On the distribution of computation for sequential decoding using the stack algorithm

Johannesson, Rolf

Published in:
IEEE Transactions on Information Theory

1979

On the Distribution of Computation for Sequential Decoding Using the Stack Algorithm

ROLF JOHANNESSON, MEMBER, IEEE

Abstract—**An analytical procedure is presented for generating the computational distribution for the Zigangirov–Jelinek stack algorithm. Multitype branching processes are employed to develop a procedure for estimating sequential decoding computation, without the need for simulation, but with sufficient accuracy to be a valid design tool. At information rates about the cutoff rate R_0 the calculated computational performance is virtually identical to that obtained by time consuming simulations.**

I. Introduction

ALTHOUGH sequential decoding demonstrates an inherent inability to deal effectively with severe bursts of noise, sequential decoding techniques are routinely employed in space communication systems. Jacobs and Berlekamp [1], Savage [2], and Jelinek [3] have shown that the behavior of systems using sequential decoding is limited by a computational distribution (conditioned on correct decoding) which is essentially Pareto. More specifically, Jacobs and Berlekamp showed that the computation required to decode the first Λ branches of the code tree $C(\Lambda)$ has a distribution satisfying

$$P[C(\Lambda) \geq N] \geq N^{-\rho} e^{-O(\sqrt{\ln N})}, \qquad (1)$$

where $O(\sqrt{\ln N})$ is an asymptotically unimportant term and the Pareto exponent ρ is the solution of $R = \hat{E}(\rho)/\rho$. The function $\hat{E}_0(\rho)$ is the smallest convex $\cap$ function greater than or equal to $E_0(\rho)$ defined by Gallager [4] and R is the code rate. Upper bounds on the computational distribution were given by Savage for integer values of ρ and by Jelinek for all real values of ρ. The latter bounds together with the lower bound (1) prove that the decoding effort random variables are asymptotically Paretean. *Asymptotically* (for large values of N), the lower bound (1) gives an accurate description of the computational distribution for a practical sequential decoder [5]. In this paper we develop a method for estimating the sequential decoding computation for *small* values of N, without the need for simulation. Hence our method should be a useful complement to the Pareto asymptote for the design of future sequential decoding systems.

Our method for analyzing the computational dynamics of one of the principal forms of sequential decoding, the stack algorithm [6], relies heavily on the theory of multitype branching processes [7]–[9]. In Section II of this paper we give a brief review of tree codes and sequential decoding. In Section III we present some basic relations for the appropriate model for the analysis of the tree codes described in Section II, viz., the multitype branching process. Next, in Section IV, we employ the multitype branching process technique to determine the distribution of the minimum of the cumulative metrics along the transmitted path in the code tree, which is a first step towards our goal, viz., the determination of the computational distribution over the region where the Pareto distribution is not useful. In Section V we determine the distribution of the number of computations made by the decoder in order to decode the first branch in the tree. This random variable, which we denote by C_1, can be used as an approximation of the average number of computations per decoded branch C_{AV}. The numerical results are discussed and comparisons with simulations are made in Section VI. As an outcome of these comparisons, we conclude that the computational distribution can be estimated with sufficient accuracy to warrant the use of our procedure as a design tool, and that multitype branching processes are very useful in studying sequential decoding.

II. Tree Codes and Sequential Decoding

Without loss of generality we shall restrict our attention to binary (semi-infinite) tree codes of rate $R = 1/2$.

A binary *tree code* of rate $R = 1/2$ (see Fig. 1) is formed by assigning two channel input symbols to each branch of a rooted tree, in which two branches stem from the root node and from each successive node. To encode a binary information sequence, we follow the upper or lower branch depending on whether the information symbol is 0 or 1.

To obtain theoretical results about tree codes, we must often restrict our consideration to the class of *random tree codes*, in which each channel input symbol on the branches in the tree is chosen independently according to a specified probability distribution. However, a constant

Manuscript received February 13, 1978; revised October 10, 1978. This research was supported by the National Aeronautics and Space Administration under NASA Grant NGL 5025 at the University of Notre Dame in liaison with the Communications and Navigation Division of the Goddard Space Flight Center. This paper was previously presented at the 1976 IEEE International Symposium on Information Theory, Ronneby, Sweden, June 21–24.

The author is with the Department of Electrical Engineering, University of Lund, P.O. Box 725, S-220 07 Lund, Sweden.

0018-9448/79/0500-0323$00.75

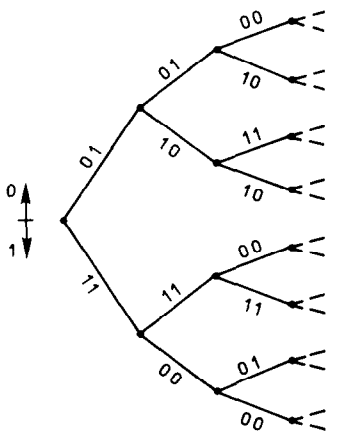

Fig. 1. An example of a binary tree code of rate 1/2 for a binary input channel.

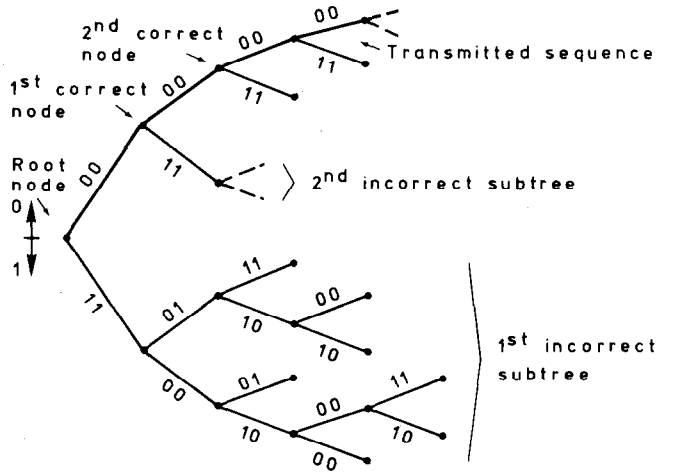

Fig. 2. An example of a complementary + random code tree which is partially explored by a sequential decoder.

linear tree code, i.e., a constant convolutional code, is preferred in a practical system.

In our analysis we shall consider the class of *complementary + random tree codes*, a ficticious entity that is shown by simulation results to be reasonably close to a good model for constant convolutional codes which have column distance $d_0=2$ (the optimum value for all tree codes of rate $R=1/2$ [10]). In a complementary + random tree code the channel input symbols on the branches on the transmitted path are all zeros, and the channel input symbols on the incorrect branches stemming from nodes on the correct path are all ones. For all other branches each channel symbol is chosen independently according to a specified probability distribution. An example of a complementary + random tree code is shown in Fig. 2.

Let z be the Fano metric [11] assigned to the branches in the tree. For memoryless channels the values of z are given by

$$z=\sum_j\left(\log_2\frac{P[y_j|x_j]}{P[y_j]}-R\right) \tag{2}$$

where x_j is the jth channel input symbol on the particular branch, y_j is the corresponding received symbol, and $P[y_j]$ is the *a priori* probability for the received symbol. Using the cumulative metrics for the explored paths in the tree, a sequential decoding algorithm makes early rejections of paths that are unlikely to be the transmitted one. Thus the sequential decoder explores the tree only partially. There exist two principal forms of sequential decoders that perform this exploration in an efficient way, viz., the Fano algorithm [12] and the stack algorithm [6], [13]. In Section V we shall analyze the stack algorithm. It can be described as follows.

Step 0: Place the root node in the stack.

Step 1: Delete (one of) the node(s) with the greatest node value, i.e., the greatest cumulative metric, from the stack and place its two successors into the stack.

Step 2: Order the stack according to increasing node values with the greatest node value at the top.

Step 3: If the topmost node is a node at the end of the tree, stop and choose the path leading to this node as the decoded information sequence. Otherwise go to step 1.

We shall consider only semi-infinite code trees and hence ignore the stopping rule in step 3.

III. Multitype Branching Processes

The theory of multitype branching processes is useful in the study of sequential decoding algorithms. In this section we present some fundamental results of multitype branching processes. Our presentation is based on Mode [8].

Suppose we have a population of individuals of m different types. Each individual produces a random number of offspring and each offspring is of a randomly determined type. The growth of this population is described by a multitype branching process.

The offspring distribution can be expressed in a simple closed form if generating functions are used. For any vector $\boldsymbol{r}=(r_1,r_2,\cdots,r_m)$ with nonnegative integer components, let $p_i(\boldsymbol{r})$ be the probability that an individual of type i produces r_j offspring of type j, for all $j=1,2,\cdots,m$. For all m-dimensional vectors of complex numbers $\boldsymbol{s}=(s_1,s_2,\cdots,s_m)$ such that $\max_k|s_k|\leqslant 1$, define

$$f_i(\boldsymbol{s})=\sum_{\boldsymbol{r}}p_i(\boldsymbol{r})s_1^{r_1}s_2^{r_2}\cdots s_m^{r_m} \tag{3}$$

to be the generating function of the probabilities $p_i(\boldsymbol{r})$.

Let the $m\times m$ matrix (p_{ij}) be the Markov transition matrix, i.e., p_{ij} is the probability that a given offspring produced by an individual of type i is of type j. Thus the probability distribution for the type of offspring from an individual of type i has the generating function

$$g(i,\boldsymbol{s})=\sum_{j=1}^{m}p_{ij}s_j. \tag{4}$$

If we let the random variable H_i represent the total number of offspring produced by an individual of type i,

we have

$$h_i(s)=\sum_{n=0}^{\infty} P[H_i=n]s^n. \tag{5}$$

We obtain an explicit form for the offspring distribution by combining (4) and (5). Thus

$$f_i(s)=h_i(g(i,s)) \tag{6}$$

for all complex vectors $s=(s_1,s_2,\cdots,s_m)$ such that $\max_k |s_k|\leqslant 1$.

Next we shall use the offspring distribution generating function to obtain the distribution of the generation sizes. The initial individual (the progenitor) is called the initial generation, the offspring of the initial individual are called the first generation, the offspring of the members of the first generation are called the second generation, and so on.

Let the random variables $Z_j(n), j=1,2,\cdots,m$, represent the number of individuals of type j in generation n and let the number of individuals of various types in generation $n=0,1,2,\cdots$ be represented by a sequence of vector-valued random variables $(\boldsymbol{Z}(n))$. If the initial individual is of type i, we set

$$\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_i \tag{7}$$

where

$$\boldsymbol{\epsilon}_i=(\delta_{1i},\delta_{2i},\cdots,\delta_{mi}) \tag{8}$$

and δ_{ji} is a Kronecker delta. For generations $n\geqslant 1$ we have

$$\boldsymbol{Z}(n)=(Z_1(n),Z_2(n),\cdots,Z_m(n)). \tag{9}$$

For $i=1,2,\cdots,m$ and for any complex vector $s=(s_1,s_2,\cdots,s_m)$ with $\max_k |s_k|\leqslant 1$, let the generating function for the probability distribution of the size of generation $n=0,1,2,\cdots$ be

$$F_n(i,s)=\sum_{\boldsymbol{r}} P[\boldsymbol{Z}(n)=\boldsymbol{r}|\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_i]s_1^{r_1}s_2^{r_2}\cdots s_m^{r_m}. \tag{10}$$

It follows immediately that

$$F_0(i,s)=s_i \tag{11}$$

and

$$F_1(i,s)=f_i(s). \tag{12}$$

If we set

$$\boldsymbol{f}(1,s)=(f_1(s),\cdots,f_m(s)) \tag{13}$$

and

$$\boldsymbol{f}(n,s)=(f_1(\boldsymbol{f}(n-1,s)),\cdots,f_m(\boldsymbol{f}(n-1,s))), \quad n\geqslant 2 \tag{14}$$

then it follows that

$$F_n(i,s)=f_i(\boldsymbol{f}(n-1,s)) \tag{15}$$

for $n\geqslant 2$ and $i=1,2,\cdots,m$. We shall exploit the recursiveness of (15) in Section V.

Finally, we shall give some results regarding the probability of extinction. Let $q_i(n)$ be the probability of extinction by the nth generation given that the progenitor is of type i. Thus

$$q_i(n)=P[\boldsymbol{Z}(n)=\boldsymbol{0}|\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_i]. \tag{16}$$

We notice that

$$q_i(n)\leqslant q_i(n+1) \tag{17}$$

and that q_i, the probability of eventual extinction, can be obtained as the limit

$$q_i=\lim_{n\to\infty} q_i(n) \tag{18}$$

for $i=1,2,\cdots,m$.

From (3) it follows that, if we interpret 0^0 as 1, then

$$q_i(1)=f_i(\boldsymbol{0}). \tag{19}$$

Let $\boldsymbol{q}(n)=(q_1(n),\cdots,q_m(n))$. It follows from (14) and (15) that, for $n\geqslant 1$,

$$q_i(n)=f_i(\boldsymbol{q}(n-1)) \tag{20}$$

for $i=1,2,\cdots,m$. Taking the limit as $n\to\infty$ we find that the probability of extinction $\boldsymbol{q}=(q_1,\cdots,q_m)$ is the smallest nonnegative root of the vector equation

$$\boldsymbol{q}=\boldsymbol{f}(1,\boldsymbol{q}). \tag{21}$$

IV. Distribution of the Minimum of the Cumulative Metrics

The distribution of the minimum of the cumulative metrics for the correct path is fundamental in characterizing the performance of a sequential decoder.

Let M_k be the cumulative metric for the first k branches of the correct path, i.e.,

$$M_k=\sum_{j=1}^{k} z_j \tag{22}$$

where z_j is the branch metric for the jth branch. Let D_k, $k=0,1,2,\cdots$ be the difference between the cumulative metric and the smallest succeeding value, i.e.,

$$D_k=M_k-M_{\min,k} \tag{23}$$

where

$$M_{\min,k}=\min\{M_k,M_{k+1},\cdots\}. \tag{24}$$

The nonnegative random variables D_k all have the same distribution since the metrics for the correct path stemming from the kth node have the same statistical character for all k.

We shall consider only memoryless channels with binary output alphabets and thus a branch metric which adopts only three different values $\{a,-b,-c\}$, where

$$0<a<b<c \tag{25}$$

and

$$c=2b+a. \tag{26}$$

The generalization to larger output alphabets or soft decision decoding is straightforward. Furthermore, since the metrics can be scaled and rounded into integers with any degree of accuracy, we shall consider only integer-valued metrics.

Let E_i, $i=0,1,2$, be the event that exactly i of the two channel symbols on a branch are erroneous, and let

$$P[E_0]=P[z_j=a]=p_0 \tag{27a}$$

$$P[E_1]=P[z_j=-b]=p_1 \tag{27b}$$

$$P[E_2]=P[z_j=-c]=p_2 \tag{27c}$$

for $j\geq 1$.

A multitype branching process can be used to determine the probabilities $P[D_k\geq i-1]$, $i=1,2,\cdots,m$ in the following way. Let the progenitor of the process correspond to the kth node on the correct path and be of type i, and let, in general, the individual of the nth generation correspond to the $(k+n)$th node on the correct path. Individuals of type 1 have no offspring, i.e., type 1 is an absorbing state, while all other individuals have one offspring. Thus if we let the generating function $h_i(s)$ represent the distribution of the number of offspring, we have

$$h_i(s)=\begin{cases}1, & i=1,\\ s, & 1<i\leq m.\end{cases} \tag{28}$$

Suppose that the individual belonging to the nth generation is of type j, $1<j\leq m-a$. Then, if $j+z_{k+n+1}\geq 1$, let the offspring be of type $j+z_{k+n+1}$; otherwise, let it be of type 1. If we let $g(j,s)$ be the generating function defined in (4) we have

$$g(j,s)=\begin{cases}s_1, & j=1\\ (p_2+p_1)s_1+p_0s_{j+a}, & 1<j\leq 1+b\\ p_2s_1+p_1s_{j-b}+p_0s_{j+a}, & 1+b<j\leq 1+c\\ p_2s_{j-c}+p_1s_{j-b}+p_0s_{j+a}, & 1+c<j\leq m-a\\ s_j, & m-a<j\leq m\end{cases} \tag{29}$$

where the last row is added to give the required finite number of types. However, since we are interested only in the first part of the distribution function, the number of types m can be made large enough not to affect the results.

From (6) and (28) it follows that

$$f_i(s)=\begin{cases}1, & i=1,\\ g(i,s), & 1<i\leq m.\end{cases} \tag{30}$$

From the correspondence between branch metrics and types we have

$$P[D_k\geq i-1]=P[\text{extinction}|\boldsymbol{Z}(0)=\epsilon_i] \tag{31}$$

where $\boldsymbol{Z}(0)$ is defined in Section III. From (16), (18), and (31) it follows that

$$P[D_k\geq i-1]=q_i \tag{32}$$

where q_i is the smallest nonnegative solution of

$$q_i=f_i(\boldsymbol{q}). \tag{33}$$

From (20) it is clear that (33) can be solved iteratively, and in practice the convergence $q_i(n)\to q_i$ is usually rapid. If we let π_l, $l=0,1,2,\cdots$, denote the probability distribution for the random variable D_k, then we have immediately from (32)

$$\pi_l=P[D_k=l]=q_{l+1}-q_{l+2}. \tag{34}$$

In practice we can often choose $a=1$ with little loss of accuracy. For this special case Massey [14] noticed that the infinite sequence $\cdots,D_{k+1},D_k,D_{k-1},\cdots$ is a queuing process which is equivalent to a unit-decrement Markov chain. From any state $i\geq 1$ in this Markov chain there is a single transition of one step ($a=1$) toward the origin state 0 and two transitions $(-b,-c)$ away from the origin state. From the fact that the unconditional drift for the Markov chain is zero, the stationary probabilities $\pi_0,\pi_1,\pi_2,\cdots$ can be evaluated directly from

$$\pi_j=\begin{cases}(p_0)^{-1}(p_0-bp_1-cp_2), & j=0\\ (p_0)^{-1}(1-p_0)\pi_0, & j=1\\ (p_0)^{-1}\pi_{j-1}, & 1<j\leq b\\ (p_0)^{-1}(\pi_{j-1}-p_1\pi_{j-1-b}), & b<j\leq c\\ (p_0)^{-1}(\pi_{j-1}-p_1\pi_{j-1-b}-p_2\pi_{j-1-c}), & j>c.\end{cases} \tag{35}$$

As an example we choose a binary symmetric channel (BSC) with crossover probability $p=0.045$ which corresponds to transmission at the cutoff rate $R_0=0.50$ [15]. We have immediately

$$\begin{aligned}p_0&=(1-p)^2=0.912,\\ p_1&=2p(1-p)=0.086,\\ p_2&=p^2=0.002,\end{aligned} \tag{36}$$

and the Fano metric set is $\{+1,-4,-9\}$. The probabilities π_0, π_1, π_2, π_3, π_4, π_5, $\pi_6,\cdots$ are 0.603, 0.058, 0.064, 0.070, 0.077, 0.027, 0.024,$\cdots$. We notice that as many as 60.3 percent of the nodes on the correct path will be "breakout nodes" [4], and the interesting (but not surprising) fact that

$$\pi_1<\pi_2<\pi_3<\pi_4 \tag{37}$$

for this set of metric values.

V. Distribution of C_1

We call the node at depth 1 on the correct path the *first correct node* (Fig. 2). The node at depth 2 on the correct path is called the *second correct node* and so on. Let C_j, $j=1,2,\cdots$ be the number of computations made by the sequential decoder in order to decode the jth correct node (or, equivalently, to decode the branch between the $(j-1)$th correct node and the jth correct node). That is, the random variable C_j equals the single computation made in order to extend the $(j-1)$th correct node plus the number of computations made in order to extend the nodes in the jth incorrect subtree (i.e., the incorrect subtree stemming from the $(j-1)$th correct node).

We note that for a tree of unbounded length the random variables $C_1,C_2,C_3,\cdots$ are identically distributed

but *not* independent. The distribution of C_j is important for determining the performance of a sequential decoder. Of even greater importance is the average number of computations per decoded branch,

$$C_{AV} = \lim_{L\to\infty} C_{AV}(L) = \lim_{L\to\infty} \frac{1}{L}\sum_{j=1}^{L} C_j. \qquad (38)$$

However, in this paper we shall deal primarily with the distribution of C_j and in particular we shall compute the distribution of C_1.

It is easily proved (e.g., by induction) that for binary trees the number of computations C_j equals the number of *terminal nodes*, i.e., nodes for which the cumulative metric is less than the minimum cumulative metric along the correct path, in the jth incorrect subtree. For reasons that will become obvious we concatenate strings of nodes of infinite length to the terminal nodes in the first incorrect subtree (Fig. 3). Since the strings do not affect the number of nodes which are not extended, the random variable C_1 equals the number of nodes at depth ∞ in the first incorrect subtree.

We shall use the multitype branching process described in Section III to compute the distribution of this latter random variable when the stack algorithm is used to decode an equiprobable complementary + random tree code.

Let the node at depth 1 in the first incorrect subtree be the progenitor (the initial generation) and, in general, let the nodes at depth $n+1$, $n=1,2,\cdots$ in the first incorrect subtree constitute the nth generation in a multitype branching process.

The values of the cumulative metrics determine to which "type" the different nodes in the first incorrect subtree belong. Let a node be of type 1 if its cumulative metric M is less than or equal to a threshold value M_t. The random variable M_t represents the largest possible cumulative metric for nodes in the first incorrect subtree that will not be extended by the sequential decoding algorithm. In general, let the nodes with the cumulative metrics $M = M_t + i - 1$, $i = 1,2,\cdots,m$, be type i nodes.

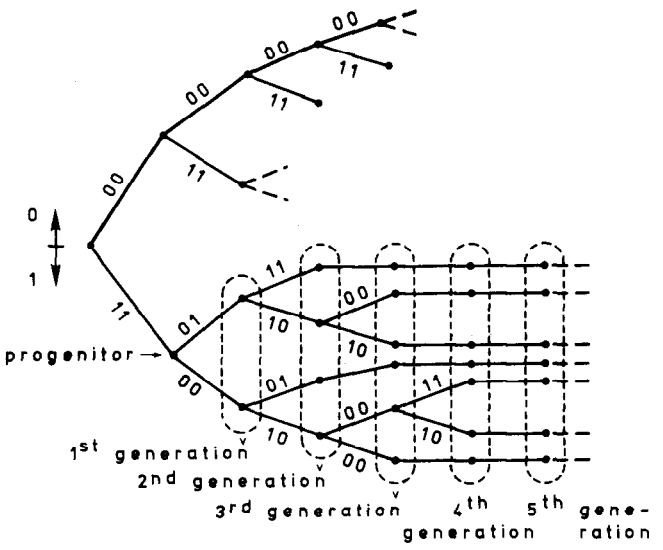

Fig. 3. An example of a code tree with strings of nodes of infinite length concatenated to the terminal nodes in the first incorrect subtree. The number of offspring in the fourth generation equals the number of computations made in the first incorrect subtree in Fig. 2.

For computational and practical reasons we are interested only in the nonasymptotic part of the probability distribution for the random variable C_1. Thus we make the finite number of types m so large that we do not have to take its finiteness into account.

The offspring of the original terminal nodes are the nodes on the strings. Let the branch metrics in the strings all be zero. Thus all nodes on the strings will be of type 1. From the description of the sequential decoding algorithm and from the relation between the cumulative metrics and the types, it is clear that if, as an example, we choose the branch metrics $\{+1, -4, -9\}$, the $m \times m$ Markov transition matrix (p_{ij}), defined in Section III, can be written, for $m = 15$ say, as

$$(p_{ij}) = \begin{array}{c|ccccccccccccccc} & j=1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ \hline i=1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 3/4 & 0 & 1/4 & & & & & & & & & & & & 0 \\ 3 & 3/4 & & & 1/4 & & & & & & & & & & & 0 \\ 4 & 3/4 & & & & 1/4 & & & & & & & & & & 0 \\ 5 & 3/4 & & & & & 1/4 & & & & & 0 & & & & 0 \\ 6 & 1/4 & 1/2 & & & & & 1/4 & & & & & & & & 0 \\ 7 & 1/4 & & 1/2 & & & & & 1/4 & & & & & & & 0 \\ 8 & 1/4 & & & 1/2 & & & & & 1/4 & & & & & & & 0 \\ 9 & 1/4 & & & & 1/2 & & 0 & & & 1/4 & & & & & 0 \\ 10 & 1/4 & & & & & 1/2 & & & & & 1/4 & & & & & 0 \\ 11 & 0 & 1/4 & & & & & 1/2 & & & & & 1/4 & & & & 0 \\ 12 & 0 & & 1/4 & & 0 & & & 1/2 & & & & & 1/4 & & & 0 \\ 13 & 0 & & & 1/4 & & & & & 1/2 & & & & & & 1/4 & 0 \\ 14 & 0 & & 0 & & 1/4 & & & & & 1/2 & & & & & & 1/4 \\ 15 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \qquad (39)$$

and the corresponding generating functions are

$$g(i,s)=\begin{cases} s_1, & i=1 \\ 3/4s_1+1/4s_{i+1}, & 1<i\leqslant 5 \\ 1/4s_1+1/2s_{i-4}+1/4s_{i+1}, & 5<i\leqslant 10 \\ 1/4s_{i-9}+1/2s_{i-4}+1/4s_{i+1}, & 10<i\leqslant m-1 \\ s_i, & i=m. \end{cases} \tag{40}$$

Assume, in general, that the set of branch metrics is $\{a,-b,-c\}$ where a, b, and c are positive integers and $a<b<c$. The generalization of (40) is obvious:

$$g(i,s)=\begin{cases} s_1, & i=1 \\ 3/4s_1+1/4s_{i+a}, & 1<i\leqslant 1+b \\ 1/4s_1+1/2s_{i-b}+1/4s_{i+a}, & 1+b<i\leqslant 1+c \\ 1/4s_{i-c}+1/2s_{i-b}+1/4s_{i+a}, & 1+c<i\leqslant m-a \\ s_i, & m-a<i\leqslant m. \end{cases} \tag{41}$$

All individuals of type 1 produce exactly one offspring each while all individuals of all other types produce exactly two offspring each. Thus the generating functions reflecting the distribution of the number of offsprings produced by an individual of type i are given by

$$h_i(s)=\begin{cases} s, & i=1 \\ s^2, & i=2,3,\cdots,m. \end{cases} \tag{42}$$

Then it follows from (6), (41), and (42) that

$$f_i(s)=\begin{cases} s_1, & i=1 \\ (3/4s_1+1/4s_{i+a})^2, & 1<i\leqslant 1+b \\ (1/4s_1+1/2s_{i-b}+1/4s_{i+a})^2, & 1+b<i\leqslant 1+c \\ (1/4s_{i-c}+1/2s_{i-b}+1/4s_{i+a})^2, & 1+c<i\leqslant m-a \\ s_i^2, & m-a<i\leqslant m. \end{cases} \tag{43a}$$

For purposes that will become obvious, it is useful to expand the squares in (43a). Thus we write

$$f_i(s)=\begin{cases} s_1, & i=1 \\ 9/16s_1^2+1/16s_{i+a}^2+3/8s_1s_{i+a}, & 1<i\leqslant 1+b \\ 1/16s_1^2+1/4s_{i-b}^2+1/16s_{i+a}^2 & \\ \quad +1/4s_1s_{i-b}+1/8s_1s_{i+a} & \\ \quad +1/4s_{i-b}s_{i+a}, & 1+b<i\leqslant 1+c \\ 1/16s_{i-c}^2+1/4s_{i-b}^2+1/16s_{i+a}^2 & \\ \quad +1/4s_{i-c}s_{i-b}+1/8s_{i-c}s_{i+a} & \\ \quad +1/4s_{i-b}s_{i+a}, & 1+c<i\leqslant m-a \\ s_i^2, & m-a<i\leqslant m. \end{cases} \tag{43b}$$

Let $G_n(i,s)$ be the generating function representation of the probability distribution for the total number of individuals in the nth generation, given that the progenitor is of type i. Hence for all positive integers r,

$$G_n(i,s)=\sum_{r=1}^{\infty} P\left[\sum_{j=1}^{m} Z_j(n)=r|\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_i\right]s^r \tag{44}$$

where $Z_j(n)$ is defined in Section III. We are interested in determining the probability distribution for the random variable C_1. From (44) and from the fact that the number of terminal nodes equals C_1, it is clear that the distribution is given by the generating function

$$G_\infty(i,s)=\lim_{n\to\infty} G_n(i,s). \tag{45}$$

If we let $r=\sum_{j=1}^{m} r_j$ and $\boldsymbol{s}^*=(s,s,\cdots,s)$ then it follows from (10) and (44) that

$$G_n(i,s)=F_n(i,\boldsymbol{s}^*). \tag{46}$$

For every complex number s such that $|s|\leqslant 1$, we let

$$G_\infty(i,s)=\sum_{r=1}^{\infty} c(i,r)s^r \tag{47}$$

where

$$c(i,r)=P[C_1=r|\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_i]. \tag{48}$$

It is perhaps somewhat surprising that there is a simple way to compute the probabilities $c(i,r)$. From (13)–(15), (43b), and (45)–(47), it follows that the probabilities $c(i,r)$ for $i=1,2,\cdots,m$ and $r\geqslant 3$ can be computed from the values of $c(i,r')$, $i=1,2,\cdots,m$ and $r'<r$. (We notice that this is not true for a general generating function $f_i(s)$, e.g., it is not true if in (43b) $f_1(s)=1$.) For $r=1,2$ we have immediately

$$c(i,1)=\begin{cases} 1, & i=1 \\ 0, & i>1 \end{cases} \tag{49a}$$

and

$$c(i,2)=\begin{cases} 0, & i=1 \\ 9/16, & 1<i\leqslant 1+b \\ 1/16, & 1+b<i\leqslant 1+c \\ 0, & i>1+c. \end{cases} \tag{49b}$$

From (15), (43b), and (46), it follows that for $r\geqslant 3$ we have

$$c(i,r)=\begin{cases} 0, & i=1 \\ 1/16\sum_{j=1}^{r-1} c(i+a,j)c(i+a,r-j) & \\ \quad +3/8c(1,1)c(i+a,r-1), & 1<i\leqslant 1+b \\ 1/4\sum_{j=1}^{r-1} c(i-b,j)c(i-b,r-j) & \\ \quad +1/16\sum_{j=1}^{r-1} c(i+a,j)c(i+a,r-j) & \\ \quad +1/4c(1,1)c(i-b,r-1) & \\ \quad +1/8c(1,1)c(i+a,r-1) & \\ \quad +1/4\sum_{j=1}^{r-1} c(i-b,j)c(i+a,r-j), & \\ & 1+b<i\leqslant 1+c \end{cases} \tag{49c}$$

$$c(i,r)=\begin{cases}1/16\sum_{j=1}^{r-1}c(i-c,j)c(i-c,r-j)\\ \quad+1/4\sum_{j=1}^{r-1}c(i-b,j)c(i-b,r-j)\\ \quad+1/16\sum_{j=1}^{r-1}c(i+a,j)c(i+a,r-j)\\ \quad+1/4\sum_{j=1}^{r-1}c(i-c,j)c(i-b,r-j)\\ \quad+1/8\sum_{j=1}^{r-1}c(i-c,j)c(i+a,r-j)\\ \quad+1/4\sum_{j=1}^{r-1}c(i-b,j)c(i+a,r-j), & 1+c<i\leqslant m-a\\ 0, \quad i>m-a. & \end{cases} \tag{49c}$$

From (49) it follows that

$$c(i,r)=0 \tag{50}$$

for all i and r such that

$$i>1+kc \tag{51}$$

and

$$r<2^{k+1} \tag{52}$$

where $k=0,1,2,\cdots$.

As mentioned before, we are interested only in a finite maximum number of computations made by the sequential decoder in order to decode the first correct node. Let this number be $r_{\max}$. Then from (51) and (52) it follows that if

$$m-a\geqslant 1+c\lfloor\log_2 r_{\max}\rfloor \tag{53}$$

the finiteness of the number of types in the multitype branching process will not affect the computed probability distribution.

Let us express the probability distribution for C_1 as a sum

$$P[C_1=k]=\sum_i P[C_1=k|\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_i]P[\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_i]. \tag{54}$$

From (48) and (54) we have

$$P[C_1=k]=\sum_i c(i,k)P[\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_i]. \tag{55}$$

The probability that the progenitor is of type i, i.e., $P[\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_i]$, can be determined from the stationary probabilities $\pi_0,\pi_1,\pi_2,\cdots$ for the Markov chain described in Section IV. The main difficulty remaining is to describe the effects of resolving ties. In practice ties are usually resolved according to the rule "last in/first out". In our analysis we can handle only a few very special cases, e.g., when the extending of the root node results in the same cumulative metrics for both successors (i.e., a single error occurs during the transmission of the first two channel symbols) then one of the successors is randomly chosen for the top position on the stack. Thus if we let E_j, $j=0,1,2$, denote the event that exactly j of the first two channel symbols are erroneous, then, since we assume that the information symbols are independent and equally likely to be zeros or ones, we have

$$P[\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_1|E_1]=P[\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_2|E_1]=1/2, \quad \text{if } D_1=0 \tag{56}$$

where D_1 is defined in Section IV. The random resolution of this particular type of tie and the linearity of convolutional codes justify the use of a fixed transmitted information sequence, e.g., the all zero sequence, instead of the more realistic random sequence and more practical fixed resolution of this type of tie (e.g., "extend the one-branch first") when simulating the sequential decoding algorithm. From the "last in/first out" rule we also conclude that

$$P[\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_2|E_1]=P[\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_3|E_1]=1/2, \quad \text{if } D_1=1 \tag{57}$$

$$P[\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_1|E_0]=1, \quad \text{if } D_1\leqslant a+c \tag{58}$$

and

$$P[\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_3|E_0]=1, \quad \text{if } D_1=a+c+1. \tag{59}$$

However, it is very difficult to analyze how ties are resolved in general. We shall give upper and lower bounds on the distribution function for C_1, $P[C_1\geqslant k]$, $k=1,2,\cdots$, to avoid these difficulties. When there is no obvious way to resolve the ties we use the policy "in case of ties extend an incorrect node first" to obtain an upper bound and we use the policy "in case of ties extend the correct node first" to obtain a lower bound. The different policies correspond to the following values of the threshold M_t

$$\text{upper bound:} \quad M_t=M_{\min,1}-1 \tag{60}$$

$$\text{lower bound:} \quad M_t=M_{\min,1} \tag{61}$$

where $M_{\min,1}$ is defined by (24).

From the definition of D_1 (23) we have

$$M_{\min,1}=z_1-D_1 \tag{62}$$

where z_1 is the metric for the first branch along the correct path. If ΔM_t is the difference between the cumulative metric for the progenitor $\bar{z}_1$ and the threshold M_t, i.e.,

$$\Delta M_t=\bar{z}_1-M_t \tag{63}$$

then it follows from the relation between the threshold M_t and the different types that

$$P[\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_i]=P[\Delta M_t=i-1]. \tag{64}$$

Thus from (59)–(63) and (64) we have the following generating functions for the probability distributions for the type of progenitor:

$$\begin{aligned}\text{upper bound:} \quad &\sum_{i=1}^{\infty}P[\boldsymbol{Z}(0)=\boldsymbol{\epsilon}_i]s^i\\ &=p_0\left\{\left(\sum_{j=0}^{a+c}\pi_j\right)s+\pi_{a+c+1}s^3+\sum_{j=4}^{\infty}\pi_{j-2+a+c}s^j\right\}\\ &\quad+p_1\left\{\frac{1}{2}\pi_0 s+\frac{1}{2}(\pi_0+\pi_1)s^2+\frac{1}{2}\pi_1 s^2+\sum_{j=4}^{\infty}\pi_{j-2}s^j\right\}\\ &\quad+p_2\left\{\sum_{j=2+a+c}^{\infty}\pi_{j-2-a-c}s^j\right\}\end{aligned} \tag{65a}$$

$$\text{lower bound:} \quad \sum_{i=1}^{\infty} P[Z(0)=\epsilon_i]s^i$$
$$= p_0\left\{\left(\sum_{j=0}^{a+c}\pi_j\right)s + \pi_{a+c+1}s^3 + \sum_{j=3}^{\infty}\pi_{j-1+a+c}s^j\right\}$$
$$+ p_1\left\{\frac{1}{2}\pi_0 s + \frac{1}{2}(\pi_0+\pi_1)s^2 + \frac{1}{2}\pi_1 s^3 + \sum_{j=3}^{\infty}\pi_{j-1}s^j\right\}$$
$$+ p_2\left\{\sum_{j=1+a+c}^{\infty}\pi_{j-1-a-c}s^j\right\} \tag{65b}$$

where

$$p_i = P[E_i], \qquad i=0,1,2. \tag{66}$$

Finally, (49) and (65) can be inserted into (55) to obtain the probability distribution for the random variable C_1.

VI. Discussion of Numerical Results and Simulations

Using the branching processes methods described in Sections IV and V, theoretical upper and lower bounds on the distribution function of the random variable C_1, $P[C_1 \geq N]$, have been evaluated for the case where transmission takes place over a binary symmetrical channel (BSC) with crossover probability $p=0.045$. Since we are only considering rate $R=1/2$ codes—the codes of greatest practical importance—this crossover probability corresponds to transmission at the computational cutoff rate R_0. For this particular channel the Fano metric set is approximately $\{+1,-4,-9\}$. The bounds are evaluated for this "matched" metric set and for the "unmatched" sets $\{+1,-3,-7\}$, $\{+1,-5,-11\}$, and $\{+1,-6,-13\}$. The bounds are shown in Fig. 4 together with the simulations of $P[C_1 \geq N]$ for the complementary + random code.

It is interesting to notice the close agreement between the simulated C_1-curves and the corresponding lower bounds. This is a strong indication that the tie-resolving rule "last in/first out" is near-optimal, which is not surprising since this rule is a "depth first" rule, i.e., it deepens the search, while the "first in/first out" rule broadens the search and is a "parallel" rule [16].

To investigate the relationship between C_1 and C_{AV}, the total number of computations per information digit, as many as 10 000 frames consisting of 256 information symbols augmented by a tail of $M=23$ zeros were transmitted at rate $R=R_0$ and decoded with different metrics. The best presently known fixed convolutional code with memory length $M=23$, viz., the ODP QLI code [10], was used. The results are shown in Fig. 5 together with the corresponding lower bounds obtained with the branching processes method. The great discrepancies at $N=3$ between our lower bounds and the simulated C_1 curves are due to the complementary structure in the convolutional

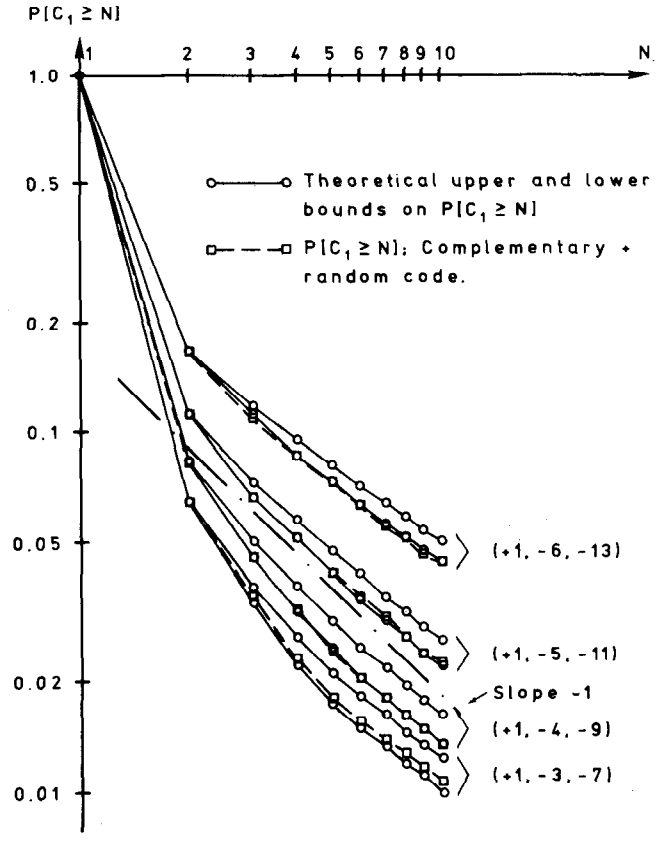

Fig. 4. The computational distribution function $P[C_1 \geq N]$ obtained from sequential decoding simulations for the complementary + random code, and theoretical upper and lower bounds on $P[C_1 \geq N]$.

tree code. Furthermore, we notice that in the range of interesting metric sets (the metric set $\{+1,-3,-7\}$ is not interesting since it is not matched to any BSC!) the lower bounds on $P[C_1 \geq N]$ obtained with the branching processes method are good approximations of both $P[C_1 \geq N]$ and, more important, $P[C_{AV} \geq N]$ for a practical sequential decoder. For $N > 10$ we can extend our bounds with the well-known Pareto asymptote.

Finally in Fig. 6 we compare the branching processes bounds with $P[C_1 \geq N]$ for the complementary + random code and with $P[C_{AV} \geq N]$ for the fixed convolutional code. The channel transition probabilities correspond to transmission at a rate 2.5 percent above R_0 ($p=0.048$) and at a rate 4 percent below R_0 ($p=0.040$). We notice that in the range of interesting rates, viz., about R_0, the branching processes method gives a close approximation to the computational performance of a sequential decoder, viz., $P[C_{AV} \geq N]$.

VII. Remark

In principle the distribution of the average number of computations C_{AV} can be estimated by a procedure which is a straightforward generalization of the procedure developed in Section V. However, the generalized procedure contains sums of such high numbers of terms that it is too time consuming to be an attractive design tool.

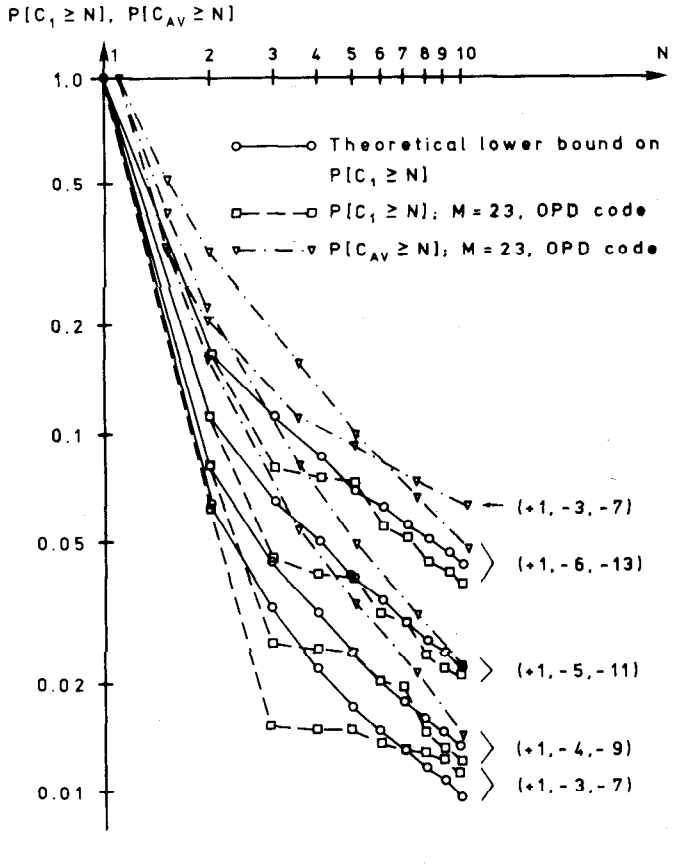

Fig. 5. The computational distribution functions $P[C_1 \geq N]$ and $P[C_{AV} \geq N]$ obtained from sequential decoding simulations for the $M=23$ ODP convolutional code, and theoretical lower bound on $P[C_1 \geq N]$.

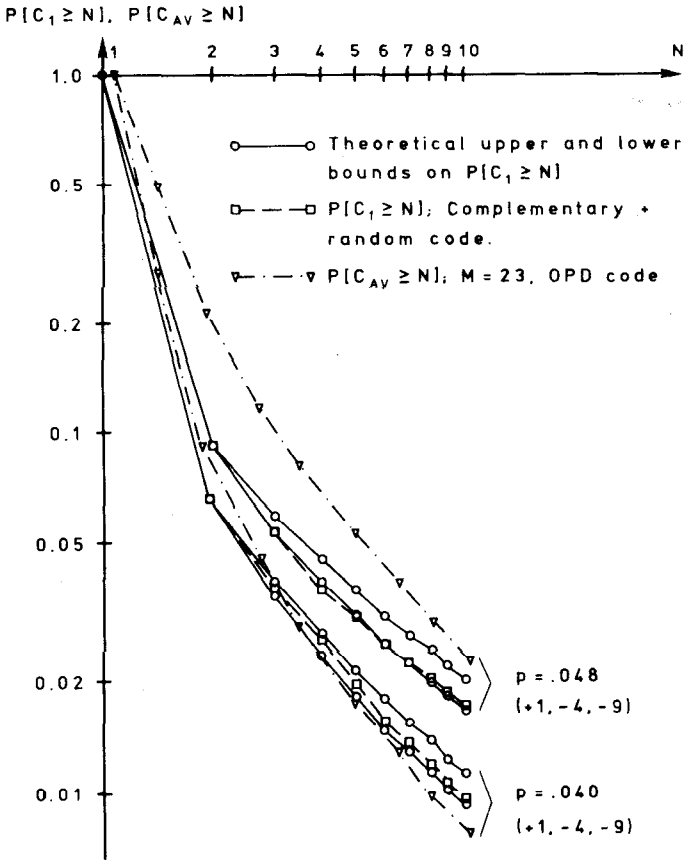

Fig. 6. The computational distribution functions $P[C_1 \geq N]$ and $P[C_{AV} \geq N]$ obtained from sequential decoding simulations for the $M=23$ ODP convolutional code, and theoretical upper and lower bounds on $P[C_1 \geq N]$.

Acknowledgment

I am deeply indebted to Professor James L. Massey for innumerable stimulating and encouraging discussions on the research and the results presented in this paper. His constant encouragement can hardly be overestimated. I also wish to thank one of the anonymous referees for drawing my attention to references [17] and [18]. Finally, I am indebted to the American-Scandinavian Foundation and to the Swedish Telephone Company L. M. Ericsson for their support.

References

[1] I. M. Jacobs and E. R. Berlekamp, "A lower bound to the distribution of computation for sequential decoding," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 167–174, Apr. 1967.

[2] J. E. Savage, "The distribution of sequential decoding computation time," *IEEE Trans. Inform. Theory*, vol. IT-12, pp. 145–147, Apr. 1966.

[3] F. Jelinek, "An upper bound on moments of sequential decoding effort," *IEEE Trans. Inform. Theory*, vol. IT-15, pp. 140–149, Jan. 1969.

[4] R. G. Gallager, *Information Theory and Reliable Communication*. New York: Wiley, 1968.

[5] J. M. Geist, "An empirical comparison of two sequential decoding algorithms," *IEEE Trans. Commun. Tech.*, vol. COM-19, pp. 415–419, Aug. 1971.

[6] F. Jelinek, "A fast sequential decoding algorithm using a stack," *IBM J. Res. Dev.*, vol. 13, pp. 675–685, Nov. 1969.

[7] T. E. Harris, *The Theory of Branching Processes*. Berlin: Springer-Verlag, 1963.

[8] C. J. Mode, *Multitype Branching Processes*. New York: American Elsevier, 1971.

[9] K. B. Athreya and P. E. Ney, *Branching Processes*. New York: Springer-Verlag, 1972.

[10] R. Johannesson, "Robustly optimal rate one-half binary convolutional codes," *IEEE Trans. Inform. Theory*, vol. IT-21, pp. 464–468, July 1975.

[11] J. L. Massey, "Variable-length codes and the Fano metric," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 196–198, Jan. 1972.

[12] R. M. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. Inform. Theory*, vol. IT-9, pp. 64–74, Apr. 1963.

[13] K. Sh. Zigangirov, "Some sequential decoding procedures," *Probl. Peredachi Inform.*, vol. 2, pp. 13–25, 1966.

[14] J. L. Massey *et al.*, "Certain infinite Markov chains and sequential decoding," *Discrete Math.*, vol. 3, pp. 163–175, Sept. 1972.

[15] J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering*. New York: Wiley, 1965.

[16] I. Pohl, "First results on the effect of error in heuristic search" in *Machine Intelligence*, vol. 5, Meltzer and Michie, Eds. New York: American Elsevier, 1970, p. 235.

[17] N. D. Vvdenskaja and K. Sh. Zigangirov, "On the computation time distribution of the sequential decoding," *Probl. Peredachi Inform.*, vol. 5, pp. 78–80, 1969.

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