14p

DRA

[redacted]

19

Department of

ELECTRICAL ENGINEERING

UNIVERSITY OF NOTRE DAME, NOTRE DAME, INDIANA

REAL-TIME MINIMAL-BIT-ERROR PROBABILITY DECODING OF CONVOLUTIONAL CODES*

Lin-nan Lee
Dept. of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

Technical Report No. EE-7312

July 25, 1973

* This work was supported by the National Aeronautics and Space Administration under NASA Grant NGL 15-004-026 at the University of Notre Dame in liaison with the Communications and Navigation Division of the Goddard Space Flight Center and will be part of a dissertation to be submitted to the University of Notre Dame in partial fulfillment of the requirements for the Ph.D. degree.

Real-Time Minimal-Bit-Error Probability Decoding of Convolutional Codes*

by

Lin-nan Lee
Department of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

ABSTRACT

A recursive procedure is derived for decoding of rate R=1/n binary convolutional codes which minimizes the probability of the individual decoding decisions for each information bit subject to the constraint that the decoding delay be limited to Δ branches. This new decoding algorithm is similar to, but somewhat more complex than, the Viterbi decoding algorithm. A "real-time," i.e. fixed decoding delay, version of the Viterbi algorithm is also developed and used for comparison to the new algorithm on simulated channels. It is shown that the new algorithm offers advantages over Viterbi decoding in soft-decision applications such as in the inner coding system for concatenated coding.

* This work was supported by the National Aeronautics and Space Administration under NASA Grant NGL 15-004-026 at the University of Notre Dame in liaison with the Communications and Navigation Division of the Goddard Space Flight Center and forms part of a dissertation to be submitted to the University of Notre Dame in partial fulfillment of the requirements for the Ph.D. degree.

//

I. Introduction

A binary convolutional code of rate R=1/n and memory order m can be encoded using a shift-register of m stages together with n modulo-two adders as shown in Figure 1. In this diagram a_t is the

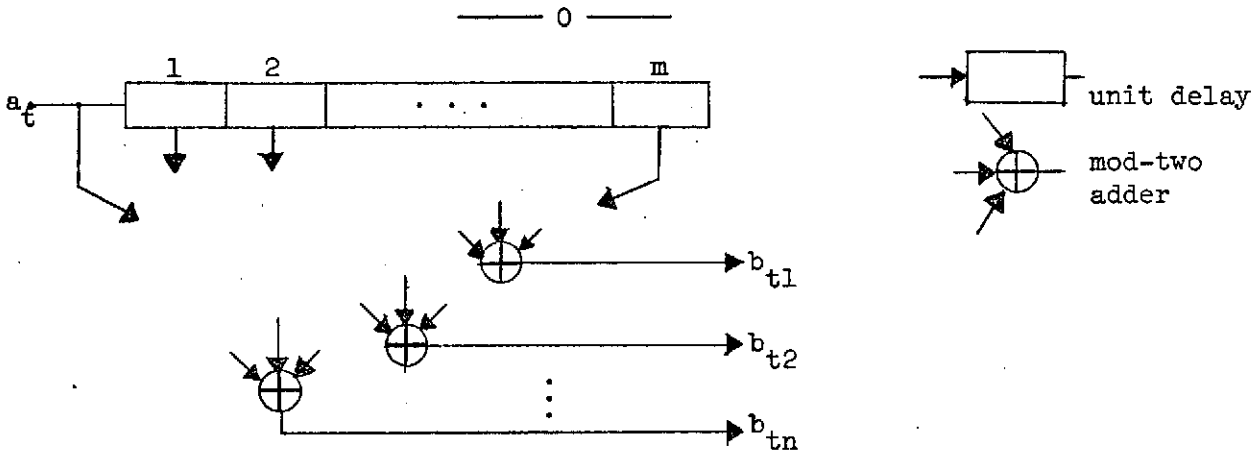

Fig. 1 An Encoder for a Rate R=1/n Convolutional Code of Memory Order m

0

input information digit at time t and

$$\underline{b}_t = [b_{t1}, b_{t2}, \ldots b_{tn}]$$

is the encoded branch at time t consisting of the n digits formed by the modulo-two adders in the encoder. We assume that the encoded digits are transmitted serially over a discrete memoryless channel (or DMC) and we let

$$\underline{r}_t = [r_{t1}, r_{t2}, \ldots r_{tn}]$$

denote the corresponding received branch.

The encoding shift-register is initially loaded with zeroes after which $a_1, a_2, \ldots a_L$ are encoded and then followed by m zeroes (i.e. $a_{L+1} = \ldots = a_{L+m} = 0$) to clear the encoder. L is called the frame length. The case $L=\infty$ is not without interest and in fact most practical threshold decoders for convolutional codes have operated with an infinite frame length or, as it is usually stated, without resynchronization of the encoder. Sequential decoders on the other hand have

always been employed with rather small frame lengths (on the order of 10m.)

We shall let $\underline{a}_{[t,t']}=[a_t,a_{t+1},\ldots a_{t'}]$ denote the information sequence over time units t through t' inclusive, and similarly for $\underline{b}_{[t,t']}$ and $\underline{r}_{[t,t']}$. The Viterbi decoding algorithm [1], which is the maximum likelihood decoding algorithm for a convolutionally-coded frame sent over a discrete memoryless channel chooses as its estimate $\hat{\underline{a}}_{[1,L]}$ of the information sequence that sequence which maximizes the conditional probability

$$P(\underline{a}_{[1,L]}|\underline{r}_{[1,L+m]})$$

and hence this algorithm minimizes the frame-error probability

$$P_{FE}=P_r(\hat{\underline{a}}_{[1,L]}\neq\underline{a}_{[1,L]}) \tag{1}$$

which is the probability of the event that at least one information bit in the frame is wrongly decoded.

For any interesting channel, it must be true that

$$\lim_{L\to\infty} P_{FE} = 1$$

so that P_{FE} is not a meaningful optimality criterion for large frames. The criterion of real interest in all cases is the bit-error probability

$$P_{BE} = \frac{1}{L}\sum_{t=1}^{L} P_r(\hat{a}_t\neq a_t) \tag{2}$$

which gives the fraction of information digits which are wrongly decoded. From (1) and (2) it follows that

$$P_{BE} \leq P_{FE} \leq L\, P_{BE}$$

so that when L is fairly small it makes little difference whether P_{BE} or P_{FE} is minimized (which explains the appropriateness of Viterbi decoding when L is small.)

The bit-error probability, P_{BE}, is minimized by the decoding rule which for each t, $1\leq t\leq L$, chooses its estimate $\hat{a}_t$ as the binary digit which maximizes the conditional probability

$$P(a_t|\underline{r}_{[1,L+m]}).$$

Algorithms, similar to Viterbi's, to accomplish this maximization have been proposed independently by Bahl-Cocke-Jelinek-Raviv [2] and by McAdam-Welch-Weber [3]. These algorithms require receipt of the entire frame $r_{[1,L+m]}$ before decoding begins and so cannot be used without resynchronization. Moreover their implementation requires storage which grows linearly in L and hence are practical alternatives to Viterbi decoding only for small L.

In the following section we derive a recursive procedure for "real-time minimum-bit-error probability decoding" of rate R=1/n convolutional codes to minimize P_{BE} under the constraint that the decoding delay be limited to Δ branches. In section III we formulate the corresponding decoding algorithm and show that its storage requirements are independent of L. For comparison purposes, we formulate a "real-time" modified Viterbi decoding algorithm in Section IV. Section V. reports the results of using these decoding procedures on a simulated additive white Gaussian noise channel. It is concluded that the improvement in P_{BE} for the real-time minimum-bit-error probability decoding algorithm is not enough to justify the added complexity compared to Viterbi decoding but, as shown in Section VI, the new algorithm offers advantages in soft-decision applications such as concatenated coding.

II. Derivation of a Real-Time Minimum-Bit-Error Probability Decoding Procedure.

As in all previous optimal (in some sense) decoding procedures for convolutional codes, we shall make important use of the encoder <u>state</u> which at time t is defined as the contents of the shift-register in Fig. 1 i.e. the m-tuple of past information bits

$$s_t = [a_{t-1}, a_{t-2}, \ldots a_{t-m}] \tag{3}$$

and where, by our convention, $a_i=0$ for $i\leq 0$ and $i>L$. As the term "state" implies, s_t completely accounts for the past history of the encoder input in the sense that

s_t and the input segment $\underline{a}_{[t,L]}$ uniquely determine the output segment $\underline{b}_{[t,L+m]}$. By conditioning on the encoder state, the calculation of the probabilities required for the decision rule can be simplified

The decoding rule which minimizes P_{BE} under the "real-time" constraint that $\hat{a}_t$ be decided from $\underline{r}_{[1,t+\Delta]}$ is that which chooses

$$\hat{a}_t = 0 \quad \text{if}$$

$$P(a_t=0|\underline{r}_{[1,t+\Delta]}) \geq \frac{1}{2} \tag{4}$$

and chooses $\hat{a}_t=1$ otherwise (where we have arbitrarily resolved ties in favor of the decision $\hat{a}_t=0$.) Since

$$P(a_t=0|\underline{r}_{[1,t+\Delta]}) = \frac{P(a_t=0,\underline{r}_{[1,t+\Delta]})}{P(\underline{r}_{[1,t+\Delta]})}$$

and since the probabilities on the righthand side of this latter equation can be expressed as the summation over all states of the joint probabilities including the state, we have

$$P(a_t=0|\underline{r}_{[1,t+\Delta]}) = \frac{\sum_s P(a_t=0,\underline{r}_{[1,t+\Delta]},s_{t+\Delta+1}=s)}{\sum_s P(\underline{r}_{[1,t+\Delta]},s_{t+\Delta+1}=s)}. \tag{5}$$

We now proceed to develop recursive formulas for the two probabilities appearing on the righthand side of (5).

For any t, $t > 1$, we may write

$$P(\underline{r}_{[1,t]},s_{t+1}) = \sum_{s_t} P(\underline{r}_{[1,t]},s_t,s_{t+1}). \tag{6}$$

But also

$$P(\underline{r}_{[1,t]},s_{t-1},s_t) = P(\underline{r}_{[1,t-1]},s_t)P(\underline{r}_t,s_{t+1}|\underline{r}_{[1,t-1]},s_t)$$

$$= P(\underline{r}_{[1,t-1]},s_t)P(\underline{r}_t,s_{t+1}|s_t) \tag{7}$$

where we have made use of the state property and our assumption that the channel is memoryless. Writing

$$P(r_t,s_{t+1}|s_t) = P(s_{t+1}|s_t)P(r_t|s_t,s_{t+1}) \ , \tag{8}$$

we then use the fact seen from (3) that the state $s_{t+1} = [a_t,a_{t-1},\ldots a_{t-m+1}]$ has only two possible predecessors s_t, namely $[a_{t-1},\ldots,a_{t-m+1},0]$ $[a_{t-1},\ldots,a_{t-m+1},1]$, to write

$$P(s_{t+1}|s_t) = \begin{cases} \frac{1}{2} & \text{if } s_t \epsilon P(s_{t+1}) \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where here and hereafter we write $P(s)$ for the set containing the two possible predecessors of a state s. Finally, we note that, for $s_t \epsilon P(s_{t+1})$, since s_{t+1} specifies a_t (by its first digit) then it follows that the encoded branch $\underline{b}_t$ is uniquely determined by s_{t+1} and s_t [and we write $\underline{b}(s_t,s_{t+1})$ for this branch] so that

$$P(\underline{r}_t|s_t,s_{t+1}) = P(\underline{r}_t|\underline{b}(s_t,s_{t+1})) \tag{10}$$

and we note that this quantity is determined by the channel transition probabilities. Substituting (7), (8), (9) and (10) into (6) we have our desired recursion

$$P(\underline{r}_{[1,t]},s_{t+1}) = \sum_{s_t \epsilon P(s_{t+1})} \frac{1}{2} P(\underline{r}_t|\underline{b}(s_t,s_{t+1})) \, P(\underline{r}_{[1,t-1]},s_t). \tag{11}$$

The starting value $P(\underline{r}_{[1,1]},s_2) = P(\underline{r},s_2)$ needed to apply the recursion is simply

$$P(\underline{r}_1,s_2) = \begin{cases} \frac{1}{2} P(\underline{r}_1|\underline{b}(\underline{0},s_2)) & \text{if } \underline{0} \epsilon P(s_2) \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

where we have used the fact that $s_1 = \underline{0}$.

By an entirely analogous argument whose details we omit, the following recursion for the other probability on the righthand side of (5) may be obtained:

$$P(a_t,\underline{r}_{[1,t+i]},s_{t+i+1}) = \sum_{s_{t+i} \epsilon P(s_{t+i+1})} \frac{1}{2} P(\underline{r}_{t+i}|\underline{b}(s_{t+i},s_{t+i+1}))P(a_t,\underline{r}_{[1,t+i-1]},s_{t+i}) \tag{13}$$

which we shall use for $1 \leq i \leq \Delta$. The starting value needed for this recursion is

$$\begin{aligned} P(a_t,\underline{r}_{[1,t]},s_{t+1}) &= P(\underline{r}_{[1,t]},s_{t+1})P(a_t|\underline{r}_{[1,t]},s_{t+1}) \\ &= \begin{cases} P(\underline{r}_{[1,t]},s_{t+1}) & \text{if } a_t \text{ is the first digit of } s_{t+1} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{14}$$

so that the quantities obtained from the recursion (11) directly provide the necessary initial conditions (14) to be used with the recursion (13). Hence we have now obtained a complete recursive procedure for performing real-time minimum-bit-error probability decoding of convolutional codes.

III. Implementation of the Decoding Algorithm

We now describe an algorithm for implementing the decoding procedure whose recursive basis was developed in the previous section. Our algorithm requires the storage of two real numbers for each of the 2^m encoder states. We denote these stored quantities for state s as f(s) and g(s). At time t (in the algorithm) the first of these quantities will store the value

$$f(s) = P(\underline{r}_{[1,t]}, s_{t+1}=s) \tag{16}$$

and the second, will store the value

$$g(s) = P(a_t=0, \underline{r}_{[1,t+\Delta]}, s_{t+\Delta+1}=s). \tag{17}$$

We shall store the previous Δ values of this latter quantity summed over all states and denote these stored values as $G_1, G_2, \ldots G_\Delta$ so that

$$G_i = \sum_s P(a_{t-i}=0, \underline{r}_{[1,t+\Delta-i]}, s_{t+\Delta-i+1}=s) \tag{18}$$

for $1 \leq i \leq \Delta$ will be the stored values at time t (in the algorithm). It follows then that G_Δ and

$$\sum_s f(s) = \sum_s P(\underline{r}_{[1,t]}, s_{t+1}=s) \tag{19}$$

are the desired numerator and denominator on the righthand side of (5) for the decoding of $a_{t-\Delta}$ which is accomplished at time t (in the algorithm.) The only additional storage required is that for the received branches $\underline{r}_t, \underline{r}_{t+1}, \ldots \underline{r}_{t+\Delta}$.

The recursions (11) and (14) directly correspond to the following:

The Real-Time Minimum-Bit-Error Probability Decoding Algorithm

Step 0: Set t=1, set $f(s)\frac{1}{2}P(\underline{r}_1|\underline{b}(0,s))$ for the two states s having 0 as a predecessor, and set f(s)=0 for all other states. Set $G_i=0$ for $1 \leq i \leq \Delta$.

Step 1: Set $g(s)=f(s)$ if the first digit of s is a 0 and set $g(s)=0$ otherwise.

Step 2: For $i=1,2,\ldots\Delta$, make the replacement

$$g(s) \leftarrow \sum_{s'\in P(s)} \frac{1}{2} P(\underline{r}_{t+i} | \underline{b}(s',s)) g(s') \quad \text{for all } s.$$

Step 3: If $t \leq \Delta$, go to step 2. Otherwise set $\hat{a}_{t-\Delta}=0$ if

$$\sum_s f(s)/G_\Delta \geq \frac{1}{2}$$

and set $\hat{a}_{t-\Delta}=1$ otherwise.

Step 4: Increase t by 1. Make the replacements

$$G_{\Delta-i+1} \leftarrow G_{\Delta-i} \quad \text{for } i=1,2,\ldots,\Delta-1$$

and set

$$G_1 = \sum_s g(s).$$

Step 5: Make the replacement

$$f(s) \leftarrow \sum_{s'\in P(s)} \frac{1}{2} P(\underline{r}_t | \underline{b}(s',s)) f(s') \quad \text{for all } s$$

and then return to step 1.

[NOTE: For simplicity we have omitted the obvious "end game" modifications needed when $t>L$ which of course are necessary only if a finite frame length is used. It should also be pointed out that our "trick" of storing the Δ past values of the $g(s)$ summation actually results in a "true decoding delay" of 2Δ branches since we require the use of $\underline{r}_{t+\Delta}$ in step 2 at time t (in the algorithm) when $a_{t-\Delta}$ is decoded. To reduce the "true decoding delay" to Δ branches requires the storage of $\Delta+2$ rather than 2 real numbers per state since $f(s)$ must be updated by Δ branches and the $\Delta-1$ previous values of $f(s)$ stored for each state or, alternatively, the storage of 3 real numbers and considerable extra computation within the algorithm.]

The algorithm as given above is directly suited for software implementation. It should be noted that steps 2, 3, and 5 call for both addition and multiplication of the computed probabilities so that floating-point arithmetic would normally

be chosen for the calculation. [This contrasts with the Viterbi algorithm as will be seen in the next section.] For each t, a total of $\Delta+1$ calculations are made in steps 2 and 5 each involving a sum over all 2^m states [whereas, as we shall see, only one similar calculation is needed for the Viterbi algorithm.]

A hardware realization of the real-time minimum-bit-error algorithm could be made employing 2^m small computational units (or CU's) each of which corresponds to an encoder state s. Each such unit would receive f(s') and g(s') from the two CU's corresponding to the two states s' which are predecessors of s and, with the aid of the received branches as inputs, would compute new values of f(s) and g(s) and pass these values on in turn to the two CU's corresponding to the states for which s is a predecessor. Each CU would execute $\Delta+1$ computational cycles for each t [as compared to 1 cycle for the CU's in a hardware Viterbi decoder.]

IV. Real-Time Viterbi Decoding

As mentioned in section I, the Viterbi decoding algorithm chooses $\hat{\underline{a}}_{[1,L]}$ to be the information sequence which maximizes the conditional probability $P(\underline{a}_{[1,L]}|\underline{r}_{[1,L+m]})$. The following decoding rule, which we call <u>real-time Viterbi decoding</u>, is the natural modification of this rule to satisfy the "real-time constraint" that a_t be decoded from $\underline{r}_{[1,t+\Delta]}$: Choose $\hat{a}_t$ as the digit a_t in the information segment $\underline{a}_{[1,t+\Delta]}$ which maximizes the conditional probability $P(\underline{a}_{[1,t+\Delta]}|\underline{r}_{[1,t+\Delta]})$.

In keeping with our previous notation, we let

$$s_{[t,t']} = [s_t, s_{t+1}, \ldots, s_{t'}]$$

denote the sequence of encoder states from time to to time t' inclusive. It follows from (3) that $s_{[1,t+\Delta+1]}$ and $\underline{a}_{[1,t+\Delta]}$ uniquely determine one another and, moreover, that a_t is the first digit of s_{t+1}. Hence we may rephrase the real-time Viterbi decoding rule as: Choose $\hat{a}_t$ as the first digit of the state s_{t+1} in the state sequence $s_{[1,t+\Delta+1]}$ which maximizes the conditional probability

$$P(s_{[1,t+\Delta+1]}|\underline{r}_{[1,t+\Delta]}) = \frac{P(s_{[1,t+\Delta+1]},\underline{r}_{[1,t+\Delta]})}{P(\underline{r}_{[1,t+\Delta]})} . \tag{20}$$

Since the denominator on the righthand side of (20) is independent of $s_{[1,t+\Delta+1]}$, we can equivalently maximize the numerator alone.

To obtain a recursion for the numerator in (20) we use the same arguments as in Section II which, for $t \geq 2$, give

$$\begin{aligned} P(s_{[1,t+1]},\underline{r}_{[1,t]}) &= P(s_{[1,t]},\underline{r}_{[1,t-1]})P(s_{t+1},\underline{r}_t|s_{[1,t]},\underline{r}_{[1,t-1]}) \\ &= P(s_{[1,t]},\underline{r}_{[1,t-1]})P(s_{t+1},\underline{r}_t|s_t) \\ &= P(s_{[1,t]},\underline{r}_{[1,t-1]})P(s_{t+1}|s_t)P(\underline{r}_t \mid s_t,s_{t+1}) \\ &= \begin{cases} \frac{1}{2} P(\underline{r}_t \mid \underline{b}(s_t,s_{t+1}))P(s_{[1,t]},\underline{r}_{[1,t-1]}) & \text{if } s_t \epsilon P(s_{t+1}) \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{21}$$

Equation (21) is our desired recursion. The initial condition to be used for t=2 is

$$P(s_{[1,2]},\underline{r}_{[1,1]}) = \begin{cases} \frac{1}{2} P(\underline{r}_1|\underline{b}(\underline{0},s_2)) & \text{if } \underline{0}\epsilon P(s_2) \\ 0 & \text{otherwise.} \end{cases} \tag{22}$$

Just as for the ordinary Viterbi algorithm, the key to the efficient implementation of real-time Viterbi decoding is the fact [readily seen from (21)] that the best state sequence [in the sense of maximizing the joint probability on the lefthand side of (21)] $s_{[1,t+1]}$ with $s_{t+1}=s$ must be the extension of the best sequence $s_{[1,t]}$ with $s_t=s'$ or $s_t=s''$ where s' and s'' are the predecessors of s. Hence at each time t the only storage required for each state is the best sequence to that state. Actually, since real-time Viterbi decoding requires only knowledge of the first digit of the state Δ states previously along the sequence, we need store only Δ bits per state together with the joint probability needed for the recursion in (21). We write $B_i(s)$, $1 \leq i \leq \Delta$, and $h(s)$ for these stored quantities which at time t (in the algorithm) have the values

$$h(s)=P(s^*_{[1,t+1]},r_{[1,t]}) \tag{23}$$

where $s^*_{[1,t+1]}$ is the best path $s_{[1,t+1]}$ with $s_{t+1}=s$ and

$$B_i(s) = \begin{array}{l}\text{first digit of state } s_{t+1-i} \\ \text{in the path } s^*_{[1,t+1]}.\end{array} \tag{24}$$

Then we may state:

The Real-Time Viterbi Decoding Algorithm:

Step 0: Set t=1, set $h(s)=\frac{1}{2}P(\underline{r}_1|\underline{b}(\underline{0},s))$ for the two states s having $\underline{0}$ as a predecessor, and set h(s)=0 otherwise. Set $B_i(s)=0$ for $1\leq i\leq\Delta$ and all states s.

Step 1: If $t \leq \Delta$, go to Step 2. Otherwise set

$$\hat{a}_{t-\Delta}=B_\Delta(s)$$

where s is the state for which h(s) is maximum.

Step 2: Increase t by 1. Make the replacement $B_{\Delta-i+1}(s) \leftarrow B_{\Delta-i}(s)$ for i=1,2,..., Δ-1 and for all s.

Step 3: For each s, make the replacement $h(s) \leftarrow \frac{1}{2}P(\underline{r}_t|\underline{b}(s',s))h(s')$ where s' is the predecessor of s which maximizes the replacing quantity and set

$B_1(s)$ = first digit of s'.

Then return to step 1.

The algorithm just given is directly suited for software implementation. Since the algorithm calls for only multiplication of the computed probabilities and selection of a maximum, logarithms may be used with the result that only computer additions are required and hence fixed-point arithmetic would normally be chosen for the calculation. For eact t, only one maximum over all states need be taken, an operation equivalent in complexity to a sum over all states as is required Δ+1 times in the algorithm of the preceeding section. In a hardware realization of the real-time Viterbi algorithm, the CU corresponding to state S would receive h(s') from the two CU's corresponding to the two predecessors s' of s and, with the aid of the received branch, would compute the new value of h(s)

and pass this value on in turn to the two CU's for the states having s as a predecessor. Each CU would execute only one computational cycle for each time unit t and would be somewhat simpler than the CU's described in the preceeding section since only one quantity, h, (rather than two, f and g) would be processed and only additions need be performed.

It should be emphasized that real-time Viterbi decoding may be used for $L=\infty$, i.e. when the convolutional encoder is not periodically resynchronized. Moreover, ordinary Viterbi decoding can be considered the special case of real-time Viterbi decoding for finite L when $\Delta=L+m-1$.

V. Simulation Results

To evaluate the performance of real-time minimum-bit-error probability decoding (hereafter called RTMBEP decoding), a rate R=1/2 convolutional coding system was implemented for a simulated additive white Gaussian noise (AWGN) channel with 8-level output quantization and with binary antipodal signalling. The results of this simulation are given in Table I where E_b is the energy per information bit, N_0 is the one-sided noise power spectral density,and K=m+1 is the encoder constraint length measured in information bits.

The simulation reported in Table I shows that the RTMBEP decoding algorithm gives a noticeable improvement in P_{BE} compared to real-time Viterbi decoding and ordinary Viterbi decoding only for small E_b/N_0, the improvement being about .2dB and .4dB respectively when E_b/N_0 is 0 dB for K=3. We conclude that the slight improvement in P_{BE} for RTMBEP decoding would generally not justify the added complexity relative to real-time Viterbi decoding.

In Table II, we show the effect of the decoding delay Δ on P_{BE} for K=3. From this table we see that a Δ of about 3K is sufficient for near optimal performance.

$\frac{E_b}{N_0}$	K	L	No. Frames Decoded	Δ	P_{BE} RTMBEP Decoding	P_{BE} Real-Time Viterbi	P_{BE} Ordinary Viterbi
0dB	3	2400	1	9	.109	.113	.119
2dB	3	2400	2	9	.0165	.0186	.0173
4dB	3	2400	5	9	.00083	.00083	.00083
0dB	5	2400	1	19	.154	.169	
2dB	5	2400	2	19	.0165	.0184	
4dB	5	2400	5	19	.00033	.0033	

Table I. Results of Decoding for a Simulated Additive White Gaussian Noise Channel

$\frac{E_b}{N_0}$	K	L	No. Frames Decoded	Δ	P_{BE}
0dB	3	1200	1	4	.133
0dB	3	1200	1	5	.125
0dB	3	1200	1	7	.113
0dB	3	1200	1	9	.109
0dB	3	1200	1	19	.106
0dB	3	1200	1	29	.102
2dB	3	1200	2	4	.0325
2dB	3	1200	2	9	.0165
2dB	3	1200	2	19	.0140

Table II. Effect of Decoding Delay Δ for RTMBEP Decoding for a Simulated Additive White Gaussian Noise Channel

VI. Erasure

Although RTMBEP decoding is not an attractive alternative to Viterbi decoding (either real-time or ordinary) for hard-decision decoding, it has good potential for use in soft-decision decoding such as in an inner coding system for concatenated decoding (See [4] for an example of a concatenated scheme using an inner convolutional system.) This follows from the fact that the a posteriori probability

$$P(a_{t-\Delta}=0|\underline{r}_{[1,t]}) = \sum_{s} f(s)/G_{\Delta} \qquad (25)$$

is directly computed (in step 3) by the algorithm so that the quality of the decoding decision for $a_{t-\Delta}$ is directly known.

To indicate the potential of RTMBEP decoding for soft-decision decoding, we consider the simplest case, for the AWGN channel viz. erasure decoding where the decision of $a_{t-\Delta}$ is either 0, 1 or E where E is an erasure symbol. The overall convolutional coding system then converts the channel to a binary symmetric erasure channel (BSEC) as shown in Figure 1 where q is the erasure probability and p is the

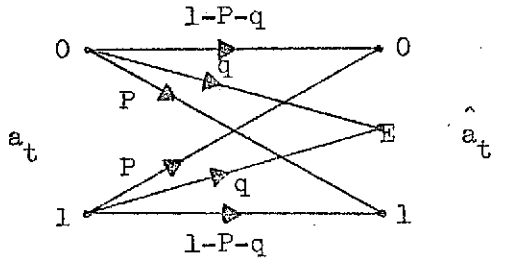

Fig. 2: The Binary Symmetric Erasure Channel (BSEC) Resulting from Erasures Decoding

crossover probability. It whould be stressed that the BSEC model applies for a single decoding decision only since the encoder memory introduces memory into the overall channel. Nonetheless, the capacity C of the memoryless BSEC is a lower bound on the capacity of the true channel and hence a meaningful measure of the quality of the erasures decoding.

The natural erasures rule for RTMBEP decoding is: Choose $\hat{a}_t$=E when

$$\frac{1}{2} - \alpha < P(a_{t-\Delta}=0|\underline{r}_{[1,t]}) < \frac{1}{2} + \alpha \ . \qquad (26)$$

In Table III, we show the results of RTMBEP erasures decoding on the AWGN channel when α in (26) is chosen to maximize the capacity C of the memoryless BSEC. [It is interesting to note that the optimum α in all cases corresponded to the case when 1/3 of the erased digits would have been converted to errors in hard-decision decoding while 2/3 of the erased digits would have been correctly decoded.] We also show the capacity for hard decisions, i.e. q=0, as a standard to evaluate the gain for erasures decoding. The increase in capacity is seen to be .8 dB and 1.0 dB for K=3 and K=5 respectively. We conclude that RTMBEP decoding has considerable potential for use in soft-decision applications for convolutional codes.

$\frac{E_B}{N_0}$	K	L	Δ	No. Frame Decoded	p	q	Capacity C of BSEC	Capacity for Hard Decisions
0dB	3	1200	9	1	.0284	.248	.577	.480
0dB	5	1200	19	1	.0342	.327	.478	.380

Table III: The Effectiveness of RTMBEP Erasures Decoding for the Additive White Gaussian Noise Channel

It should be remarked that there seems to be no simple way to do effective erasures decoding with either real-time or ordinary Viterbi decoding since there is no simple way to determine the quality of the individual bit decoding decisions. For this reason, erasure of the entire frame has been used previously when Viterbi decoding is used for the inner decoder in a concatenated system [4]. This requires the interleaving of many frames to obtain satisfactory performance which is a substantial complication of the overall concatenated system which is avoided when individual bit decoding decisions can be effectively erased.

VII. Summary and Remarks

We have given a fairly comprehensive treatment of optimum real-time decoding of convolutional codes and introduced an algorithm to minimize the decoding bit error probability. We also stated a real-time Viterbi decoding algorithm which, although not previously given in the literature, is probably the form of the Viterbi algroithm which has actually been used in many previous investigations.

We remark that, although we considered only rate R=1/n convolutional codes, our discussion is easily generalized to the case R=k/n where k>1. In this case

$$\underline{a}_t = [a_{t1}, a_{t2}, \cdots a_{tk}]$$

is the vector of information bits at time t. The RTMBEP decoding algorithm as we have given it becomes, mutatis mutandis, the algorithm which minimizes the symbol error probability

$$P_{sE} = \frac{1}{L} \sum_{t=1}^{L} P(\hat{\underline{a}}_t \neq \underline{a}_t).$$

Since a_t contains only k bits, it follows that

$$P_{sE} \leq P_{BE} \leq k\, P_{sE}$$

and since k is always small in applications there seems to be no practical justification for a decoding algorithm which minimizes P_{BE} when $k > 1$.

Our major conclusion from simulations of the RTMBEP decoding algorithm is that, although it does not reduce P_{BE} sufficiently to be a practical alternative to Viterbi decoding in hard-decision applications, the fact that it provides a direct measure of the quality of its decoding decisions makes it an attractive candidate for the inner decoder in concatenated coding systems.

VIII. Acknowledgment

The author is deeply indebted to Prof. James L. Massey who not only spent many hours guiding and commenting on this work but also shaped this paper into its present form.

References

[1] A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimal Decoding Algorithm," IEEE Transactions on Information Theory, IT-13, pp. 260-269, April 1967.

[2] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," 1972 International Symposium on Information Theory, Asilomar, California, January 1972.

[3] P. L. McAdam, L. R. Welch, and C. L. Weker, "M.A.P. Bit Decoding of Convolutional Codes," 1972 International Symposium on Information Theory, Asilomar, California, January 1972.

[4] G. W. Zeoli, "Coupled Decoding of Block-Convolutional Concatenated Codes," IEEE Transactions on Communications, COM-21, pp. 219-226, March, 1973.