Estimations of the Transfer Functions of Noncatastrophic Convolutional Encoders *

V.B.Balakirsky

Abstract. A computational method, which allows us to upper-bound the solution of a wide class of the systems of linear recurrent equations, is proposed. This method is used to estimate the transfer functions of noncatastrophic convolutional encoders.

Key words :
Convolutional codes, Viterbi decoding, Recurrent equations.

*The work was supported by a Scholarship from the Swedish Institute, Stockholm, Sweden.

1 Introduction

The transfer functions of convolutional encoders were introduced by Viterbi [1] as infinite sums

$$
\begin{aligned}
T(D) &= \sum_{d \geq d_f} t(d) \cdot D^d, \qquad (1.1)\\
T'(D) &= \sum_{d \geq d_f} t'(d) \cdot D^d
\end{aligned}
$$

of powers of a formal variable D, where d_f is the code *free distance*. These functions accumulate information on the code weight spectrum, which is used to upper-bound the burst and bit error probability of the Viterbi decoder [1-3]. The values of these functions for a specific D can be calculated after solution of the system consisting of 2^m linear equations with 2^m variables [1], where m is the *memory* of the encoder (we restrict our attention to the codes of *rate* $R = 1/N$). Such a procedure requires $O(2^{3m})$ operations, and it is rather hard in realization even for small m.

We restrict our attention to noncatastrophic convolutional encoders. These encoders generate code sequences of infinite Hamming weight for every information sequence of infinite weight that makes impossible a catastrophic error propagation at the output of the Viterbi decoder [1,4]. For noncatastrophic encoders, the functions $T(D)$ and $T'(D)$ can be represented as ratios of polynomials of a finite degree, and there is a method, which gives an opportunity to find these polynomials with a desired accuracy [5]. The realization of this method can be simpler than a direct calculation of the values of $T(D)$ and $T'(D)$ for certain Ds by the Viterbi's method.

Only the initial parts of the transfer functions are usually used in practice, since "the contribution of the terms in (1.1) vanishes in the region where the known upper bounds give good results". One of the methods, which can be used to find the initial parts of $T(D)$ and $T'(D)$, is a modified Viterbi decoding algorithm. Then the complexity is estimated as $O(2^m)$ operations. Nevertheless, note that we have to run this decoding algorithm until the weight of the best path, leading to the all-zero state of the trellis, becomes greater than a certain value, and the number of the level, when the decoder stops, can be rather big. Different approaches to the solution of that problem were given in [6], where an algorithm which allows us to find the next coefficients $t(d)$ and $t'(d)$ after $2.5 \cdot 2^m$ additions, and in [7] where some ideas of sequential decoding were used.

In the present paper, we develop the ideas of [6,9] and approximate the tails of the transfer functions by infinite power series. Let

$$\begin{aligned} \hat{T}(D) &= \sum_{d=d_f}^{d_0-1} t(d) \cdot D^d + t(d_0) \cdot D^{d_0} \cdot \sum_{d \geq d_0} (cD)^{d-d_0}, \\ \hat{T}'(D) &= \sum_{d=d_f}^{d_0-1} t'(d) \cdot D^d + \frac{t'(d_0) \cdot D^{d_0}}{d_0} \cdot \sum_{d \geq d_0} d \cdot (c'D)^{d-d_0}, \end{aligned} \tag{1.2}$$

where d_0 is some constant. If the coefficients c and c' are chosen in such a way that

$$t(d) \leq t(d_0) \cdot c^{d-d_0} \tag{1.3}$$

and

$$t'(d) \leq t'(d_0) \cdot \frac{d}{d_0} \cdot (c')^{d-d_0} \tag{1.4}$$

for all $d > d_0$, then $T(D) \leq \hat{T}(D)$ and $T'(D) \leq \hat{T}'(D)$. Furthermore, if the value, which is substituted for D, is such that $cD < 1$ and $c'D < 1$, then the values $\hat{T}(D)$ and $\hat{T}'(D)$ can be obtained after a finite number of computations, since

$$\begin{aligned} \hat{T}(D) &= \sum_{d=d_f}^{d_0-1} t(d) \cdot D^d + t(d_0) \cdot D^{d_0} \cdot \frac{1}{1-cD}, \\ \hat{T}'(D) &= \sum_{d=d_f}^{d_0-1} t'(d) \cdot D^d + t'(d_0) \cdot D^{d_0} \cdot \left(\frac{1}{1-c'D} + \frac{1}{d_0} \cdot \frac{c'D}{(1-c'D)^2} \right). \end{aligned}$$

We have several arguments to justify the statement of the problem.

1) From a formal point of view, substitution of a finite sum for an infinite one is incorrect when we are interested in upper bounds on the decoding error probabilities.

2) If the code rate is greater than the computational cut-off rate and the encoder has rather long memory, we can get reasonable estimates of the burst and bit decoding error probabilities taking into account an initial part of the transfer function. However, if we compute these

estimates with approximations of the tails of $T(D)$ and $T'(D)$, then we obtain infinity as a result. This observation is illustrated in Section 4 for the encoder of memory 14, and we come to the conclusion that, from a formal point of view, the truncation of the code trellis is very important, since it does not allow an infinite number of terms affect the final result. In this case, we can obtain good estimates using the upper bounds [1,2] even for very noisy channels.

3) Initial parts of the transfer functions are found and tabulated for many good encoders (see, in particular, [6-8]). However, there are, so called, 'mixed' codes [10] and their number is much greater. The mixed codes are defined by M convolutional codes of rates $1/N_1, ..., 1/N_M$ and a periodic function $\varphi(t)$, $t = 1, 2, ...$, taking values in the set $\{1, ..., M\}$. At the time instant t, a current information bit enters all the encoders, but the code subblock will be taken from the output of the encoder $\varphi(t)$, while all the other subblocks will be lost (the well-known punctured codes belong to this class). It is possible to fix M convolutional encoders and generate good mixed codes of different rates using different functions φ. The approaches of the present paper can be easily extended to approximate the transfer functions of these codes.

4) Estimations of the tails of the transfer functions are important in calculations of the values of Cedervall-Johannesson-Zigangirov' upper bound on the burst error probability [3]. This bound and new bounds on the burst and bit error probability [1] essentially extend the region where transfer functions can be used.

The paper is organized as follows. In Section 2 we introduce a concept of recurrently solvable systems of recurrent equations and show how to approximate the solution of such systems. In Section 3 we give the systems of recurrent equations for the coefficients of the transfer functions and show that they are recurrently solvable iff the encoder is noncatastrophic. Then we describe an algorithm, constructing "a strategy of computations" for these systems and illustrate the approaches for the encoder (5,7). Furthermore, we estimate the transfer functions of the random convolutional encoders and use

[1] The abstaract has been published in Proceedings of ISIT'94, p.271. The text of the paper is in preparation.

these estimations while discussing the numerical results for fixed encoders in Section 4.

2 Computational Method for Solution of Recurrent Equations

Let us consider the following system of recurrent equations :

$$\begin{aligned} f_0(d) &= f_2(d), \\ f_1(d) &= f_0(d-2) + f_2(d-1), \\ f_2(d) &= f_3(d), \\ f_3(d) &= f_1(d) + f_3(d-1), \end{aligned} \tag{2.1}$$

where $d > 0$. Suppose that we are given the vectors $\mathbf{f}(d) = (f_0(d), ..., f_{K-1}(d))$ for all $d \leq 0$ and we want to construct the estimates $\hat{\mathbf{f}}(d) = (\hat{f}_0(d), ..., \hat{f}_{K-1}(d))$ such that $\hat{\mathbf{f}}(d) \geq \mathbf{f}(d)$ for all $d > 0$; $K = 4$. For any $d > 0$, the vector $\mathbf{f}(d)$ can be constructed directly from (2.1) if we know the vectors $\mathbf{f}(d-1)$ and $\mathbf{f}(d-2)$ and read the equations in the order : 1,3,2,0. Let us find the vectors $\mathbf{f}(d)$ for all $d \leq d_0$ and set

$$\hat{\mathbf{f}}(d) = \begin{cases} \mathbf{f}(d), & \text{if } d \leq d_0, \\ c^{d-d_0} \cdot \mathbf{f}(d_0), & \text{if } d > d_0. \end{cases} \tag{2.2}$$

To find the coefficient c, we substitute $\hat{f}_k$ for f_k and "$\geq$" for "$=$" into (2.1) , i.e., we consider the following system of recurrent inequalities :

$$\begin{aligned} \hat{f}_0(d) &\geq \hat{f}_2(d), \\ \hat{f}_1(d) &\geq \hat{f}_0(d-2) + \hat{f}_2(d-1), \\ \hat{f}_2(d) &\geq \hat{f}_3(d), \\ \hat{f}_3(d) &\geq \hat{f}_1(d) + \hat{f}_3(d-1). \end{aligned}$$

It is easy to see that these inequalities are valid for all $d > 0$ if

$$\begin{aligned} c \cdot f_1(d_0) &\geq f_0(d_0 - 1) + f_2(d_0), \\ c^2 \cdot f_1(d_0) &\geq f_0(d_0) + c \cdot f_2(d_0), \\ c \cdot f_3(d_0) &\geq c \cdot f_1(d_0) + f_3(d_0), \end{aligned}$$

and the minimal value of c satisfying these inequalities, can be found. The justification of the formal steps, used in this example, is given below in general case.

Let

$$\begin{aligned} f_k(d) &= \sum_{j\in J_k} f_j(d-w_{jk}) + \Delta_k(d), \ \text{ for all } \ d>0, \\ & \quad k=0,...,K-1, \end{aligned} \tag{2.3}$$

be a system of recurrent equations, where $\Delta_k(d) \geq 0$ are given constants, $J_k \subseteq \{0,1,...,K-1\}$, and w_{jk} are non-negative integers; $j,k=0,...,K-1$. Let the boundary conditions of (2.3) be given by the vectors $\mathbf{f}(d), d \leq 0$. We may also say that (2.3) defines a direct graph with K vertices enumerated by integers $0,...,K-1$. The vertex j is connected with the vertex k by a branch of weight w_{jk} iff $j \in J_k$. The weight of a path on the graph is defined as the sum of the weights of its branches.

Definition. The system of recurrent equations (2.3) will be referred to as *recurrently solvable* if there exists a vector $\mathbf{s} = (s_0,...,s_{K-1})$ with the following properties :

1) $\{s_0,...,s_{K-1}\} = \{0,...,K-1\}$;
2) to calculate $f_{s_k}(d)$ from (2.3) it is sufficient to know the vectors $\mathbf{f}(d')$, $d' < d$ and coefficients $f_{s_l}(d), l<k$, where $k=0,...,K-1$.

Any vector $\mathbf{s}$, satisfying conditions above, will be referred to as a *strategy of computations.*

Proposition 2.1. The system (2.3) is recurrently solvable iff the corresponding graph does not contain cycles of the zero weight.

Proof. Let τ_k be the length of the longest path having the zero weight and leading to the vertex k. If all the branches, entering k, have nonzero weights, we set $\tau_k = 0$, and if k belongs to a cycle with the zero weight, we set $\tau_k = \infty$. If the graph does not contain cycles of the zero weight, i.e., $\tau_k < \infty$ for all $k = 0,...,K-1$, then a strategy of computations can be constructed after several sequential examinations of the graph : first we write down all the vertices k such that $\tau_k = 0$, then all the vertices k such that $\tau_k = 1$, and so on until the current dimension of $\mathbf{s}$ is less than K. The converse statement is also evident. If the graph contains cycles of the zero weight, then the procedure above leads to a vector $\mathbf{s}$ whose current dimension will not be increased after the current examination. Any vertex, which is not included into $\mathbf{s}$, belongs to one of these cycles. Q.E.D.

Proposition 2.2. Let $d_0 > 0$ be a constant and let the system (2.3) be recurrently solvable. If

$$\hat{\mathbf{f}}(d) = \mathbf{f}(d) \text{ for all } d \leq d_0 \tag{2.4}$$

and

$$\hat{f}_k(d) \quad \geq \quad \sum_{j \in J_k} \hat{f}_j(d - w_{jk}) + \Delta_k(d) \text{ for all } d > d_0, \tag{2.5}$$

where $k = 0, ..., K-1$, then

$$\hat{\mathbf{f}}(d) \geq \mathbf{f}(d) \text{ for all } d > 0. \tag{2.6}$$

Proof. Let us denote $\delta_k(d) = \hat{f}_k(d) - f_k(d)$. Then, combining (2.3)-(2.5), we conclude that

$$\delta_k(d) \quad \geq \quad \sum_{j \in J_k} \delta_j(d - w_{jk}) \text{ for all } d > 0, \tag{2.7}$$

where $k = 0, ..., K-1$. Suppose that the inequalities $\delta_k(d) \geq 0$ are valid up to $d < d^*$ for some $d^* > 0$; $k = 0, ..., K-1$. Since the system (2.3) is recurrently solvable, there exists a strategy of computations $\mathbf{s} = (s_0, ..., s_{K-1})$. It means that $\delta_{s_0}(d^*)$ can be lower-bouded by a linear combination of the values $\delta_k(d)$, $d < d^*$ $k = 0, ..., K-1$ (see (2.7)), which are non-negative by our assumption. Hence, $\delta_{s_0}(d^*) \geq 0$. The value of $\delta_{s_1}(d^*)$ can be also lower-bouded by a linear combination of these values and the value of $\delta_{s_0}(d^*)$. Hence, $\delta_{s_1}(d^*) \geq 0$. We repeat these considerations for $\delta_{s_2}(d^*), ..., \delta_{s_{K-1}}(d^*)$ and conclude they are non-negative. Because of the property 1 of the vector $\mathbf{s}$, it means that $\delta_k(d^*) \geq 0$ for all $k = 0, ..., K-1$, and (2.6) has been proved by induction. Q.E.D.

3 Approximation of the transfer functions

3.1 Notations

As well-known, convolutional encoder may be represented as an automata with $e = 2^m$ states, where the *state* at time unit t is defined by the content of the shift register of the encoder (Fig.1) :

$$n_t = \sum_{s=0}^{m-1} n_t^{(s)} \cdot 2^s.$$

Then the current state n_t and the next bit i_t define both the next state :

$$n_{t+1} = (2 \cdot n_t + i_t), \text{ mod } e$$

and the code subblock $(v_t^{(1)}, ..., v_t^{(N)})$,

$$v_t^{(j)} = g_0^{(j)} i_t + g_1^{(j)} n_t^{(0)} + ... + g_m^{(j)} n_t^{(m-1)}, \quad j = 1, ..., N.$$

Coefficients $g_0^{(j)}, ..., g_m^{(j)}$ are defined by the *generator polynomials*

$$G^{(j)}(x) = g_0^{(j)} + g_1^{(j)} x + ... + g_m^{(j)} x^m, \quad j = 1, ..., N,$$

of a formal variable x, where

$$\begin{aligned} g_0^{(j)} &= 1 \quad \text{for some } \; j \in \{1, ..., N\}; \\ g_m^{(j)} &= 1 \quad \text{for some } \; j \in \{1, ..., N\}. \end{aligned} \tag{3.1}$$

To simplify formalization, we enumerate the code subblocks by integers $0, ...,$ $2e-1$ and say that the $(2n+i)$-th code subblock will be generated when $n \in$ $\in \{0, ..., e-1\}$ is the current state and bit i enters the encoder. We denote the Hamming weight of that subblock by w_{2n+i}.

All possible transitions of the encoder can be shown on the directed graph, which is known as the *state diagram* of the encoder [1]. If the self-loop of the vertex 0 has been deleted from the state diagram, and this vertex is splitted into two vertices, which are the source and receiver of all paths on the graph, then the resulting graph is regarded as the *modified state diagram* (MSD) of the encoder. An expression $D^d \cdot I^i$ is assigned to every branch, and it is referred to as the *transfer function* of that branch. The parameters d

and i are the Hamming weight of the code subblock, assigned to the branch, and information bit corresponding to it, respectively (Fig.2). The transfer function of a path on the MSD is defined as a product of transfer functions of its branches. The power of the variable D of the transfer function of some path is regarded as the *weight* of that path. All the cycles on the MSD of any noncatastrophic encoder have non-zero weights [1,4].

3.2 Recurrent equations for coefficients of the transfer functions

Let

$$T_n(D,I) = \sum_{d,i} t_n(d,i) \cdot D^d \cdot I^i$$

be the sum of the transfer functions of the paths on the MSD, leading to the vertex n, and let

$$\begin{aligned} T_n(D) &= \sum_{d} t_n(d) \cdot D^d, \\ T'_n(D) &= \sum_{d} t'_n(d) \cdot D^d, \end{aligned}$$

where

$$\begin{aligned} t_n(d) &= \sum_{i} t_n(d,i), \\ t'_n(d) &= \sum_{i} i \cdot t_n(d,i). \end{aligned}$$

Then

$$T(D) = T_0(D), \;\; T'(D) = T'_0(D).$$

It is easy to see that the transfer functions $T_n(D,I)$ are connected one with another by the following equations :

$$\begin{aligned} T_0(D,I) &= D^{w_e} \cdot T_{e/2}(D,I), \\ T_1(D,I) &= D^{w_1} \cdot I + D^{w_{1+e}} \cdot I \cdot T_{e/2}(D,I), \\ T_{2n+i}(D,I) &= D^{w_{2n+i}} \cdot I^i \cdot T_n(D,I) + D^{w_{2n+i+e}} \cdot I^i \cdot T_{n+e/2}(D,I), \\ & \quad \text{for all } \; n = 1, ..., e/2-1, \;\; i = 0,1. \end{aligned} \tag{3.2}$$

Really, the state $2n+i$ can be reached both from the state n, when 0 is stored in the m-th cell of the shift register of the encoder, and from the state $n+e/2$, when this bit is 1 (see Fig.3). The states 0 and 1 should be examined separately because of the transformation of the state diagram of the encoder to the MSD.

Using (3.2) we obtain the following equations for the coefficients of the transfer functions :

$$
\begin{aligned}
t_0(d) &= t_{e/2}(d-w_e), \\
t_1(d) &= \chi\{\ d=w_1\ \}+t_{e/2}(d-w_{1+e}), \\
t_{2n+i}(d) &= t_n(d-w_{2n+i})+t_{n+e/2}(d-w_{2n+i+e}), \\
& \text{for all}\ \ n=1,...,e/2-1,\ \ i=0,1
\end{aligned}
\tag{3.3}
$$

and

$$
\begin{aligned}
t'_0(d) &= t'_{e/2}(d-w_e), \\
t'_1(d) &= t'_{e/2}(d-w_{1+e})+t_1(d), \\
t'_{2n+i}(d) &= t'_n(d-w_{2n+i})+t'_{n+e/2}(d-w_{2n+i+e})+ \\
&+ t_{2n+i}(d)\cdot\chi\{\ i=1\ \}, \\
& \text{for all}\ \ n=1,...,e/2-1,\ \ i=0,1
\end{aligned}
\tag{3.4}
$$

for all $d>0$, where χ denotes the indicator function ($\chi\{\beta\}=1$ if the statement β is true, and $\chi\{\beta\}=0$ otherwise).

Proposition 3.1. For any noncatastrophic convolutional encoder, the system of linear recurrent equations (3.3) is recurrently solvable. Given coefficients $t_n(d)$, $n=0,...,e-1$, $d=1,2,...$, the system (3.4) is also recurrently solvable.

Proof. The first part directly follows from the definition of a noncatastrophic convolutional encoder and Proposition 2.1. These coefficients are constants for the equations of the system (3.4), which has the same structure as (3.3). Q.E.D.

In Section 2, the same multiplier c^{d-d_0} was used to approximate $f_0(d),...,$ $f_{K-1}(d)$. We will also use this approach to approximate the coefficients of the transfer functions $t_0(d),...,t_{e-1}(d)$ and $t'_0(d),...,t'_{e-1}(d)$. We denote

$$
\begin{aligned}
\mathbf{t}(d) &= (t_0(d),...,t_{e-1}(d)), \\
\mathbf{t}'(d) &= (t'_0(d),...,t'_{e-1}(d))
\end{aligned}
$$

and set more strong restrictions on c and c' compared to (1.3) and (1.4), i.e.,

$$\hat{\mathbf{t}}(d) = \begin{cases} \mathbf{t}(d), & \text{if } d \leq d_0, \\ c^{d-d_0} \cdot \mathbf{t}(d_0), & \text{if } d > d_0 \end{cases} \tag{3.5}$$

and

$$\hat{\mathbf{t}}'(d) = \begin{cases} \mathbf{t}'(d), & \text{if } d \leq d_0, \\ c^{d-d_0} \cdot (d/d_0) \cdot \mathbf{t}'(d_0), & \text{if } d > d_0. \end{cases} \tag{3.6}$$

In Subsection 3.4 we show how to use these inequalities to find c and c' for the encoder (5,7). The same approach is used in general case.

3.3 Constructing of a strategy of computations

A strategy of computations for the systems (3.3), (3.4) can be constructed for any noncatastrophic encoder using the algorithm described in Section 2: we define τ_n as the length of the longest path of the zero weight leading to vertex n and write down a sequence of the vertices $\{n\}$ that corresponds to a non-decreasing sequence $\{\tau_n\}$. This algorithm requires to calculate $\tau_n, n = 0, ..., e-1$, after several examinations of the list of the vertices. The algorithm below examines this list only once.

We note that any encoder of rate $1/N$, whose generator polynomials satisfy (3.1), has a state diagram such that at least one branch of a non-zero weight leaves each vertex and at least one branch of a non-zero weight enters each vertex. It gives an opportunity to represent the strategy of computations as a sequence of *chains* on the MSD, where the chain is defined as a path of the zero weight such that both branches, entering the first vertex of the chain, and both branches, leaving the last vertex of the chain, have non-zero weights (Fig.4).

In the algorithm below, we select the last vertex of some chain. Then we find the first vertex of that chain and put all the intermidiate vertices into the array *List*. When the first vertex has been found, we store *List* in $\mathbf{s}$.

1. Set $s_0 = 0,\ V = 1,\ n = 1$.

2. If $w_{2n} = 0$ or $w_{2n+1} = 0$, then go to 7.

3. Set $j = 0, List_j = n$.

4. Set $l = List_j$. If $w_l > 0$ and $w_{l+e} > 0$, then go to 6.
5. Set $j = j+1$. If $w_l = 0$, then set $List_j = \lfloor l/2 \rfloor$. Otherwise, set $List_j = \lfloor l/2 \rfloor + e/2$ ($\lfloor l/2 \rfloor$ denotes the maximal integer which is not greater than $l/2$). Go to 4.
6. Set $s_{V+i} = List_{j-i}$ for all $i = 0, ..., j$. Set $V = V + j + 1$.
7. Set $n = n + 1$. If $n < e$, then go to 2.
8. End. If $V < e - 1$, then the encoder is catastrophic, and any node n, which is not included into $\mathbf{s}$, belongs to a cycle of the zero weight. If $V = e - 1$, then $\mathbf{s}$ is a strategy of computations.

3.4 Approximation of the transfer functions for the encoder (5,7)

The encoder (5,7) has memory $m = 2$, rate $R = 1/2$, and generator polynomials :

$$G^{(1)}(x) = 1 + x^2, \quad G^{(2)}(x) = 1 + x + x^2.$$

A scheme of the encoder and its MSD are shown in Fig.1,2. Since

$$(w_0, ..., w_7) = (0, 2, 1, 1, 2, 0, 1, 1),$$

equations (3.3), (3.4) are of the following form :

$$\begin{aligned} t_0(d) &= t_2(d-2), \\ t_1(d) &= \chi\{\ d = 2\ \} + t_2(d), \\ t_2(d) &= t_1(d-1) + t_3(d-1), \\ t_3(d) &= t_1(d-1) + t_3(d-1), \end{aligned} \tag{3.7}$$

and

$$\begin{aligned} t'_0(d) &= t'_2(d-2), \\ t'_1(d) &= t'_2(d) + t_1(d), \\ t'_2(d) &= t'_1(d-1) + t'_3(d-1), \\ t'_3(d) &= t'_1(d-1) + t'_3(d-1) + t_3(d). \end{aligned} \tag{3.8}$$

The encoder is noncatastrophic, and a strategy of computations is any permutation of the components $(0, 1, 2, 3)$ such that 2 preceeds 1. In particular, the algorithm of Section 3.3 constructs the vector $\mathbf{s} = (0, 2, 1, 3)$. It means that, for all $d > 0$, we use the following order of computations : $t_0(d), t_2(d), t_1(d), t_3(d)$. After that we find $t'_0(d), t'_2(d), t'_1(d), t'_3(d)$, since $t_1(d)$ and $t_3(d)$ enter (3.8) as constants.

We repeat the considerations of Section 2 and choose the coefficient c in such a way that

$$\begin{aligned} c \cdot t_0(d_0) &\geq t_2(d_0 - 1), \\ c^2 \cdot t_0(d_0) &\geq t_2(d_0), \\ c \cdot t_2(d_0) &\geq t_1(d_0) + t_3(d_0), \\ c \cdot t_3(d_0) &\geq t_1(d_0) + t_3(d_0). \end{aligned} \tag{3.9}$$

Thus,

$$c = \max\left\{ \frac{t_2(d_0 - 1)}{t_0(d_0)}, \sqrt{\frac{t_2(d_0)}{t_0(d_0)}}, \frac{t_1(d_0) + t_3(d_0)}{t_2(d_0)}, \frac{t_1(d_0) + t_3(d_0)}{t_3(d_0)} \right\}.$$

Let us construct a coefficient c', satisfying (3.6). The first and the third equations in (3.8) give the inequalities :

$$\begin{aligned} c' \cdot t'_0(d_0) &\geq t'_2(d_0 - 1), \\ (c')^2 \cdot t'_0(d_0) &\geq t'_2(d_0), \\ c' \cdot t'_2(d_0) &\geq t'_1(d_0) + t'_3(d_0), \end{aligned} \tag{3.10}$$

while the second and the fourth equations lead to the inequalities :

$$\begin{aligned} (c')^{d-d_0} \cdot t'_1(d_0) &\geq (c')^{d-d_0} \cdot t'_2(d_0) + t_1(d), \\ (c')^{d-d_0} \cdot t'_3(d_0) &\geq (c')^{d-d_0-1} \cdot t'_1(d_0) + (c')^{d-d_0-1} \cdot t'_3(d_0) + t_3(d) \end{aligned}$$

for all $d > d_0$. Let

$$c' > c. \tag{3.11}$$

Then

$$t_n(d) \leq c^{d-d_0} \cdot t_n(d_0) \leq (c')^{d-d_0} \cdot t_n(d_0)$$

for all $d > d_0$, where $n = 1, 3$. Therefore, we get more strong restrictions on c' if we write :

$$\begin{aligned}(c')^{d-d_0} \cdot t'_1(d_0) &\geq (c')^{d-d_0} \cdot t'_2(d_0) + (c')^{d-d_0} \cdot t_1(d_0),\\ (c')^{d-d_0} \cdot t'_3(d_0) &\geq (c')^{d-d_0-1} \cdot t'_1(d_0) + (c')^{d-d_0-1} \cdot t'_3(d_0) + (c')^{d-d_0} \cdot t_3(d_0).\end{aligned}$$

The first inequality is valid for any c', and the second one is of the following form :

$$c' \cdot t'_3(d_0) \geq t'_1(d_0) + t'_3(d_0) + c' \cdot t_3(d_0). \tag{3.12}$$

Combining (3.10), (3.11), and (3.12), we get

$$c' = \max\left\{ c,\ \frac{t'_2(d_0-1)}{t'_0(d_0)},\ \sqrt{\frac{t'_2(d_0)}{t'_0(d_0)}}, \frac{t'_1(d_0)+t'_3(d_0)}{t'_2(d_0)},\ \frac{t'_1(d_0)+t'_3(d_0)}{t'_3(d_0)-t_3(d_0)} \right\}.$$

For the encoder (5,7), we know that

$$\begin{aligned}\mathbf{t}(d) &= (\quad 2^{d-5},\quad 2^{d-3},\quad 2^{d-3},\quad 2^{d-3}\ ),\\ \mathbf{t}'(d) &= (\ (d-4)\cdot 2^{d-5},\ (d-1)\cdot 2^{d-3},\ (d-2)\cdot 2^{d-3},\ (d-1)\cdot 2^{d-3}\ )\end{aligned}$$

for all $d \geq 5$. Therefore, as it is easy to see, we obtain the following coefficients:

$$c = 2,\ \ c' = 2 \cdot \frac{d_0 - 3}{d_0 - 4}.$$

3.5 Approximation of the transfer functions of random convolutional encoders

We will compare the numerical results that are obtained for fixed convolutional encoders with the results for random encoders, since they may be expressed in a short form and the structure of a time-invariant code tends to the structure of a random code when m grows. We do not claim that the result, given below as Proposition 3.2, is new, but we cannot give the exact reference. A short proof, which is based on the same technique as in Section 2, is given in Appendix.

Proposition 3.2. Let $\bar{t}(d)$ and $\bar{t}'(d)$ be coefficients of the transfer functions of a random convolutional encoder when each coefficient of the generator polynomials is equal to 0 or 1 with probability 1/2 and such an assignement is realized independently at every level of the code trellis. Then

$$\begin{aligned} \overline{t}(d) &\leq 2^{-m} \cdot (\overline{c})^d, \\ \overline{t}'(d) &\leq 2^{-m} \cdot (d+1) \cdot (\overline{c})^d \text{ for all } d \geq 0, \end{aligned} \tag{3.13}$$

where

$$\overline{c} = \frac{1}{2^{1-R} - 1}. \tag{3.14}$$

Corollary. Let us suppose that the data are transmitted over a binary-input L-ary output discrete memoryless channel such that the conditional probability to receive $y \in \{0, ..., L-1\}$ is equal to $W(y)$ when 0 was sent and it is equal to $W(L-1-y)$ when 1 was sent. Let

$$R_{comp} = 1 - \log_2 \left(1 + \sum_{y=0}^{L-1} \sqrt{W(y) \cdot W(L-1-y)} \right) \tag{3.15}$$

denote the computational cut-off rate of that channel. Then the inequality $\overline{c} D_W < 1$, where

$$D_W = \sum_{y=0}^{L-1} \sqrt{W(y) \cdot W(L-1-y)},$$

is equivalent to the inequality $R < R_{comp}$.

4 Numerical results

We can find the vectors $\mathbf{t}(d)$ and $\mathbf{t}'(d)$ for all $d > 0$, based on a strategy of computations and the vectors $\mathbf{t}(d-1), ..., \mathbf{t}(d-N)$ and $\mathbf{t}'(d-1), ..., \mathbf{t}'(d-N)$. Therefore, the total size of the memory, needed for our algorithm, is not greater than $2 \cdot (N+1) \cdot 2^m$. Each equation for $t_n(d)$ and $t'_n(d)$ generates not more than N inequalities for c and c'. Therefore, taking into account the fact that the equation for $t'_n(d)$ contains two summands when n is even, and three summands when n is odd, we conclude that not more than $2.5 \cdot 2^m$ and $2.5 \cdot N \cdot 2^m$ additions are required to find $(\mathbf{t}(d), \mathbf{t}'(d))$ and (c, c'), respectively. The constants before 2^m can be reduced for particular classes of convolutional encoders (for example, when $R = 1/2$ and $g_0^{(1)} = g_0^{(2)} = g_m^{(1)} = g_m^{(2)} = 1$).

In Table 1 we show the coefficients c and c', which estimate the tails of the transfer functions $T(D)$ and $T'(D)$ starting from the k-th term of infinite power series (1.1). The generator polynomials of the encoders are given by the vectors $\mathbf{g}^{(j)} = (g_0^{(j)}, ..., g_m^{(j)})$ written in the octal notation, $j = 1, 2$. The first 10 terms $t(d)$ and $t'(d), d = d_f, ..., d_f + 9$, for these encoders are given in [7]. The results show that c and c' very slightly depend on m and generator polynomials and tend to the coefficient $\overline{c} = \sqrt{2} + 1$, whose powers estimate the coefficients of the transfer functions for random encoders of rate 1/2. This fact is not a suprise, because we expect that the difference between the number of codewords of a certain Hamming weight, d, vanishes with d for a fixed and a random encoder, when m is large enough. The estimates of the transfer functions allow us to formalize this observation. Furthermore, such a note gives the reasons to expect that our estimations are rather tight, and more complicated forms of approximation functions, compared to (1.3) and (1.4), cannot essentially improve them.

The lines, corresponding to $m = 6$ and $m = 10$, are absent in Table 1 since the data in [7] contain transparent codes having these memories, i.e., $t(d) = t'(d) = 0$ for all odd values of d, and the estimations (1.3) and (1.4) can be constructed with $c, c' < \infty$ only when d_0 is even. However, we use more strong estimations (3.5) and (3.6), and there are vertices on the MSD of these encoders such that $t(d) = t'(d) = 0$ for all even values of d. Therefore, in this case we have to change the form of approximation functions. Note that the replacement of the coefficients c and c' with $\tilde{c}^2$ and $(\tilde{c}')^2$ leads to a different system of inequalities for these coefficients compared to the previous one.

Estimations of the transfer functions are important when the code rate is close to the computational cut-off rate, defined in (3.15). Table 2 contains the values of the 'Van de Meeberg'-type upper bound [3] on the burst and bit error probabilities, P_B and P_b, for the encoder of memory 14 and binary symmetric channel (BSC) with crossover probability p. The value $D_p = 2\sqrt{p(1-p)}$ is substituted for D when we calculate the value of the transfer functions, given p. The results tend to infinity when p is close to 0.045 (the point when $R_{comp} \approx 1/2$.) However, we get rather good results if we take into account only a finite number of terms in (1.1). Truncation of the code trellis [11], which does not allow infinite number of terms of the transfer functions to affect the final result, seems to be very important (note that in this case we also have to estimate the truncation decoding error probability).

5 Conclusion

The algorithms, given in the paper, use $O(2^m)$ additions to construct both the coefficients of the transfer functions and the coefficients c and c', which are used to approximate the tails of these functions. It is known that the first few terms $t(d_f), ..., t(d_f + k - 1)$ and $t'(d_f), , ..., t'(d_f + k - 1)$ $d = d_f, ..., d_f + k - 1$, can be found much easier using the ideas of sequential decoding [7]. However, the complexity of the algorithm grows with k and becomes an exponential function of m when k is large enough. On the other hand, when k grows, the coefficients c and c' approach the value obtained for the random encoders. Therefore, we assume that the number of operations of any procedure, which solves both problems should be of order 2^m.

6 Acknowledgement

The author is very grateful to Prof. Mats Cedervall whose support, encouragement, and suggestions essentially stimulated this work. The comments and suggestions of Professor Rolf Johannesson and Prof. Kamil Sh. Zigangirov are also of great value. In particular, the term 'recurrently solvable systems of recurrent equations' belongs to Prof. Johannesson. The author is also grateful to the anonymous reviewer for a copy of the paper [5] and helpful comments.

Table 1: Coefficients c and c' for codes of rate 1/2 having the maximal free distance for different k. The first line contains coefficients c, and the second line contains coefficients c'. The value obtained for random encoders : $\overline{c} = 2.414214...$

m	$\mathbf{g}^{(1)}$	$\mathbf{g}^{(2)}$	d_f	$k=5$	$k=10$	$k=25$	$k=50$
2	5	7	5	2.000000	2.000000	2.000000	2.000000
				2.400000	2.200000	2.080000	2.040000
3	13	17	6	2.208333	2.205578	2.205570	2.205570
				2.561538	2.402611	2.289774	2.248653
4	27	31	7	2.916667	2.390074	2.312989	2.312660
				2.916667	2.577593	2.398268	2.357178
5	55	75	8	2.530612	2.353647	2.346413	2.346413
				2.754967	2.532119	2.430419	2.390743
7	275	313	10	3.200000	2.498301	2.400263	2.399922
				3.375000	2.671544	2.486266	2.445348
8	557	751	12	3.058824	2.547591	2.407920	2.407220
				3.411765	2.728692	2.490125	2.451674
9	1363	1755	12	3.530612	2.498193	2.410744	2.410675
				3.722222	2.638945	2.495666	2.455861
11	5175	6643	14	3.270270	2.485447	2.413276	2.413164
				3.644501	2.628035	2.494990	2.457444
12	11651	14677	15	3.235294	2.499170	2.413788	2.413679
				3.405275	2.632605	2.497584	2.458569
13	22555	37457	16	3.024000	2.492311	2.414032	2.413938
				3.291032	2.635347	2.494996	2.458007
14	43677	65231	17	3.710526	2.480563	2.414164	2.414072
				3.710526	2.634856	2.494639	2.457990

Table 2: Estimates of P_B and P_b for the encoder (43677, 65231) of memory $m = 14$. The second line contains estimates obtained using first k terms of $T(D)$ and $T'(D)$, and the first line contains estimates with approximation of tails of these functions.

		P_B			
p	$1/D_p$	$k = 5$	$k = 10$	$k = 25$	$k = 50$
0.050	2.294	∞	∞	∞	∞
		$1.74 \cdot 10^{-5}$	$4.34 \cdot 10^{-5}$	$1.76 \cdot 10^{-4}$	$8.12 \cdot 10^{-4}$
0.045	2.412	∞	∞	∞	∞
		$6.83 \cdot 10^{-6}$	$1.51 \cdot 10^{-5}$	$3.98 \cdot 10^{-5}$	$8.18 \cdot 10^{-5}$
0.040	2.552	∞	$1.85 \cdot 10^{-5}$	$1.17 \cdot 10^{-5}$	$1.17 \cdot 10^{-5}$
		$2.40 \cdot 10^{-6}$	$4.66 \cdot 10^{-6}$	$8.64 \cdot 10^{-6}$	$1.09 \cdot 10^{-5}$
0.035	2.721	∞	$2.07 \cdot 10^{-6}$	$1.89 \cdot 10^{-6}$	$1.89 \cdot 10^{-6}$
		$7.30 \cdot 10^{-7}$	$1.25 \cdot 10^{-6}$	$1.78 \cdot 10^{-6}$	$1.88 \cdot 10^{-6}$
0.030	2.931	∞	$3.49 \cdot 10^{-7}$	$3.40 \cdot 10^{-7}$	$3.40 \cdot 10^{-7}$
		$1.84 \cdot 10^{-7}$	$2.81 \cdot 10^{-7}$	$3.37 \cdot 10^{-7}$	$3.40 \cdot 10^{-7}$
0.025	3.203	∞	$5.39 \cdot 10^{-8}$	$5.35 \cdot 10^{-8}$	$5.35 \cdot 10^{-8}$
		$3.61 \cdot 10^{-8}$	$4.93 \cdot 10^{-8}$	$5.34 \cdot 10^{-8}$	$5.35 \cdot 10^{-8}$
0.020	3.571	∞	$6.26 \cdot 10^{-9}$	$6.25 \cdot 10^{-9}$	$6.25 \cdot 10^{-9}$
		$4.91 \cdot 10^{-9}$	$6.06 \cdot 10^{-9}$	$6.25 \cdot 10^{-9}$	$6.25 \cdot 10^{-9}$
	$c =$	3.710526	2.480563	2.414164	2.414072

		P_b			
p	$1/D_p$	$k = 5$	$k = 10$	$k = 25$	$k = 50$
0.050	2.294	∞	∞	∞	∞
		$1.06 \cdot 10^{-4}$	$3.74 \cdot 10^{-4}$	$2.96 \cdot 10^{-3}$	$2.72 \cdot 10^{-2}$
0.045	2.412	∞	∞	∞	∞
		$4.15 \cdot 10^{-5}$	$1.25 \cdot 10^{-4}$	$5.90 \cdot 10^{-4}$	$2.09 \cdot 10^{-3}$
0.040	2.552	∞	∞	$8.61 \cdot 10^{-3}$	$1.63 \cdot 10^{-3}$
		$1.45 \cdot 10^{-5}$	$3.73 \cdot 10^{-5}$	$1.08 \cdot 10^{-4}$	$1.85 \cdot 10^{-4}$
0.035	2.721	∞	$9.40 \cdot 10^{-4}$	$6.27 \cdot 10^{-5}$	$2.47 \cdot 10^{-5}$
		$4.36 \cdot 10^{-6}$	$9.59 \cdot 10^{-6}$	$1.86 \cdot 10^{-5}$	$2.17 \cdot 10^{-5}$
0.030	2.931	∞	$1.51 \cdot 10^{-5}$	$3.58 \cdot 10^{-6}$	$3.03 \cdot 10^{-6}$
		$1.09 \cdot 10^{-6}$	$2.05 \cdot 10^{-6}$	$2.93 \cdot 10^{-6}$	$3.03 \cdot 10^{-6}$
0.025	3.203	∞	$7.64 \cdot 10^{-7}$	$4.10 \cdot 10^{-7}$	$4.04 \cdot 10^{-7}$
		$2.12 \cdot 10^{-7}$	$3.39 \cdot 10^{-7}$	$4.02 \cdot 10^{-7}$	$4.04 \cdot 10^{-7}$
0.020	3.571	∞	$5.08 \cdot 10^{-8}$	$4.22 \cdot 10^{-8}$	$4.21 \cdot 10^{-8}$
		$2.84 \cdot 10^{-8}$	$3.9149 \cdot 10^{-8}$	$4.21 \cdot 10^{-8}$	$4.21 \cdot 10^{-8}$
	$c' =$	3.710526	2.634856	2.494639	2.457990

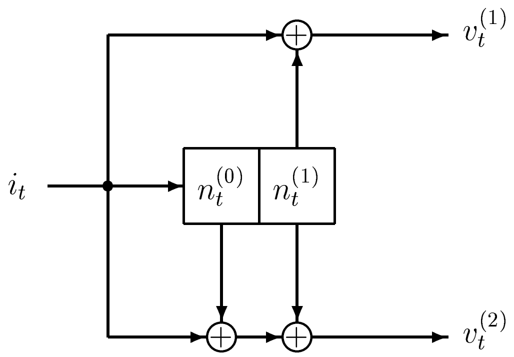

Figure 1: Encoder (5,7); $G^{(1)}(x) = 1 + x^2,\ G^{(2)}(x) = 1 + x + x^2$.

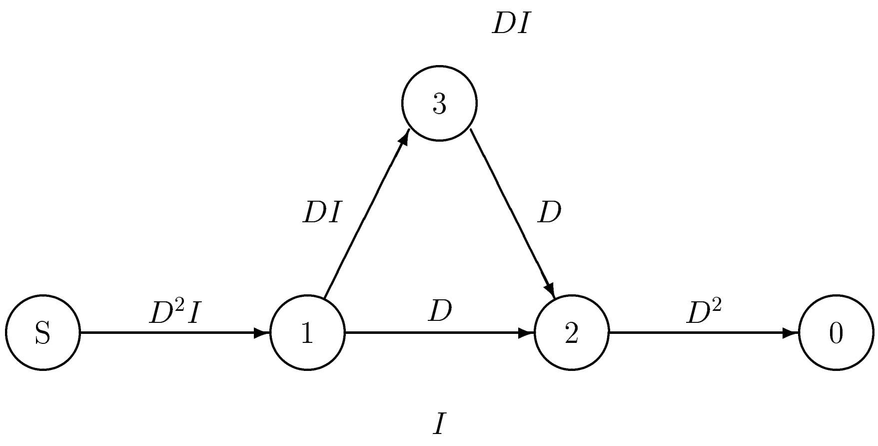

Figure 2: Modified state diagram of the encoder (5,7).

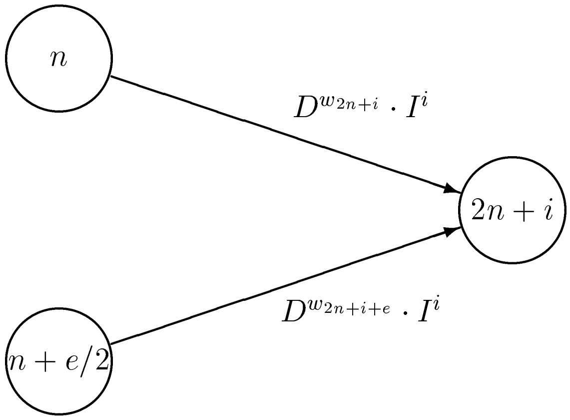

Figure 3: A fragment of the modified state diagram; $n \in \{1, ..., e/2 - 1\}$, $i \in \{0, 1\}$.

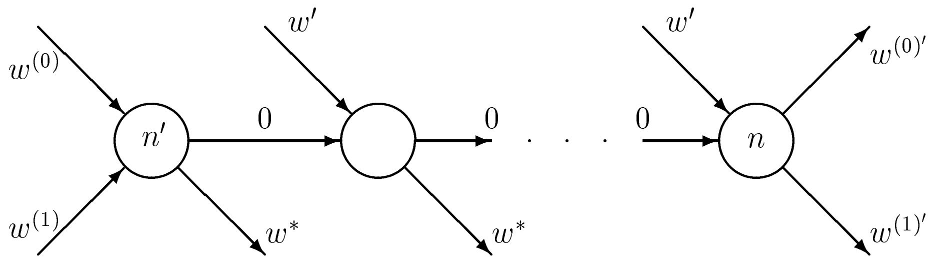

Figure 4: A chain of the zero weight between the vertices n' and n; $w^{(0)}, w^{(1)}, w^{(0)'}, w^{(1)'}, w', w^* > 0$.

Appendix

Proof of Proposition 3.2

It is well known [12] that if coefficients of generator polynomials of the encoder are chosen uniformly at each time instant, then the code symbols assigned to any given path on the MSD, leading from the origin to receiver and having length l, are statistically independent random variables, chosen uniformly from $\{0,1\}$. The number of code symbols corresponding to these paths is equal to Nl, and there are not more than $\binom{l-m}{i}$ paths corresponding to information sequences of weight i. Thus,

$$\begin{aligned}
\bar{t}(d) &\leq \sum_{l\geq m+1} \binom{Nl}{d} \cdot 2^{-Nl} \cdot \sum_{i\geq 1} \binom{l-m}{i} \leq \\
&\leq 2^{-m} \cdot \sum_{l\geq m+1} \binom{Nl}{d} \cdot 2^{-(N-1)l}, \\
\bar{t}'(d) &\leq \sum_{l\geq m+1} \binom{Nl}{d} \cdot 2^{-Nl} \cdot \sum_{i\geq 1} i \cdot \binom{l-m}{i} = \\
&= 2^{-m} \cdot \sum_{l\geq m+1} \binom{Nl}{d} \cdot 2^{-(N-1)l} \cdot \frac{l-m}{2}.
\end{aligned} \tag{A.1}$$

Let us introduce the functions :

$$\begin{aligned}
F(d) &= \sum_{l\geq 1} \binom{Nl}{d} \cdot 2^{-(N-1)l}, \\
F'(d) &= \frac{1}{2} \cdot \sum_{l\geq 1} \binom{Nl}{d} \cdot 2^{-(N-1)l} \cdot l.
\end{aligned} \tag{A.2}$$

Then

$$\begin{aligned}
\bar{t}(d) &\leq 2^{-m} \cdot F(d), \\
\bar{t}'(d) &\leq 2^{-m} \cdot \left(F'(d) - \frac{m}{2} \cdot F(d)\right) < 2^{-m} \cdot F'(d),
\end{aligned}$$

as it follows from (A.1).

Using the identity :

$$\binom{Nl}{d} = \sum_{w=0}^{N} \binom{N}{w} \cdot \binom{N(l-1)}{d-w}$$

and interchanging the order of summation in (A.2), we write :

$$\begin{aligned} F(d) &= 2^{-(N-1)} \cdot \sum_{w=0}^{N} \binom{N}{w} \cdot \sum_{l} \binom{N(l-1)}{d-w} \cdot 2^{-N(l-1)} \cdot 2^{l-1} = \\ &= 2^{-(N-1)} \cdot \sum_{w=0}^{N} \binom{N}{w} \cdot F(d-w) \ \text{ for all } \ d > 0, \end{aligned}$$

where

$$F(0) = \sum_{l \geq 1} 2^{-(N-1)l} = \frac{2^{-(N-1)}}{1 - 2^{-(N-1)}} \leq 1$$

and

$$F(d) = 0 \ \text{ for all } \ d < 0.$$

Similar transformation leads to the equation :

$$F'(d) = 2^{-(N-1)} \cdot \sum_{w=0}^{N} \binom{N}{w} \cdot F'(d-w) + \frac{1}{2} \cdot F(d) \ \text{ for all } \ d > 0,$$

where

$$F'(0) = \frac{1}{2} \cdot \sum_{l \geq 1} 2^{-(N-1)l} \cdot l = \frac{2^{-N}}{(1 - 2^{-(N-1)})^2} \leq 1$$

and

$$F'(d) = 0 \ \text{ for all } \ d < 0.$$

For any $\overline{c}$, the functions $(\overline{c})^d$ and $(d+1) \cdot (\overline{c})^d$ upper-bound $F(d)$ and $F'(d)$ for all $d \leq 0$. Therefore [13], $\overline{c}$ satisfies (3.13) if

$$\begin{aligned} (\overline{c})^d &\geq 2^{-(N-1)} \cdot \sum_{w=0}^{N} \binom{N}{w} \cdot (\overline{c})^{d-w}, \\ (d+1) \cdot (\overline{c})^d &\geq 2^{-(N-1)} \cdot \sum_{w=0}^{N} (d-w+1) \cdot \binom{N}{w} \cdot (\overline{c})^{d-w} + \frac{1}{2} \cdot (\overline{c})^d \end{aligned}$$

or

$$\begin{aligned}
1 &\geq 2^{-(N-1)} \cdot (1 + 1/\overline{c})^N, \\
d+1 &\geq 2^{-(N-1)} \cdot \left(d + 1 - \frac{N/\overline{c}}{1+1/\overline{c}}\right) \cdot (1 + 1/\overline{c})^N + 1/2.
\end{aligned} \tag{A.3}$$

Let $\overline{c}$ be defined by (3.14). Then

$$1 = 2^{-(N-1)} \cdot (1 + 1/\overline{c})^N.$$

Besides, this coefficient also satisfies the second inequality in (A.3), since

$$\overline{c} \leq 2N - 1$$

and

$$d + 1 \geq d + 1 - \frac{N/\overline{c}}{1+1/\overline{c}} + 1/2.$$

Q.E.D.

References

[1] A.J.Viterbi, "Convolutional codes and their performance in communication systems," *IEEE Trans.Commun.Techn.,* COM-19, pt.2, no.5, 1971, pp.751-772.

[2] L. van de Meeberg, "A tightened upper bound on the error probability of binary convolutional codes with Viterbi decoding," *IEEE Trans.Inform.Theory* IT-20, no.3, 1974, pp.389-391.

[3] M.Cedervall, R.Johannesson, and K.Sh.Zigangirov, "A new upper bound on the first-event error probability for maximum-likelihood decoding of fixed binary convolutional codes," *IEEE Trans.Inform.Theory* IT-30, no.5, 1984, pp.762-766.

[4] J.L.Massey and M.K.Sain, "Inverses of linear sequential circuits," *IEEE Trans.Computers,* C-17, no. 4, 1968, pp.330-337.

[5] P.Fitspatrick and G.H.Norton, "Linear recurring sequences and the weight enumerator of a convolutional code," *Electronic Letters,* Vol.27, no.1, 3rd Jan.1991, pp.98-99.

[6] V.B.Balakirsky, "Algorithms for estimations of the error probabilities at the output of the Viterbi decoder," *Tekhnika sredstv svyazi,* Ser.'Tekhnika radiosvyazi', M.: 1981, no. 5, pp.12-23.

[7] M.Cedervall and R.Johannesson, "A fast algorithm for computing distance spectrum of convolutional codes," *IEEE Trans.Inform.Theory* IT-35, no.5, 1989, pp.1146-1159.

[8] J.Connan, "The weight spectra of some short low-rate convolutional codes," *IEEE Trans.Commun.Techn.,* COM-33, no. 9, 1984, pp.1050-1053.

[9] M.Cedervall, *Private communication,* Nov., 1993.

[10] V.B.Balakirsky, "A neccessary and sufficient condition for time-variant convolutional encoders to be noncatastrophic," *Lecture Notes in Computer Science,* no. 781, 1993, pp.1-10.

[11] J.B.Anderson and K.Balachandran, "Decision depths of convolutional codes," *IEEE Trans.Inform.Theory,* IT-35, no.2, 1989, pp.455-459.

[12] A.J.Viterbi and J.K.Omura, *Principles of Digital Communication and Coding.* New York : McGraw-Hill, 1979.

[13] K.Sh.Zigangirov, *Procedures of Sequential Decoding* (in Russian). Moscow: "Svyaz", 1974.