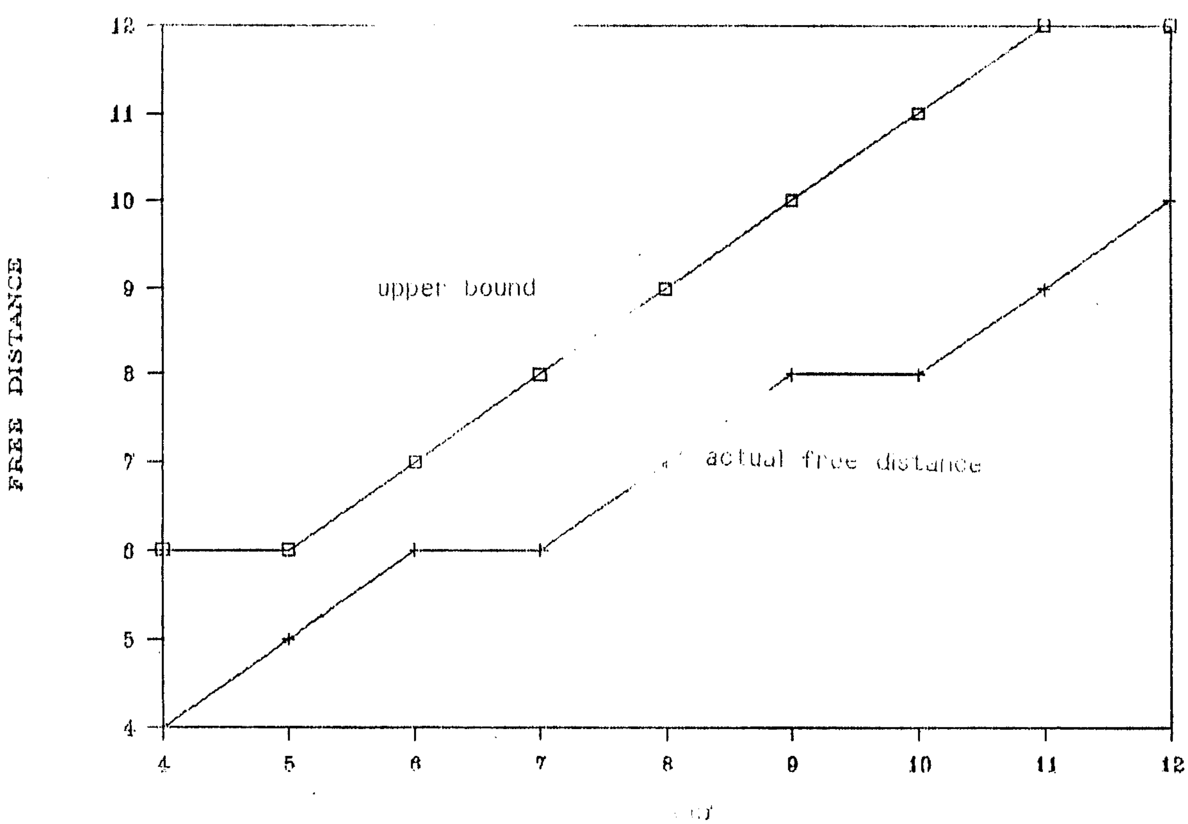

Figure 9. Free distances of som[illegible] 3/4 codes found in compari[illegible] with the upper bound [illegible] as a function of the memory.

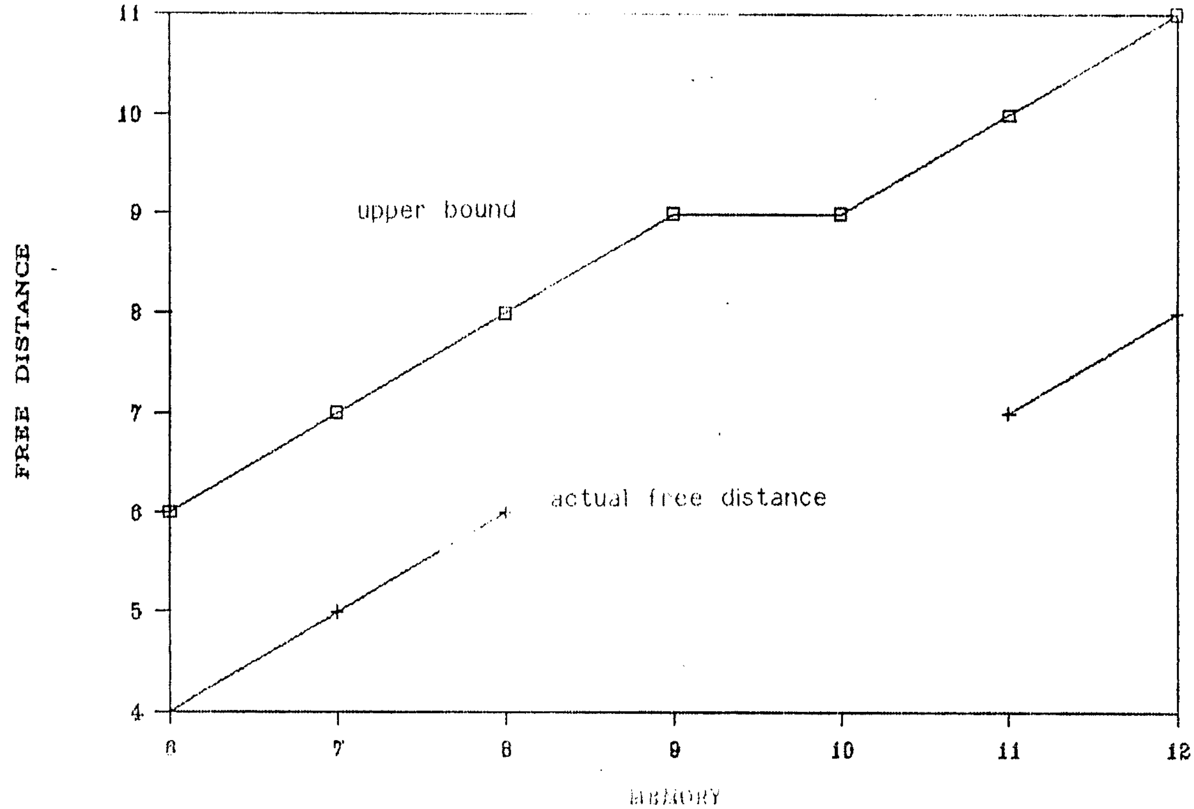

Figure 8. Free distances of some of the rate 2/3 codes found in comparison with the upper bound of (26), as a function of the memory.

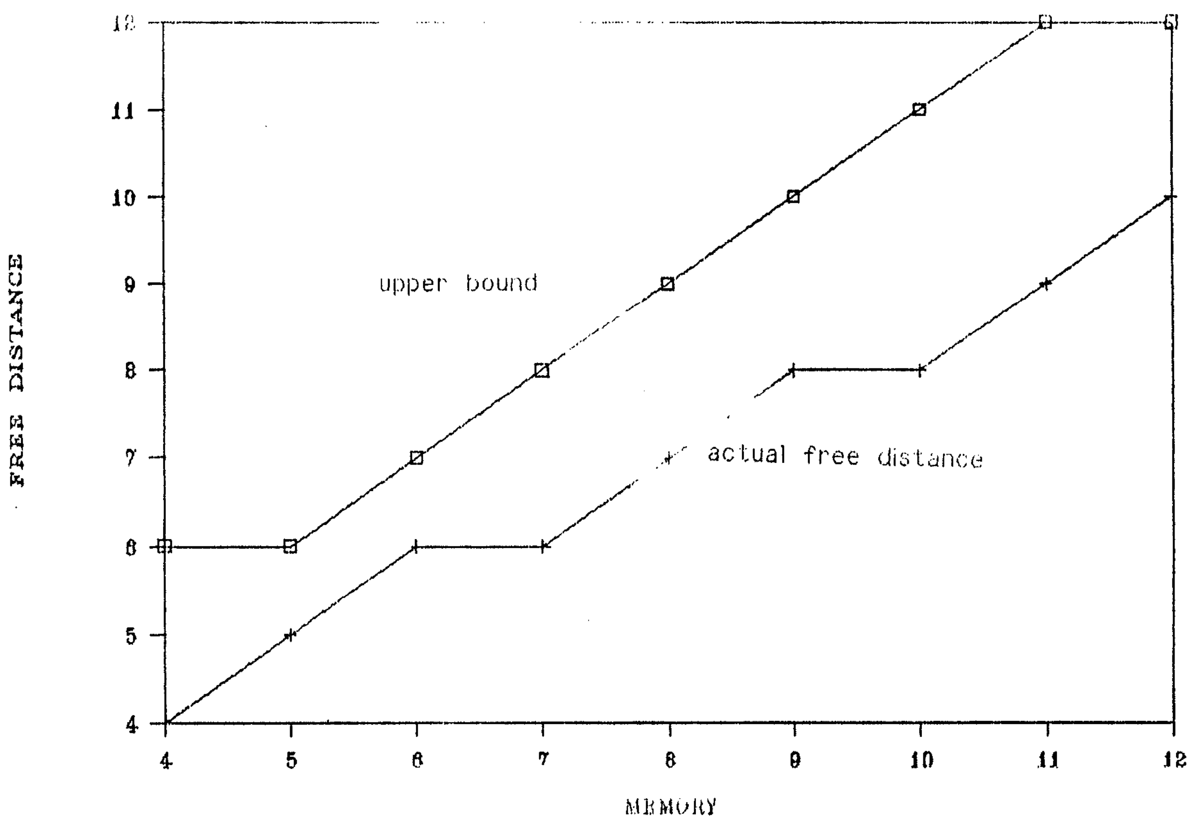

Figure 9. Free distances of some of the rate 3/4 codes found in comparison with the upper bound of (26), as a function of the memory.

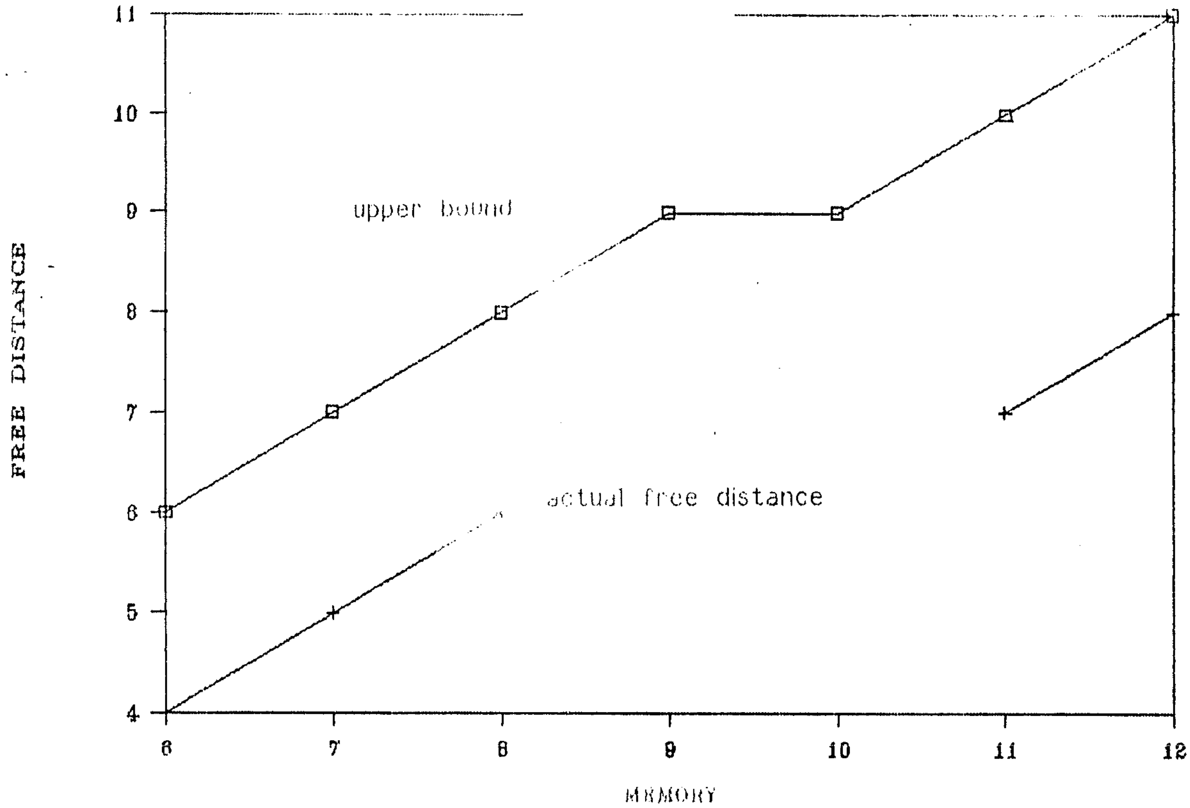

Figure 8. Free distances of some of the rate 2/3 codes found in comparison with the upper bound of (26), as a function of the memory.

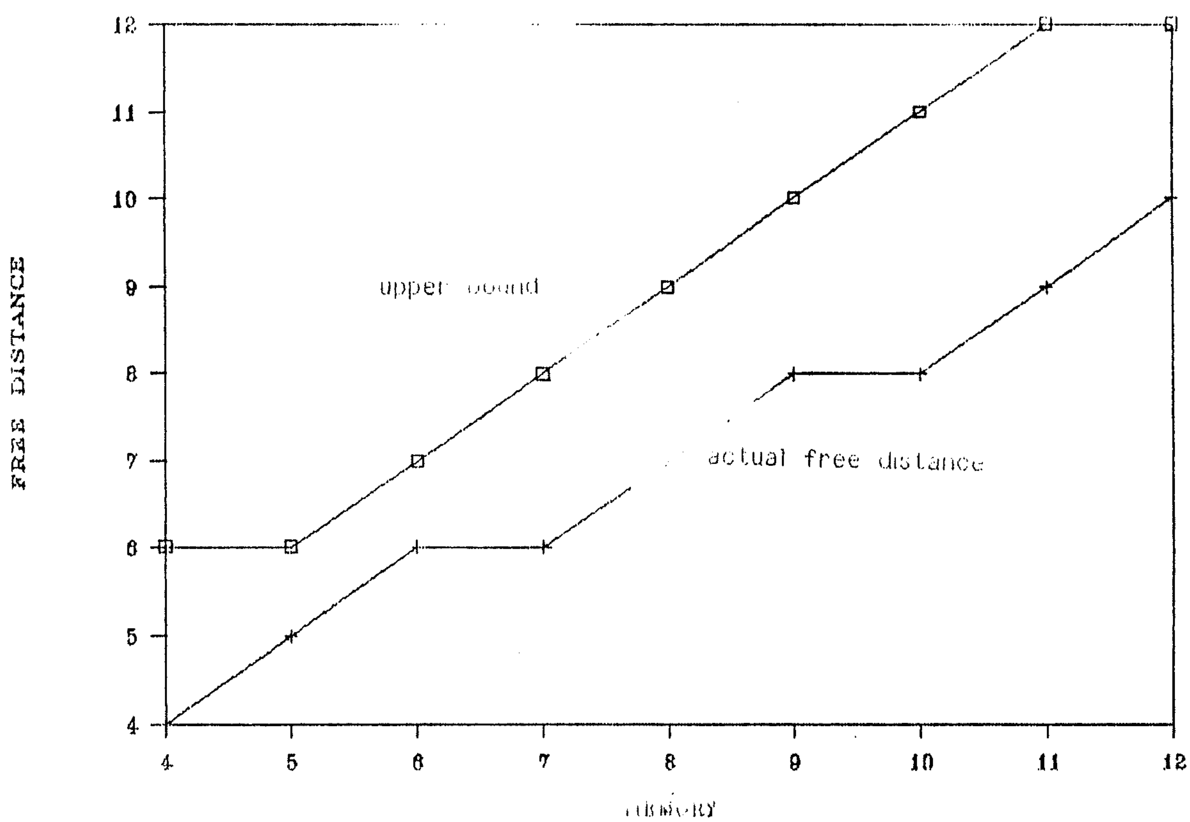

Figure 9. Free distances of some of the rate 3/4 codes found in comparison with the upper bound of (26), as a function of the memory.

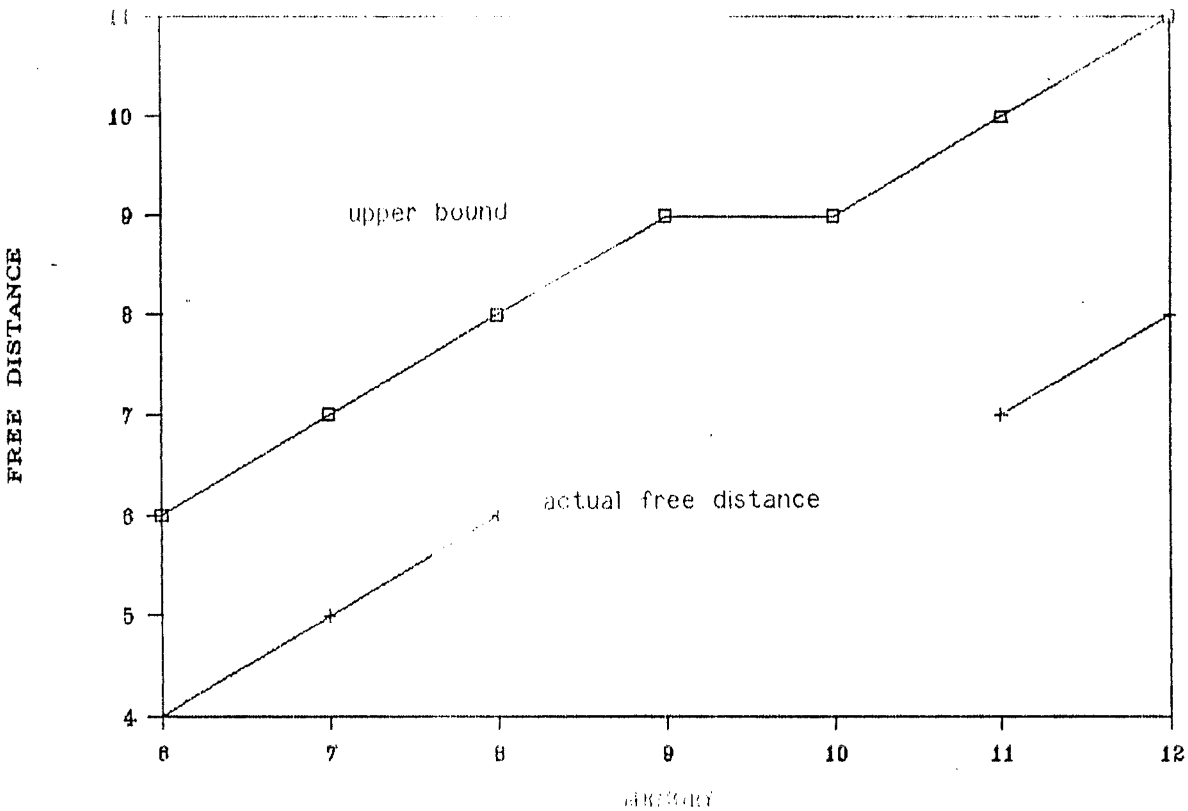

Figure 8. Free distances of some [illegible] 2/3 codes found in comparis[illegible] with the upper bound of (26), as a function of the memory.

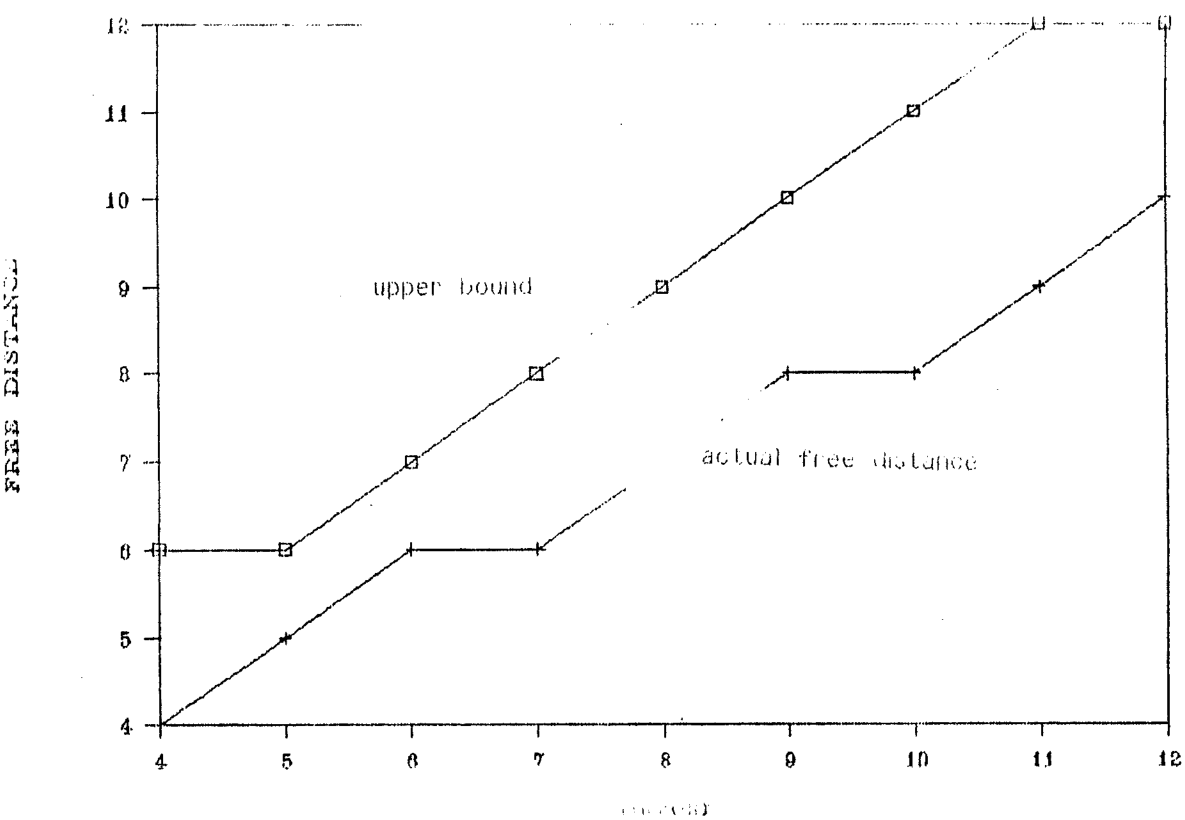

Figure 9. Free distances of som[illegible] 3/4 codes found in compari[illegible] with the upper bound of [illegible], as a function of the memory.