Lower and Upper Bounds on the Distribution of Computation for Sequential Decoding of Specific Linear Tree Codes. *

Vladimir B. Balakirsky , Johan Nyström †
The authors are with the Department of Information Theory, University of Lund, P.O. Box 118, S-221 00 Lund, Sweden.

Abstract. Lower and upper bounds on the distribution of computation in the first incorrect subtree, are examined for specific tree codes. These bounds coincide for the initial part of the distribution both for a particular version of the stack algorithm and for idealized models. The computational performance of the investigated version is close to that of the optimistic model.

The distribution of computation in the first incorrect subtree (conditioned on correct decoding), $Pr\{\ C \geq c\ \}$, is one of the main characteristics of a sequential decoder for memoryless channels. It is known that this distribution is asymptotically Paretean for random codes [1,2]. The difficulties that arise during generalizations of the approaches for random codes to specific codes are connected to the fact that *collections* of paths in the first incorrect subtree should be examined [3-5]. Our considerations are based on the observation that the computational efforts of a sequential decoder may be represented as the sum of two components : the computations in the initial part of the tree, say up to some depth j, and the computations in the rest of the tree. If we first fix j subblocks of the received sequence,

*This work is partially supported by a Scholarship from the Swedish Institute, Stockholm, Sweden.

†Currently at Ericsson Radio Systems AB, Stockholm.

we can explicitely find the metrics of all nodes of the first incorrect subtree up to the j-th level. The total number of computations of the decoder may be estimated based on these metrics. The final result will be obtained after summarizing on all possible subblocks of the received sequence. Lower and upper estimates on the distribution of computation may be obtained in this way for all $c \leq 2^j$, and for $c \leq j$ these estimates coincide. It turns out that the difference between the upper and lower estimates is "rather small" for c up to some c_{max}, where we suppose that c_{max} grows exponentially with j. This point will be illustrated using numerical examples.

We restrict our considerations to the BSC, rate $R = 1/n$ codes, and a specific version of the stack algorithm. Suppose that the all-zero code sequence is transmitted over a BSC with crossover probability $p = 1 - q < 1/2$. We may then write the probability to receive a sequence

$$\mathbf{y} = (y_1, y_2, \dots), \quad y_t \in \{0,1\}^n \ \text{ for all } \ t = 1, 2, \dots$$

as follows:

$$P_0(\mathbf{y}) \quad = \quad \prod_{t \geq 1} p^{w_H(y_t)} \cdot q^{n - w_H(y_t)}, \ \text{ for all } \ y_t \in \{0,1\}^n,$$

where w_H denotes the Hamming weight of a binary vector.

The stack decoder contains a list of paths in the code tree, selects the path with the largest metric which we denote m^*, and extends it [3]. The occurrence of several paths with the metric m^* generates *a tie.* Different ways of resolving ties give different versions of the stack algorithm. Each path extension is called a *computation.* We consider a particular version of the stack algorithm and compare results to the idealized models. The description is given below.

Longest-Path version: If there are several paths with metric m^* then select the longest one. If there are several longest paths, then one of them is selected in accordance with the uniform distribution.

Pessimistic model: If there is an incorrect path with metric m^*, then it is selected.

Optimistic model: If the correct path has metric m^*, then it is selected.

We introduce the following representation of the distribution of computation:

$$Pr\{\ C \geq c\ \} = \sum_{\mathbf{y}_j \in \{0,1\}^{jn}} P_0(\mathbf{y}_j) \cdot Pr\{\ C \geq c \ \mid \ \mathbf{y}_j\ \},$$

where j is given, and use the following notation [6]. Let π_d be the probability that the dip of the metric along the correct path is not less than d, let $d_c(\mathbf{y}_j)$ be the minimal dip required to make at least c computations in the nodes of the first incorrect subtree, and let $d(\mathbf{y}_j)$ be the minimal dip necessary to make a computation in a node of the first incorrect subtree located at level j, when $\mathbf{y}_j$ is given. Let us note that the values of $d_c(\mathbf{y}_j)$ and $d(\mathbf{y}_j)$ depend on the particular version of resolution of ties.

Proposition. For all $c = 1, ..., c_{max}$, the conditional distribution of computation satisfies the following relations:

$$Pr\{\ C \geq c \ \mid \ \mathbf{y}_j\ \} = \pi_{d_c(\mathbf{y}_j)}, \ \text{ if } \ d_c(\mathbf{y}_j) < d(\mathbf{y}_j)$$

and

$$\pi_{d_c(\mathbf{y}_j)} \leq Pr\{\ C \geq c \ \mid \ \mathbf{y}_j\ \} \leq \pi_{d(\mathbf{y}_j)}, \ \text{ if } \ d_c(\mathbf{y}_j) \geq d(\mathbf{y}_j),$$

where the numbers $d_c(\mathbf{y}_j)$ and $d(\mathbf{y}_j)$ are defined above.

In Table 1, the distribution of computation is given for a specific tree code when the initial part of the generator polynomials is fixed.

Investigations of specific tree codes are usually based on some model for the code tree(e.g., complementary + random codes [4]). This makes it possible to apply analytical approaches to estimate the distribution of computation, but requires some justification of the model of the first incorrect subtree. Our scheme combines both calculations of metrics in the initial part of the first incorrect subtree and estimations of these metrics in the rest of the tree by analytical methods. These methods are based only on the correct path behaviour and do not use any model for the first incorrect subtree.
The values obtained from the lower and upper bounds on the probability $Pr\{\ C \geq c\ \}$ provide a good approximation both for the real value and for its accuracy, contrary to results of simulation.
Any version of resolution of ties gives an algorithm whose performance lies between those of the pessimistic and optimistic models. Furthermore, comparison of the Longest-Path version with these models shows that it is close to the optimal one.

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

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

[4] R.Johannesson, "On the distribution of computation for sequential decoding using the stack algorithm," *IEEE Trans.Inform.Theory,* vol. IT-25, pp.323-331, May 1979.

[5] P.R.Chevillat and D.J.Costello, "An analysis of sequential decoding for specific time-invariant convolutional codes," *IEEE Trans.Inform.Theory,* vol. IT-24, pp.443-451, July 1978.

[6] J.L.Massey, M.K.Sain, and J.M.Geist, "Certain infinite Markov chains and sequential decoding," *Discrete Math.,* vol. 3, pp.163-175, Sept. 1972.

Table 1: Values of the lower and upper bound on $Pr\{ \quad C \geq c \quad \}$, for the Optimistic model (lower bound), Longest-Path version (lower and upper bounds), and the Pessimistic model (upper bound) when $j = 9$ and the initial part of the generator polynomials is given by $G = (612\ , 567)$, in octal notation. The values $+1, -4, -9$ were used as metrics for zero, one, and two disagreements between the code and received subblocks. The crossover probability $p = 0.045$ corresponds to the computational cut-off rate 1/2.

c	$Optimistic$ $model$	$Longest - Path$ $version$		$Pessimistic$ $model$
1	5.604e-02	8.195e-02	8.195e-02	1.148e-01
2	2.752e-02	2.752e-02	2.752e-02	3.301e-02
3	2.566e-02	2.569e-02	2.569e-02	3.061e-02
4	2.478e-02	2.487e-02	2.487e-02	2.952e-02
5	2.008e-02	2.034e-02	2.034e-02	2.350e-02
6	1.606e-02	1.639e-02	1.639e-02	1.919e-02
7	1.313e-02	1.346e-02	1.346e-02	1.599e-02
8	1.247e-02	1.269e-02	1.269e-02	1.492e-02
9	1.186e-02	1.191e-02	1.191e-02	1.391e-02
10	1.087e-02	1.106e-02	1.107e-02	1.298e-02
11	8.510e-03	1.006e-02	1.033e-02	1.206e-02
12	6.814e-03	8.434e-03	1.024e-02	1.191e-02
20	3.650e-03	3.835e-03	8.518e-03	9.952e-03
30	2.178e-03	2.287e-03	8.434e-03	9.841e-03
40	1.336e-03	1.389e-03	8.434e-03	9.841e-03