Correspondence

Path Map Symbol Release Rules and the Exponential Metric Tree

WEN-WHEI CHANG AND JERRY D. GIBSON

Abstract—**To design a tree coder for source coding with a fidelity criterion, one must choose a suitable code generator, an efficient tree search algorithm, an appropriate distortion measure, and a path map symbol release rule. The performance of several path map symbol release rules when used with exhaustive searching of the exponential metric tree is investigated. The average single-letter distortion of fixed length symbol release rules and two variable length symbol release rules are derived for shallow search depths and compared to simulation results. The incremental or single symbol release rule is shown to yield the best performance.**

I. Introduction

Tree encoding is known to be capable of performing arbitrarily close to the rate distortion bound for any memoryless source and single-letter fidelity criterion [1], [2]. It employs a multipath search that pursues some or all of the paths in the code tree and chooses among them the best path at some search depth L. Designs of such a source encoder usually involve choosing a suitable code generator, an efficient tree search algorithm, and an appropriate distortion measure. In addition, it is also important to specify a rule to release the path map symbols to the channel. This correspondence addresses the problem of path map symbol release rule performance for the exponential metric tree. We present theoretical analyses for exhaustive tree searching that yield the average single-letter distortion performance of fixed length symbol release rules and two variable length symbol release rules, and we substantiate the theoretical values with simulation results. The fixed symbol release rule results augment and check those of Bodie [3]. The organization of this correspondence is as follows. Section II gives a general description of tree coding and defines some terminology. Section III presents the theoretical analyses and simulation results of various path map symbol release rules, including incremental encoding, block encoding, and two variable symbol release rules.

II. Tree Coders

Tree coders employ encoding delay to provide a multipath search capability, and all possible output sequences are placed within a tree structure that consists of branches and nodes. There is a fixed number of N branches emanating from each node, each of which terminates in another node. In a code tree, each branch is labelled with β branch letters chosen from a reconstruction alphabet, and the encoding rate in bits per sample is defined as $(1/\beta)\log_2 N$. A path is a sequence of connected branches through the tree and can be uniquely located by the path map symbols that specify which of the N emanating branches is to be followed to the node at the next level. Instead of transmitting branch letters, tree coders transmit these path map symbols. To gain insight into path map symbol release rules, the metric tree is introduced for use in analyzing tree coder performance. It differs from the code tree only in that each branch is labelled with the metric increment that quantifies the distortion associated with the reconstruction value. The path metric is defined as the cumulative metric increment through the path.

A tree coder has four elements: a code generator, a tree search algorithm, a distortion measure, and a path map symbol release rule, as shown in Fig. 1. Given an L source letter sequence $s = s_1, s_2, \cdots, s_L$, the code generator maps all possible path maps to a branch letter sequence $\hat{s} = \hat{s}_1, \hat{s}_2, \cdots, \hat{s}_L$. Code generators can be classified as having a deterministically populated or stochastically populated code tree. The code trees associated with conventional source encoders such as pulse code modulation (PCM), differential PCM (DPCM), and adaptive DPCM (ADPCM), belong to the class of deterministic tree codes. They are more straightforward to implement, but they also provide the least gains over the single path source encoder. Stochastic tree coders use appropriately chosen random variables as the reconstruction samples to populate the tree, and asymptotically in L, provide better performance than the deterministic coders.

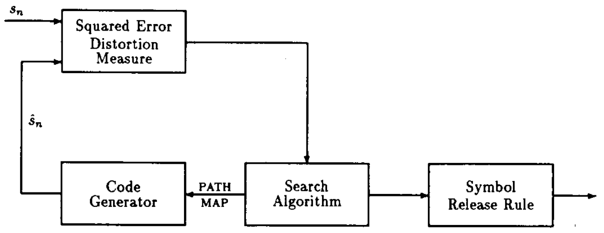

Fig. 1. Tree coder.

The distortion measure calculates the path metric $e(s, \hat{s})$ that quantifies how well the reconstruction for a given path map approximates the source letters. The most popular single-letter fidelity criterion is the mean-squared error. The tree search algorithm finds the path with the least path metric by sequentially feeding different path maps to the code generator and using the distortion measure to evaluate the reconstructed $\hat{s}$ until the best is found. Some popular tree search algorithms are the exhaustive search, the (M, L) algorithm [4], the stack algorithm [5], and the two-cycle algorithm [6]. The most effective is the exhaustive search algorithm, which sets the upper bound on the performance for all other search algorithms. It searches all possible branches of the code tree to depth L, and hence the best $\hat{s}$ is always found. However, as the search depth L is

Manuscript received September 28, 1988; revised September 5, 1989. This work was presented in part at the 1988 International Conference on Advances in Communication and Control Systems, Baton Rouge, LA, October 19–21, 1988.

W.-W. Chang is with the Department of Communication Engineering, National Chiao-Tung University, Taiwan, ROC.
J. D. Gibson is with the Department of Electrical Engineering, Texas A&M University, College Station, TX 77843.

IEEE Log Number 9035999.

increased, the complexity of the exhaustive search algorithm grows exponentially, and one of the other algorithms may be preferable [7].

The path map symbol release rule specifies how the path map symbols are presented to the channel. Most previous research on source coding with a fidelity criterion has emphasized the selection of a suitable tree code generator and an efficient tree search algorithm for some chosen symbol release rule. The most popular path map symbol release rule is single-symbol release, sometimes called incremental encoding, where no matter how deep the tree is searched, only the first symbol in the best path is released at any time instant [5]. Following Gray's suggestion [8] that undesirable path switching might occur with incremental encoding, Goris and Gibson [9] briefly examined some variable symbol release rules. However, the question as to which symbol release rule yields the best performance remains open. In subsequent sections, we compare several path map symbol release rules for exhaustive searching of the exponential metric tree.

III. Path Map Symbol Release Rules

A theoretical analysis of path map symbol release rule performance is complicated by the fact that even an elementary memoryless data source and a simple code generator have a complex metric tree structure. To permit a theoretical analysis, we apply various path map symbol release rules to a binary metric tree in which the per-level metric increase μ is distributed as the exponential density. As shown in Appendix A, this exponential metric tree corresponds to the actual situation of applying binary, rate-1/2 tree encoding to a memoryless Gaussian source. In this paper, we study various symbol release rules on a binary L-depth metric tree in which the increase in path metric per level is distributed as the exponential density e^{-x}, $x \geq 0$, also studied by Bodie [3]. We define λ_k as the least path metric attainable with an exhaustive search from level k to level L. Some density functions of λ_k are derived in Appendix B.

A. Fixed Symbol Release Rule

First, we study the fixed symbol release rule in which no matter how deep the tree is searched, a fixed number of path map symbols is released after each search. Both incremental and block tree encoding, which release 1 and L symbols, respectively, are special cases of this rule. Some theoretical analyses of the fixed symbol release rule performance are presented in Appendix C. The theoretical values shown in Table I are for the L-depth exponential metric tree using the exhaustive tree search. The parameter D_j is the average per-letter distortion attainable

TABLE I
Average Per-Letter Distortion for Fixed Symbol Release*

L	D_1	D_2	D_3	D_4	D_5
1	0.5000				
2	0.4167	0.4584			
3	0.3734	0.3951	0.4300		
4	0.3462	0.3598	0.3788	0.4091	
5	0.3275	0.3369	0.3490	0.3660	0.3928

*Theoretical results.

with j number of path map symbols released. The leftmost column ($j=1$) and rightmost column ($j=L$) correspond to incremental and block encoding, respectively. Note that none of the multiple symbol release rules outperforms the single symbol release rule. Simulation results for blocklengths 1–12 are shown in Table II. The one standard deviation confidence interval for the simulation values in Table II range from ± 0.001 to ± 0.003. Comparing Tables I and II, it is evident that the theoretical and experimental values of average distortion agree very closely.

The values of D_j for a given L in Tables I and II also indicate that performance degrades as the number of symbols released in a fixed length symbol release rule is increased. The reason for this seems to be that some path switching, which occurs whenever the coder finds a better path than the one it is pursuing, is desirable, even though there is a risk of not staying on a path long enough to achieve its average performance [8]. The more symbols released in a fixed symbol release rule, the fewer opportunities the coder has to switch to a better path [9]. Another reason for degraded performance by the fixed L-symbol release rule is that there may be a problem with locally large distortions near the block edges due to reinitialization [8].

B. Variable Symbol Release Rules

As noted by Gray [8], a variable symbol release rule should be employed to stay on a good path long enough to achieve the promised long-term fidelity. The logic behind this approach is that the first step of the good path, which has the least path metric, may be a poor one with a large sample distortion. Two variable symbol release rules are investigated here. One rule releases the path map symbols on the best path until the running average distortion is less than or equal to the long-term average distortion, while the other releases symbols until the level that has the least running average distortion on the best path. Goris and Gibson [9] suggest that the maximum number of symbols released should be constrained to $J = \lfloor L/2 \rfloor$, the largest

TABLE II
Average Per-Letter Distortion for Fixed Symbol Release*

L	1	2	3	4	5	6	7	8	9	10	11	12
D_1	0.5014	0.4229	0.3799	0.3544	0.3312	0.3232	0.3100	0.2987	0.3030	0.2876	0.2913	0.2857
D_2		0.4659	0.3950	0.3653	0.3416	0.3267	0.3183	0.2964	0.2998	0.2951	0.2915	0.2857
D_3			0.4325	0.3855	0.3557	0.3363	0.3206	0.3091	0.3049	0.2987	0.2940	0.2868
D_4				0.4147	0.3727	0.3454	0.3237	0.3141	0.3070	0.2969	0.2954	0.2908
D_5					0.4049	0.3571	0.3365	0.3223	0.3099	0.3048	0.3014	0.2901
D_6						0.3838	0.3478	0.3298	0.3172	0.3137	0.3009	0.2945
D_7							0.3638	0.3446	0.3227	0.3119	0.3069	0.2952
D_8								0.3598	0.3330	0.3188	0.3117	0.3027
D_9									0.3567	0.3282	0.3173	0.3029
D_{10}										0.3470	0.3213	0.3109
D_{11}											0.3372	0.3158
D_{12}												0.3342

*Simulation results.

integer less than $L/2$, and we adopt this rule here. The following procedures are used.

1) Exhaustively search for the best L-depth path with the minimum path metric, calculate the long-term average distortion E_{mmse} and the running average distortion at the jth level E_j, $j=1,2,\cdots,J$,

$$E_{\text{mmse}} = \frac{1}{L}\sum_{i=1}^{L}(s_i - \hat{s}_i)^2$$

and

$$E_j = \frac{1}{j}\sum_{i=1}^{j}(s_i - \hat{s}_i)^2.$$

2) Select one variable symbol release rule.

Rule 1: Release the path map symbols until the ith level whose running average distortion E_i is less than or equal to E_{mmse} [9].

Rule 2: Search for the minimum running average distortion on the best path, say E_i at level i. If E_i is less than E_{mmse}, then release i path map symbols. If E_i is greater than E_{mmse}, then release only one symbol.

3) If the number of path map symbols released in Step 2) reaches the constraint J, then stop sending.

We have analyzed both of the variable symbol release rules for blocklengths 4 and 5 in Appendix D. As shown in Table III, neither of the variable symbol release rules is able to outperform the single symbol release rule, although Rule 2 comes very close. This implies that the first step on the best path has a good sample distortion. Table IV shows the simulation results for search depths 4–12, where the one standard deviation confidence interval should be considered to be ± 0.001 to ± 0.003. Comparing with Table III, we see that the theoretical and simulation results are in good agreement.

TABLE III
AVERAGE PER-LETTER DISTORTION FOR VARIABLE SYMBOL RELEASE*

L	Incremental	Block	Rule 1	Rule 2
4	0.3462	0.4090	0.3525	0.3478
5	0.3275	0.3928	0.3325	0.3285

*Theoretical results.

TABLE IV
AVERAGE PER-LETTER DISTORTION FOR VARIABLE SYMBOL RELEASE*

L	Incremental	Block	Rule 1	Rule 2
4	0.3544	0.4147	0.3628	0.3575
5	0.3312	0.4049	0.3396	0.3353
6	0.3232	0.3838	0.3282	0.3302
7	0.3100	0.3638	0.3141	0.3110
8	0.2987	0.3598	0.3063	0.3033
9	0.3030	0.3567	0.3039	0.3031
10	0.2876	0.3470	0.2967	0.2964
11	0.2913	0.3372	0.2947	0.2925
12	0.2857	0.3342	0.2907	0.2870

*Simulation results.

IV. Conclusion

We have shown that the single symbol release rule outperforms other fixed symbol release rules and two variable symbol release rules with exhaustive search of the exponential metric tree. This work thus tends to reinforce the simulation results in [9] and [11] for speech sources. Fixed multiple symbol release rule performance is degraded with an increase in the number of released symbols by the fact that it may miss desirable path switching and may have the problem of locally large distortion near the edges of a block. Variable symbol release rules fail to outperform the single symbol release rule, but their performance difference is small. Hence, one possible use of the variable symbol release rule is in the reduction of computational load when selecting an exhaustive search [9].

Appendix A
The Exponential Metric Tree

To simplify the theoretical analyses, we have assumed a binary metric tree in which the metric increase μ per level is distributed according to the exponential density. There exists a practical situation where tree encoding an actual data source will generate the exponential metric tree. This occurs when we apply tree encoding to a memoryless Gaussian source, whose source letters are distributed independently with normal density $N(0,\frac{1}{2})$, and use a rate-1/2 code tree structure with two branches emanating from each node, each branch associated with two branch letters that are assumed to be chosen from the same $N(0,\frac{1}{2})$ as the source letters. We denote the two reconstruction letters on one branch as $\hat{S}_1$ and $\hat{S}_2$, corresponding to the source letters S_1 and S_2, and denote X_i as the difference between S_i and $\hat{S}_i$. We also define the metric increase per level $\mu = X_1^2 + X_2^2$.

Since both source letters and branch letters have the identical independent density function $N(0,\frac{1}{2})$, their difference X_i has the density $N(0,1)$. It is known that if $X_1,\cdots,X_n$ is a random sample from a normal distribution with mean ξ and variance σ^2, then $U=\Sigma_{i=1}^{n}(X_i-\xi)^2/\sigma^2$ has a chi-square distribution with n degrees of freedom [10]. Hence the metric increase per level μ will be distributed as $\mathscr{X}_2^2$, a chi-square density with two degrees of freedom. This is also the exponential density with parameter $\frac{1}{2}$,

$$f_u(\mu) = \begin{cases} \frac{1}{2}e^{-\mu/2}, & \mu \ge 0, \\ 0, & \mu < 0. \end{cases}$$

Appendix B
Exhaustive Search on the Exponential Metric Tree

We assume a binary depth L metric tree in which the increase in path metric per level (μ, the metric increase) is distributed as the exponential density and follow the development of Bodie [3]. We define λ_k as the least path metric attainable with an exhaustive search from level k to level L. Extending the search one level further back involves choosing for each node the branch that contributes to the smaller path metric from level $k-1$ to level L. These path metrics are the sum of two independent components λ_k and μ, and their density functions equal the convolution of their respective component density functions

$$f_{\lambda_k'}(x) = f_{\lambda_k}(x) * f_\mu(x).$$

Then λ_{k-1}, the least path metric from level $k-1$ to level L, has the density of the minimum of two random variables that are distributed as $f_{\lambda'_k}(x)$,

$$f_{\lambda_{k-1}}(x) = 2f_{\lambda'_k}(x)\left[1 - F_{\lambda'_k}(x)\right].$$

Since we always have $\lambda_L = 0$, $f_{\lambda_L} = \delta(x)$ at the last level of the tree. Starting from the last level, we may iterate the previous two equations to obtain the density function of λ_k at any level. Thus

$$\begin{aligned}
f_{\lambda_L}(x) &= \delta(x);\\
f_{\lambda_{L-1}}(x) &= 2e^{-2x};\\
f_{\lambda_{L-2}}(x) &= 8e^{-2x} - 12e^{-3x} + 4e^{-4x};\\
f_{\lambda_{L-3}}(x) &= 22.2e^{-2x} - 80e^{-3x} + 117.3e^{-4x} - 91.1e^{-5x}\\
&\qquad + 40e^{-6x} - 9.3e^{-7x} + 0.9e^{-8x};\\
f_{\lambda_{L-4}}(x) &= 52.6e^{-2x} - 341.8e^{-3x} + 1040.7e^{-4x}\\
&\quad - 1982.8e^{-5x} + 2650.6e^{-6x}\\
&\quad - 2629.5e^{-7x} + 1992e^{-8x} - 1167.7e^{-9x}\\
&\quad + 531.1e^{-10x} - 186.1e^{-11x}\\
&\quad + 49.4e^{-12x} - 9.6e^{-13x} + 1.3e^{-14x} - 0.1e^{-15x};\\
f_{\lambda_{L-5}}(x) &= 113.5e^{-2x} - 1188.1e^{-3x} + 6197e^{-4x}\\
&\quad - 21507.5e^{-5x} + 55789.2e^{-6x}\\
&\quad - 114968.3e^{-7x} + 195238.6e^{-8x}\\
&\quad - 279807.2e^{-9x} + 344013e^{-10x}\\
&\quad - 367097e^{-11x} + 342883.2e^{-12x}\\
&\quad - 282050.3e^{-13x} + 205210.4e^{-14x}\\
&\quad - 132438e^{-15x} + 75941.9e^{-16x}\\
&\quad - 38712.1e^{-17x} + 17534.9e^{-18x}\\
&\quad - 7047.2e^{-19x} + 2506.6e^{-20x}\\
&\quad - 786.2e^{-21x} + 216.4e^{-22x}\\
&\quad - 51.9e^{-23x} + 10.8e^{-24x} - 19e^{-25x}\\
&\quad + 0.3e^{-26x}.
\end{aligned}$$

Appendix C
Fixed Symbol Release Rule Performance

We denote $\bar{\lambda}_k$ as the expected value of λ_k, the least path metric attainable with an exhaustive search from level k to level L. From the density f_{λ_k} given in Appendix B, we can calculate both $\bar{\lambda}_k$ and D_j. See Tables V and I, respectively,

$\bar{\lambda}_k$: expected value of the path metric from level k to level L.
D_j: the least per-letter average distortion attainable with the exhaustive search and j fixed-symbol release rule.

$$\bar{\lambda}_k = E(\lambda_k) = \int_{-\infty}^{+\infty} x f_{\lambda_k}(x)\,dx$$

$$D_j = (\bar{\lambda}_0 - \bar{\lambda}_j)/j.$$

TABLE V
Expected Value of Path Metric

L	$\bar{\lambda}_0$	$\bar{\lambda}_1$	$\bar{\lambda}_2$	$\bar{\lambda}_3$	$\bar{\lambda}_4$	$\bar{\lambda}_5$
1	0.5000	0.0000				
2	0.9167	0.5000	0.0000			
3	1.2901	0.9167	0.5000	0.0000		
4	1.6363	1.2901	1.9167	0.5000	0.0000	
5	1.9638	1.6363	1.2901	0.9167	0.5000	0.0000

Appendix D
Variable Symbol Release Rule Performance

A variable symbol release rule is employed to stay on a good path long enough to achieve the promised long-term fidelity. Two variable symbol release rules are investigated here.

Rule 1: Release the path map symbols until the running average distortion is less than or equal to the long-term average distortion, or reaches the release constraint $\lfloor L/2 \rfloor$.

Rule 2: Release the path map symbols until the level that has the smallest running average distortion, or the release constraint $\lfloor L/2 \rfloor$ is reached. Only release one symbol when the smallest running average distortion is greater than the long-term average distortion.

A. Depth-4 Exponential Metric Tree Calculations

1) Performance of Rule 1: We denote R_i as the event that i path map symbols are decided to be released and with $u_1 = 3\lambda_0$, $u_2 = 4\lambda_1$, and $y = u_1 - u_2$, we have

$$\begin{aligned}
P(R_1) &= P\left[(\lambda_0 - \lambda_1) \le \frac{\lambda_0}{4}\right]\\
&= P[3\lambda_0 - 4\lambda_1 \le 0] = P[(u_1 - u_2) \le 0]\\
&= \int_{-\infty}^{0} \int_{-3y}^{+\infty} f_{u_1,u_2}(u_1, u_1 - y)\, du_1\, dy = 0.535347
\end{aligned}$$

where

$$\begin{aligned}
f_{u_1,u_2} &= f_{u_1|u_2} \cdot f_{u_2}\\
&= e^{-2u_1/3}[3.7 - 13.3e^{-u_2/4} + 19.6e^{-u_2/2} - 15.2e^{-3u_2/4}\\
&\quad + 6.7e^{-u_2} - 1.6e^{-5u_2/4} + 0.15e^{-3u_2/2}],\\
&\qquad\qquad u_1 \ge 0, \quad u_2 \ge 0, \quad u_1 \ge \tfrac{3}{4}u_2.
\end{aligned}$$

The least average path metric attainable with Rule 1 is

$$\begin{aligned}
D &= D_1 \cdot P(R_1) + D_2 \cdot P(R_2)\\
&= 0.3462 \times 0.535347 + 0.3598 \times 0.464653 = 0.3525.
\end{aligned}$$

2) Performance of Rule 2:

$$\begin{aligned}
P(R_2) &= P\left[\frac{\lambda_0 - \lambda_2}{2} \le \lambda_0 - \lambda_1 \quad \text{and} \quad \frac{\lambda_0 - \lambda_2}{2} \le \frac{\lambda_0}{4}\right]\\
&= P[(\lambda_0 - 2\lambda_1 + \lambda_2) \ge 0 \quad \text{and} \quad -\lambda_0 + 2\lambda_2 \ge 0]\\
&= \int\int\int f_{\lambda_0,\lambda_1,\lambda_2}(x,y,z)\, dz\, dy\, dx
\end{aligned}$$

where

$$\begin{aligned}
f_{\lambda_0,\lambda_1,\lambda_2} &= f_{\lambda_0|\lambda_1,\lambda_2} \cdot f_{\lambda_1|\lambda_2} \cdot f_{\lambda_2}\\
&= [2e^{-2(\lambda_0 - \lambda_1)}] \cdot [2e^{-2(\lambda_1 - \lambda_2)}]\\
&\quad \cdot [8e^{-2\lambda_2} - 12e^{-3\lambda_2} + 4e^{-4\lambda_2}]\\
&= [32e^{-2\lambda_0} - 48e^{-2\lambda_0 - \lambda_2} + 16e^{-2\lambda_0 - 2\lambda_2}],\\
&\qquad\qquad \lambda_0 \ge \lambda_1 \ge \lambda_2 \ge 0.
\end{aligned}$$

The upper and lower limits in the integral must be chosen to satisfy the following three constraints:

a) $x \ge y \ge z \ge 0$;

b) $-x + 2z \ge 0$ or $z \ge x/2$;

c) $x - 2y + z \ge 0$ or $z \ge (2y - x)$.

There exist two possible conditions:

$$\text{Event A) } z \geq (2y-x) \geq \frac{x}{2} \quad \text{and} \quad x \geq y \geq z \geq 0;$$

$$\text{Event B) } z \geq \frac{x}{2} \geq (2y-x) \quad \text{and} \quad x \geq y \geq z \geq 0.$$

Hence

$$P(\text{Event A}) = P\left[(2\lambda_1 - \lambda_0 \geq)\frac{\lambda_0}{2}\right] = P[(3\lambda_0 - 4\lambda_1) \leq 0] = 0.535347,$$

$$P(\text{Event B}) = 1 - P(\text{Event A}) = 0.464653, \text{ and}$$

$$\begin{aligned} P(R_2) &= P(R_2|\text{Event A}) \cdot P(\text{Event A}) + P(R_2|\text{Event B}) \cdot P(\text{Event B}) \\ &= \left[\int_0^{+\infty}\int_{3x/4}^{x}\int_{2y-x}^{y} f_{\lambda_0,\lambda_1,\lambda_2}(x,y,z)\,dz\,dy\,dx\right] \cdot \text{Prob}\left[(2y-x) \geq \frac{x}{2}\right] \\ &\quad + \left[\int_0^{+\infty}\int_{x/2}^{3x/4}\int_{x/2}^{y} f_{\lambda_0,\lambda_1,\lambda_2}(x,y,z)\,dz\,dy\,dx\right] \cdot \text{Prob}\left[\frac{x}{2} \geq (2y-x)\right] \\ &= 0.1283 \times 0.535347 + 0.1075 \times 0.464653 = 0.118635. \end{aligned}$$

The least average path metric attainable with Rule 2 is

$$D = D_1 \cdot P(R_1) + D_2 \cdot P(R_2) = 0.3462 \times 0.8811365 + 0.3598 \times 0.118635 = 0.3478.$$

B. Depth-5 Exponential Metric Tree Calculations

1) Performance of Rule 1: We set $u_1 = 4\lambda_0$, $u_2 = 5\lambda_1$, and $y = u_1 - u_2$, so we can write

$$\begin{aligned} P(R_1) &= P\left[(\lambda_0 - \lambda_1) \leq \frac{\lambda_0}{5}\right] \\ &= P[4\lambda_0 - 5\lambda_1 \leq 0] = P[(u_1 - u_2) \leq 0] \\ &= \int_{-\infty}^{0}\int_{-4y}^{+\infty} f_{u_1,u_2}(u_1, u_1 - y)\,du_1\,dy = 0.46916 \end{aligned}$$

where

$$\begin{aligned} f_{u_1,u_2} &= f_{u_1|u_2} \cdot f_{u_2} \\ &= e^{-u_1/2} \cdot [5.3 - 34.2e^{-u_2/5} + 104e^{-2u_2/5} - 198.3e^{-3u_2/5} \\ &\quad + 265e^{-4u_2/5} - 263e^{-u_2} + 199.2e^{-6u_2/5} - 116.8e^{-7u_2/5} \\ &\quad + 53e^{-8u_2/5} - 18.5e^{-9u_2/5} + 5e^{-2u_2} - 1.0e^{-11u_2/5} \\ &\quad + 0.13e^{-12u_2/5} - 0.01e^{-13u_2/5}], \\ &\qquad u_1 \geq 0,\ u_2 \geq 0,\ u_1 \geq \frac{4}{5}u_2. \end{aligned}$$

The least average path metric attainable with Rule 1 is

$$D = D_1 \cdot P(R_1) + D_2 \cdot P(R_2) = 0.3275 \times 0.46916 + 0.3369 \times 0.53084 = 0.3325.$$

2) Performance of Rule 2:

$$\begin{aligned} P(R_2) &= P\left[\frac{\lambda_0 - \lambda_2}{2} \leq \lambda_0 - \lambda_1 \quad \text{and} \quad \frac{\lambda_0 - \lambda_2}{2} \leq \frac{\lambda_0}{5}\right] \\ &= P[(\lambda_0 - 2\lambda_1 + \lambda_2) \geq 0 \quad \text{and} \quad (-3\lambda_0 + 5\lambda_2 \geq 0)] \\ &= \int\int\int f_{\lambda_0,\lambda_1,\lambda_2}(x,y,z)\,dz\,dy\,dx, \end{aligned}$$

where

$$\begin{aligned} f_{\lambda_0,\lambda_1,\lambda_2} &= f_{\lambda_0|\lambda_1,\lambda_2} \cdot f_{\lambda_1|\lambda_2} \cdot f_{\lambda_2} \\ &= [2e^{-2(\lambda_0 - \lambda_1)}] \cdot [2e^{-2(\lambda_1 - \lambda_2)}] \\ &\quad \cdot [22.2e^{-2\lambda_2} - 80e^{-3\lambda_2} + 117.3e^{-4\lambda_2} \\ &\quad - 91.1e^{-5\lambda_2} + 40e^{-6\lambda_2} - 9.3e^{-7\lambda_2} + 0.9e^{-8\lambda_2}] \\ &= e^{-2\lambda_0} \cdot [88.8 - 320e^{-\lambda_2} + 469.3e^{-2\lambda_2} \\ &\quad - 364.4e^{-3\lambda_2} + 160e^{-4\lambda_2} \\ &\quad - 37.2e^{-5\lambda_2} + 3.6e^{-6\lambda_2}]. \end{aligned}$$

The upper and lower limits in the integral must be chosen to satisfy the following three constraints:

a) $x \geq y \geq z \geq 0$;

b) $-3x + 5z \geq 0$ or $z \geq 3x/5$;

c) $x - 2y + z \geq 0$ or $z \geq (2y - x)$.

There exist two possible conditions:

$$\text{Event A)} \quad z \geq (2y-x) \geq \frac{3x}{5} \quad \text{and} \quad x \geq y \geq z \geq 0;$$

$$\text{Event B)} \quad z \geq \frac{3x}{5} \geq (2y-x) \quad \text{and} \quad x \geq y \geq z \geq 0.$$

Hence

$$\begin{aligned} P(R_2) &= P(R_2|\text{Event A}) \cdot P(\text{Event A}) + P(R_2|\text{Event B}) \cdot P(\text{Event B}) \\ &= \left[\int_0^{+\infty}\int_{4x/5}^{x}\int_{2y-x}^{y} f_{\lambda_0,\lambda_1,\lambda_2}(x,y,z)\,dz\,dy\,dx\right] \cdot P\left[(2y-x) \geq \frac{3}{5}x\right] \\ &\quad + \left[\int_0^{+\infty}\int_{3x/5}^{4x/5}\int_{3x/5}^{y} f_{\lambda_0,\lambda_1,\lambda_2}(x,y,z)\,dz\,dy\,dx\right] \cdot P\left[(2y-x) \leq \frac{3}{5}x\right] \\ &= 0.1313 \times 0.46916 + 0.1023 \times 0.53084 = 0.1159. \end{aligned}$$

The least average path metric attainable with Rule 2 is

$$D = D_1 \cdot P(R_1) + D_2 \cdot P(R_2) = 0.3275 \times 0.8841 + 0.3369 \times 0.1159 = 0.3285.$$

References

[1] F. Jelinek, "Tree encoding of memoryless time-discrete sources with a fidelity criterion," *IEEE Trans. Inform. Theory*, vol. IT-15, pp. 584–590, Sept. 1969.

[2] C. R. Davis and M. E. Hellman, "On tree coding with a fidelity criterion," *IEEE Trans. Inform. Theory*, vol. IT-21, pp. 373–378, July 1975.

[3] J. B. Bodie, "Multi-path tree encoding for analog data sources," Commun. Res. Lab., McMaster Univ., Hamilton, ON, Canada, M. Eng. thesis, June 1974.

[4] J. B. Anderson and J. B. Bodie, "Tree encoding of speech," *IEEE Trans. Inform. Theory*, vol. IT-21, pp. 379–387, July 1975.

[5] J. B. Anderson, "A stack algorithm for source coding with a fidelity criterion," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 211–226, Mar. 1974.

[6] J. B. Anderson and F. Jelinek, "A 2-cycle algorithm for source coding with a fidelity criterion," *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 77–92, Jan. 1973.

[7] J. B. Anderson, "Recent advances in sequential encoding of analog waveforms," *Conf. Rec.*, 1978 Nat. Telecomm. Conf., Birmingham, AL, Dec. 3–6, pp. 19.4.1–19.4.5.

[8] R. M. Gray, "Time-invariant trellis encoding of ergodic discrete-time sources with a fidelity criterion," *IEEE Trans. Inform. Theory*, vol. IT-23, pp. 71–83, Jan. 1977.
[9] A. C. Goris and J. D. Gibson, "Incremental tree coding of speech," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 511–516, July 1981.
[10] A. M. Mood and F. A. Graybill, *Introduction to the Theory of Statistics*, second ed. New York: McGraw-Hill, 1963.
[11] S. Mohan, D. Kryskowski, and C.-M. Lin, "Stack algorithm speech encoding with fixed and variable symbol release rules," *IEEE Trans. Commun.*, vol. COM-33, pp. 1015–1018, Sept. 1985.

Bounds on the Undetected Error Probabilities of Linear Codes for Both Error Correction and Detection

MAO-CHAO LIN

Abstract—**The $(n,k,d\geq 2t+1)$ binary linear codes are studied, which are used for correcting error patterns of weight at most t and detecting other error patterns over a binary symmetric channel. In particular, for $t=1$, it is shown that there exists one code whose probability of undetected errors is upper bounded by $(n+1)[2^{n-k}-n]^{-1}$ when used on a binary symmetric channel with transition probability less than $2/n$.**

I. Introduction

In pure ARQ systems, linear codes are used solely for detecting errors. Suppose that we apply linear codes to a binary symmetric channel (BSC) with transition probability p. It [1, pp. 78–79] has been proved that for each p with $0\leq p\leq 1$, there exists an (n,k) binary linear code whose probability of undetected errors (PUDE) is upper bounded by $2^{-(n-k)}$. Hamming codes and double error correcting primitive BCH codes [2], [3] have been proved to satisfy the inequality if the transition probability p is no greater than $1/2$.

Pure ARQ systems have the problem of low throughput if the transition probability in the BSC is high. Therefore, in hybrid ARQ systems [1] especially in type-I hybrid ARQ systems, linear codes are used for correcting some low weight error patterns and detecting many other error patterns. Therefore, it is interesting to study the probability of undetected errors for linear codes that are used for both error correction and error detection over the BSC. In this correspondence, our study is divided into two parts. In the first part, we study the class of $(n,k,d\geq 3)$ systematic linear codes that can be used for correcting every single error and detecting other error patterns. We show that there exists one code whose PUDE is upper bounded by $(n+1)\cdot[2^{n-k}-n]^{-1}$ when the transition probability is less than $2/n$. In the second part, we study the (n,k) systematic linear codes that are used for correcting some low weight-error patterns and detecting other error patterns. Suppose that $1-R>H(2\lambda)$. We show that there exists an $(n,Rn,d\geq 2\lambda n+1)$ linear code whose PUDE is closely upper bounded by $2^{-[1-R-H(\lambda)]n}$ as n approaches infinity and the transition probability is less than λ (if it is used to correct all the error patterns of weight at most λn and to detect other error patterns).

Manuscript received February 8, 1989; revised December 1, 1989. This work was presented at the IEEE 1990 International Symposium on Information Theory, San Diego, CA, January 14–19, 1990. This work was supported by the National Science Council of the Republic of China under grant NSC 78-0404-E002-05.

M.-C. Lin is with the Department of Electrical Engineering, National Taiwan University, Taipei 10764, Taiwan, ROC.

IEEE Log Number 9036388.

II. Codes for Error Detection and Single-Error Correction

Consider the ensemble Γ of all systematic $(n,k,d\geq 3)$ binary linear codes. The generator matrix of an (n,k) systematic linear code V is of the form $G=[I\ \ P]$, where I is the $k\times k$ identity matrix and P is some $k(n-k)$ matrix. A necessary and sufficient condition for V to have minimum distance of at least 3 is that no two rows of P are identical and each row in P must have weight of at least 2. Therefore, the cardinality of Γ is

$$|\Gamma|=[2^{n-k}-1-(n-k)]\cdot[2^{n-k}-1-(n-k)-1]\cdots[2^{n-k}-1-(n-k)-(k-1)]=\frac{[2^{n-k}-1-(n-k)]!}{[2^{n-k}-1-n]!}.\tag{1}$$

We denote the codes in Γ by $V_1,V_2,\cdots,V_{|\Gamma|}$. Let $A_{i,w}$ be the number of weight-w codewords in V_i, where $i=1,2,\cdots,|\Gamma|$, and $w=0,3,4,\cdots,n$. Suppose V_i is used to correct every single error and detect other error patterns over a BSC with transition probability p, its PUDE is

$$P(E|V_i)=\sum_{w=2}^{n}[(w+1)\cdot A_{i,w+1}+A_{i,w}+(n-w+1)\cdot A_{i,w-1}]\cdot p^w(1-p)^{n-w}.\tag{2}$$

If the probability of choosing each code in Γ is equally likely, the average PUDE over all the codes in Γ is

$$P(E)=\frac{1}{|\Gamma|}\sum_{i=1}^{|\Gamma|}P(E|V_i)=\frac{1}{|\Gamma|}\sum_{w=2}^{n}\left\{\left[(w+1)\cdot\sum_{i=1}^{|\Gamma|}A_{i,w+1}\right]+\left[\sum_{i=1}^{|\Gamma|}A_{i,w}\right]+\left[(n-w+1)\cdot\sum_{i=1}^{|\Gamma|}A_{i,w-1}\right]\right\}\cdot p^w(1-p)^{n-w}.\tag{3}$$

Note that each nonzero n-tuple appears in at most $|\Gamma'|$ codes in Γ, where

$$|\Gamma'|\leq[2^{n-k}-1-(n-k)][2^{n-k}-1-(n-k)-1]\cdots[2^{n-k}-1-(n-k)-(k-2)]=\frac{[2^{n-k}-1-(n-k)]!}{[2^{n-k}-n]!}.\tag{4}$$

Thus, we have

$$\sum_{w=2}^{n}\left[(w+1)\sum_{i=1}^{|\Gamma|}A_{i,w+1}\right]\cdot p^w(1-p)^{n-w}\leq\sum_{w=2}^{n}(w+1)\binom{n}{w+1}\cdot|\Gamma'|\cdot p^w(1-p)^{n-w}=|\Gamma'|\cdot n(1-p)\cdot\sum_{w=2}^{n}\binom{n-1}{w}\cdot p^w(1-p)^{n-1-w}\leq|\Gamma'|\cdot n(1-p)\tag{5}$$