THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 51, No. 2, February, 1972
Printed in U.S.A.

Optimal Reception for Binary Partial Response Channels

By M. J. FERGUSON

(Manuscript received September 3, 1971)

This paper describes an exceptionally simple scheme for binary partial response signal formats of the form $a_k \pm a_{k-l}$ *(for* $l \geqq 1$*, and* $a_k = \pm 1$*). The receiver implements the maximum likelihood detector of the sequence* a_k *assuming additive white Gaussian noise as the channel impairment. It is simpler and more efficient than the scheme recently described by G. D. Forney.*[1] *It is, however, not generalizable to multilevel signaling while still retaining its simplicity.*

I. INTRODUCTION

There has recently been considerable interest in using the inherent redundancy of the partial response signal formal to approach the error rate versus signal-to-noise-ratio performance equivalent to binary antipodal signaling. Forney[1] at the 1970 International Symposium on Information Theory discussed a simple decoding scheme which he shows to be asymptotically optimal for high signal-to-noise ratio for channels with white additive Gaussian noise.

This paper describes a receiver for binary partial response signaling which is optimal for white additive Gaussian noise. This demodulator is much simpler than the equivalent two-level scheme of Forney. However, the extension to four or more levels seems to result in a scheme of much greater complexity than Forney's. In the first part of the paper we briefly review binary Class IV partial response signaling. Then we derive the optimal detection scheme for binary signaling which has a particularly simple implementation. A simple analysis of the memory requirements of the implementation follows. Finally we discuss some of the problems of extensions to multilevel signaling.

II. A PARTIAL RESPONSE SYSTEM*

The motivation for binary partial response signaling schemes is to

* This section is almost entirely due to D. D. Falconer.[2] E. R. Kretzmer[3] and A. Lender[4] did the original work in this area and Lucky, Salz, and Weldon[5] have a good survey and summary.

allow transmission of two bits per cycle of bandwidth without requiring ideal boxcar filters. The train of signal waveforms is shaped so that inherent intersymbol interference does not affect decisions made by the receiver.

Figure 1 shows a basic partial response signaling scheme transmitting $1/T$ bits per second. Information bits (a_k) are represented by $+1$s and -1s. Signal shaping is done by the filter whose transfer function is $X(\omega)$. A "Class IV" partial response function $X(\omega)$ and its associated sampled impulse response are shown in Fig. 2. This particular scheme is useful since it has no transmitted dc component. It is used in several existing and proposed partial response modems.

The transmitted Class IV partial response signal $s(t)$ can be represented in terms of the sequence of samples (x_k) spaced at Nyquist intervals (T seconds) as

$$s(t) = A \sum_k x_k \operatorname{sinc}\left(\frac{\pi t}{T} - k\pi\right) \tag{1}$$

where

$$\operatorname{sinc}(x) = \frac{\sin x}{x}$$

and

$$x_k = a_k - a_{k-2}, \qquad k = 1, 2, \cdots. \tag{2}$$

When the information symbols a_k take on values ± 1, then the samples (x_k) have three possible levels: 0, $+2$, or -2. Thus the scheme would be expected to be more sensitive to noise than is a comparable binary antipodal scheme in which $x_k = \pm 1$, and in which the transmitting filter's transfer function is a "boxcar." In fact, when independent hard decisions are made on each bit, it can be shown to require 3 dB higher signal-to-noise ratio in order to achieve the same error rate as the comparable binary antipodal scheme. A more efficient conventional partial response configuration which is 2.1 dB worse than binary antipodal[5] is to provide a matched filter at the receiver by replacing

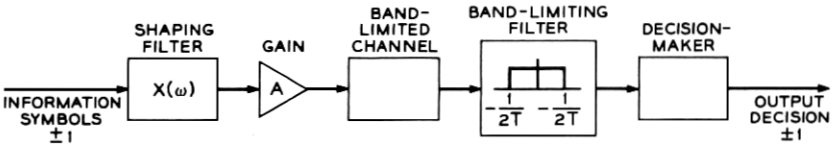

Fig. 1—Partial response system.

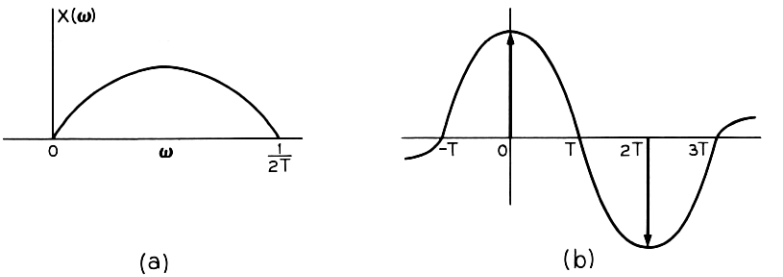

Fig. 2—Class IV partial response: (a) spectrum; (b) sampled impulse response.

$X(\omega)$ at the transmitter by $X(\omega)^{\frac{1}{2}}$ at the transmitter and $X(\omega)^{\frac{1}{2}}$ at the receiver. Other classes of binary partial response systems are worse than the ideal binary antipodal scheme by various amounts (see Table 4-2, page 91, in Ref. 5).

With $a_k = \pm 1$, $x_k = a_k - a_{k-2}$ is a sequence of three-level signals. However not all the sequences are possible! For example, if $a_k = +1$, $x_k = +2$ or 0 and if $a_k = -1$, $x_k = 0$ or -2. All the schemes described use this inherent redundancy to win back the 2.1-dB loss alluded to previously. Finally we note that all partial responses of the form

$$x_k = a_k - a_{k-l}, \qquad l \geqq 1,$$

produce l noninteracting streams of x_ks. For $l = 2$, the even x_ks and the odd x_ks are entirely independent. A scheme for $l = 1$ can be used for any $l \geqq 1$ by time sharing its operation with the other independent streams of x_ks. This observation allowed Forney to assert the applicability of his scheme for all l. It also allows us to consider only $l = 1$.

III. DERIVATION OF OPTIMAL RECEIVER

The receiver that we develop is to be optimal for additive white Gaussian noise and the signaling format

$$x_k = a_k - a_{k-1} \tag{3}$$

with

$$a_{-1} = 1.$$

A simple way to describe the sequence of x_ks resulting from a sequence of a_ks is given by the trellis in Fig. 3. The branches of the trellis are the x_k values and the nodes are the a_k values. The upper nodes are $+1$

and the lower values are -1. We trace a particular sequence of x_ks by following the branches joining the nodes for the appropriate a_k. For instance if we have the sequence starred (*) in Fig. 3, namely $\cdots$,

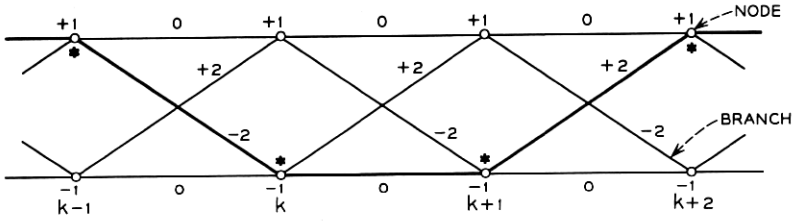

Fig. 3—Signal trellis.

$a_{k-1} = 1$, $a_k = -1$, $a_{k+1} = -1$, $a_{k+2} = +1$, $\cdots$, then we have the output sequence $\cdots$, $x_k = -2$, $x_{k+1} = 0$, $x_{k+2} = +2$, $\cdots$ by following the appropriate branches. Notice that we have the capability of describing any possible sequence of x_ks using Fig. 3. Further we note that any node has two branches leading to it and away from it.

The channel is assumed to add white Gaussian noise n_k, with density $N(0, \sigma^2)$* giving a received signal $y_k = x_k + n_k$. It is well known that the maximum likelihood receiver chooses the infinite sequence of $\hat{a}_k$s which maximize

$$\frac{1}{2}\sum_{k=0}^{\infty} y_k(\hat{a}_k + \hat{a}_{k-1}) - \frac{1}{4}\sum_{k=0}^{\infty}(\hat{a}_k - \hat{a}_{k-1})^2 \tag{4}$$

$$\hat{a}_{-1} = 1,$$

for a given sequence of y_ks. The $\hat{a}_k$s are the estimates of the transmitted sequence $\{a_k\}$. While it is clearly impossible to maximize (4) directly, it is possible to maximize (4) sequentially. We note that we can represent all possible sequences of $\hat{a}_k$ by paths in the signal trellis in Fig. 3. We also note that we can represent all possible sums in (4) as the result of paths through a trellis. We then obtain the trellis in Fig. 4. When $\hat{a}_k$ and $\hat{a}_{k+1}$ are of the same sign, the branch contributes 0 to the sum in (4) but when $\hat{a}_k = +1$, $\hat{a}_{k-1} = -1$, it contributes $y_k - 1$ to the sum. Similarly when $\hat{a}_k = -1$, $\hat{a}_{k-1} = +1$, the branch contributes $-y_k - 1$ to the sum. We say that a specific sequence of $\hat{a}_k$s through the trellis describes a *path*. All paths must have $\hat{a}_k = +1$ or $\hat{a}_k = -1$.

* $N(a, b)$ is the Gaussian density with mean a and variance b.

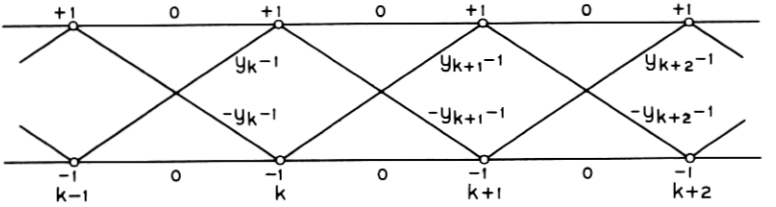

Fig. 4—Received signal trellis.

For all those paths with $\hat{a}_k = +1$ we can write (4) as

$$\left\{\frac{1}{2}\sum_{l=0}^{k} y_l(\hat{a}_l - \hat{a}_{l-1}) - \frac{1}{4}\sum_{l=0}^{k}(\hat{a}_l - \hat{a}_{l-1})^2\right\} + \left\{\sum_{l=k+1}^{\infty} y_l(\hat{a}_l - \hat{a}_{l-1}) - \frac{1}{4}\sum_{l=k+1}^{\infty}(\hat{a}_l - \hat{a}_{l-1})^2\right\} \tag{5}$$

where

$$\hat{a}_k = 1.$$

Thus it is *necessary* that any path with $\hat{a}_k = 1$ and which maximizes (5) also maximizes the first bracketed sum in (5). But this first bracketed sum in (5) depends only on $\{\hat{a}_0, \cdots, \hat{a}_{k-1}\}$ and this portion of the path can be chosen *independently* of the rest of the path. Define

$$f_k^+ \triangleq \max_{\substack{\text{all paths} \\ \text{with } \hat{a}_k = 1}} \left\{\frac{1}{2}\sum_{l=0}^{k} y_l(\hat{a}_l - \hat{a}_{l-1}) - \frac{1}{4}\sum_{l=0}^{k}(\hat{a}_l - \hat{a}_{l-1})^2\right\}. \tag{6a}$$

We similarly define, for the best path leading to $\hat{a}_k = -1$,

$$f_k^- = \max_{\substack{\text{all paths} \\ \text{with } \hat{a}_k = -1}} \left\{\frac{1}{2}\sum_{k=0}^{k} y_l(\hat{a}_l - \hat{a}_{l-1}) - \frac{1}{4}\sum_{l=0}^{k}(\hat{a}_l - \hat{a}_{l-1})^2\right\}. \tag{6b}$$

Finally we see that there are only four branches from the kth to the $(k+1)$st node. Hence, if we have the best path to $\hat{a}_k = \pm 1$, then at $\hat{a}_{k+1} = +1$ we must choose between only two paths, the one coming from $\hat{a}_k = +1$ having a value f_k^+ and the one from $\hat{a}_k = -1$ having a value $f_k^- + y_{k+1} - 1$. The best path is obviously the one with the largest value. Thus we have

$$f_{k+1}^+ = \max \begin{cases} f_k^+ & (+ \text{ PATH}) \\ f_k^- + y_{k+1} - 1 & (- \text{ PATH}). \end{cases} \tag{7a}$$

Similarly we have the best path to $\hat{a}_{k+1} = -1$ as the solution of

$$f_{k+1}^{-} = \max \begin{cases} f_k^{+} - y_{k+1} - 1 & (+ \text{ PATH}) \\ f_i^{-} & (- \text{ PATH}). \end{cases} \tag{7b}$$

Thus at any point in time we have two paths, one of which must be the beginning of the one which optimizes (4).* We say we are not *merged* at $(k-1)$ if we still have two paths left at k. Figure 5 shows

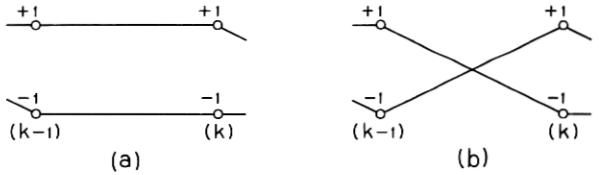

Fig. 5—Possible nonmerge paths.

the only two possibilities to remain unmerged. For the (+) path to go to $\hat{a}_{k+1} = +1$ and the (−) path to go to $\hat{a}_{k+1} = -1$ (Fig. 5a) we need both

$$f_k^{+} - f_k^{-} - y_{k+1} + 1 \geqq 0 \qquad \text{from (7a)}$$

and

$$-(f_k^{+} - f_k^{-}) + y_{k+1} + 1 \geqq 0. \qquad \text{from (7b)}$$

Thus we require

$$-1 \leqq f_k^{+} - f_k^{-} - y_{k+1} \leqq 1. \tag{8}$$

For us to remain unmerged on the "crossover" path of Fig. 5b we require

$$\begin{Bmatrix} \text{and} & (f_k^{+} + f_k^{-}) - y_{k+1} > 1 \\ & (f_k^{+} - f_k^{-}) - y_{k+1} < -1 \end{Bmatrix}.$$

This is clearly impossible. Thus we remain unmerged if and only if (8) is true. Hence when (8) is true, we are unmerged and the most likely path *from* the + node leads *to* the + node, and the most likely

* The formulation and solution of this problem is a simple example of Dynamic Programming and an application of Bellman's Principle of Optimality.[6] This may also be considered as a simple example of the Viterbi Algorithm which was shown by J. K. Omura[7] to be equivalent to Dynamic Programming. Finally, the identical formulation and solution to this problem was also obtained independently by H. Kobayashi[8] and M. Segal (unpublished).

path *from* the $-$ node leads *to* the $-$ node. We also note that

$$f_k^+ - f_k^- - y_{k+1} > 1 \tag{9}$$

implies both best paths came from $\hat{a}_{k-1} = +1$ [a $(+)$ merge] and

$$f_k^+ - f_k^- - y_{k+1} < 1 \tag{10}$$

implies both best paths came from $\hat{a}_{k-1} = -1$ [a $(-)$ merge]. Finally we see that all decisions as to merge or not are based on $f_k^+ - f_k^-$ and not on either separately. We then define

$$\Delta_k \triangleq f_k^+ - f_k^-.$$

Subtracting (7b) from (7a) and noting (8), (9), and (10) gives

$$\Delta_{k+1} = \begin{cases} y_{k+1} + 1, & \Delta_k - y_{k+1} > 1 & (+ \text{ MERGE}) \text{ at } k \\ \Delta_k, & -1 < \Delta_k - y_{k+1} < 1 & (\text{NO MERGE}). \\ y_{k+1} - 1, & \Delta_k - y_{k+1} < -1 & (- \text{ MERGE}) \end{cases} \tag{11}$$

The optimal receiver implements (11). We see that while unmerged, Δ_k remains the same. Only the testing to see if we have finally merged depends on the incoming data. The value of Δ_k while unmerged is just that resulting from the two paths leading from the most recent merge. Thus if the most recent merge was $(+)$ at node $l - 1$ then $\Delta_k = y_l + 1$ for $k \geqq l$ and no merge. Between merges, only two sequences are possible, either $\{1, 1, 1, \cdots\}$ or $\{-1, -1, -1, \cdots\}$. Hence in our implementation all we have to do is save Δ_l and the location of the most recent merge. Since we will be placing our data in a storage register prior to outputing it, we must decide which of the two between-merge sequences should we store. Obviously, the best is the most likely of the two.

We remember that after a $(+)$ merge at the $(k - 1)$ node

$$\Delta_k = y_k + 1.$$

If the $(+)$ merge is *correct* then

$$\begin{aligned} y_k &= a_k - a_{k-1} + n_k \\ &= -1 + a_k + n_k \end{aligned}$$

giving

$$\Delta_k = a_k + n_k .$$

Since $a_k = \pm 1$, then $\Delta_k = 1 + n_k$ if the transmitted path leads to the $+$ node and $\Delta_k = -1 + n_k$ if the transmitted path leads to the

— node. Hence the determination of the most likely path leading *from* the $(k-1)$ node is a binary hypothesis testing problem. The solution is to say the most likely path leads to the + node if $\Delta_k > 0$ and the — node if $\Delta_k < 0$.

For a correct — merge, the test is identical. The most likely path initially then is the one leading to $\text{sgn}(\Delta_k)$. If when we finally merge, the sign of Δ_l at the merge point is the same as the merge, the most likely path is the same as the most likely path initially chosen on the basis of $\Delta_k \gtrless 0$.

IV. IMPLEMENTATION

An implementation is suggested by the flow diagram of Fig. 6. We suppress the subscripts. The newly received signal is y and the previously stored difference is Δ. The decoded data are stored in a

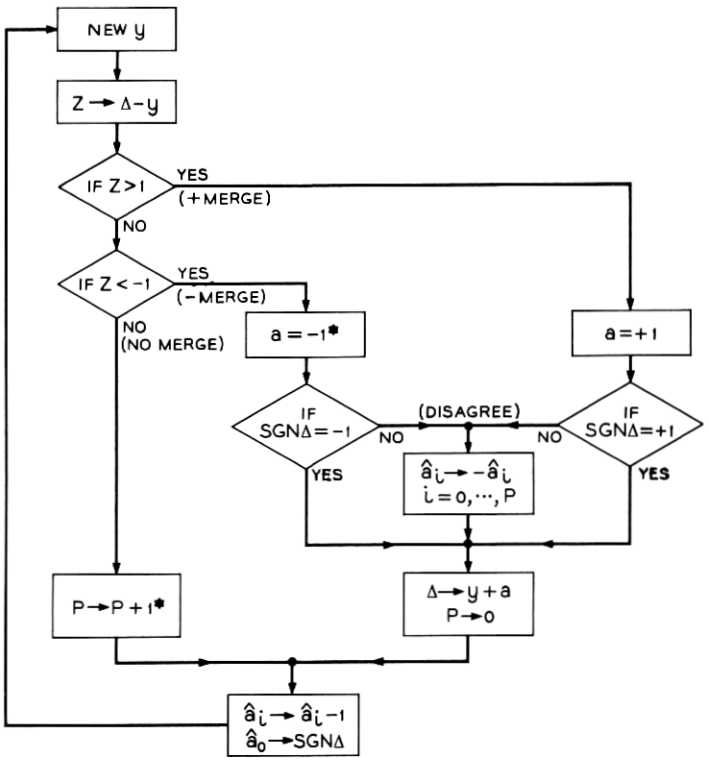

Fig. 6—Flow diagram for $x_k = a_k - a_{k-1}$.

register of length $N + 1$. $\hat{a}_o$ is the most recent decision and $\hat{a}_N$ is the bit about to be outputed. There is also a pointer, p, which indicates where the first data bit *after* the most recent merge point is located in the data register. The equations implemented are those of (11). The values of $\hat{a}_o, \cdots, \hat{a}_p$ are those of the initially most likely path as described in Section III. Referring to Fig. 6; when a new value y is obtained, we subtract it from the stored difference Δ calling this sum z. We then check to see if a merge has occurred according to (11). If $z > 1$ then we have a $+$ merge, if $z < -1$ we have a $-$ merge, and if $-1 \leqq z \leqq 1$ we have no merge. If a merge has occurred, then we will eventually replace Δ by $y + 1$ for a $+$ merge and $y - 1$ for a $-$ merge. We thus let $a \triangleq \pm 1$ for a $\pm$ merge. If the most likely path is actually the one we have been saving, then they must agree at the merge. We check this by finding out whether sgn Δ is the same as the merge value ± 1. If it is, then we have saved the most likely path. If it does not agree, the most likely path is the complement of the one we saved up to the most recent merge point p. We then complement $\hat{a}_o, \cdots, \hat{a}_p$. After we have our data set up, we replace Δ by its new value $y + a$, and reset the pointer p to 0. At this point we shift the register and place $\hat{a}_o = \text{sgn } \Delta$.

If there is no merge, then life is simpler; Δ is the same and the pointer is advanced by 1; the register is then shifted and $\hat{a}_o = \text{sgn } \Delta$. We are now ready for a new piece of data. Figure 7 shows a possible implementation of the above flow diagram.

V. BUFFER OVERFLOW (P > N) STRATEGY

Since we are saving the most likely sequence, we just output the buffer and keep $p = N$. If when we merge, we do indeed have the most likely sequence, then all is fine. If the most likely sequence is not actually held, then we complement the entire register. Although we have sent some suboptimally detected bits, this appears to be the best strategy. We could save ourselves most of the problem if we differentially encoded ("PRECODED"[5]) the data. This means we would let

$$x_k = \tilde{a}_k - \tilde{a}_{k-1}$$

where

$$\tilde{a}_k \tilde{a}_{k-1} = a_k .$$

Under these circumstances, a single suboptimal decision in the decoded

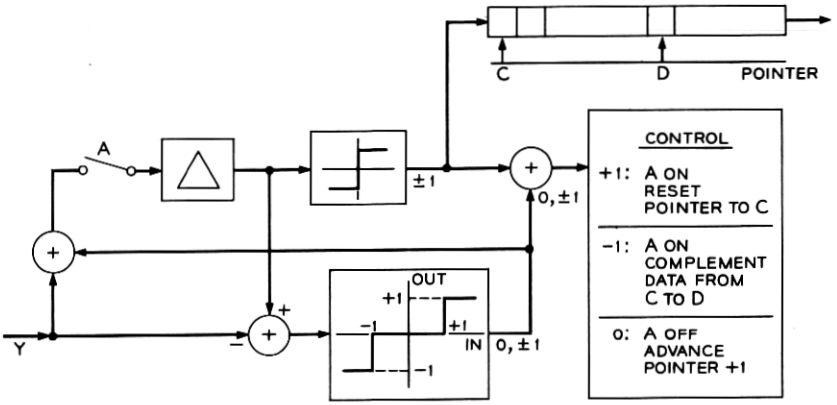

Fig. 7—Possible implementation for $x_k = a_k - a_{k-1}$.

$\tilde{a}_k$ path results in two errors in the a_k path. However, if we hold the complement $\tilde{a}_k$ path after a merge point rather than the most likely path, we only make a *single* suboptimal decision in decoding the a_ks rather than a possibly long burst. If we now complement the entire register, then we make an additional single error for the data bit affected by both $\hat{a}_N$ and $\hat{a}_{N+1}$ because $\hat{a}_N$ was complemented and $\hat{a}_{N+1}$ was not. If we do not complement the entire register, then the same phenomenon occurs at the next merge point. In both cases we make only two single suboptimal decisions whenever the buffer overflows. This obviously makes differential encoding of the data advisable.

VI. ANALYSIS OF BUFFER SIZE

In this section we determine the approximate probability of overflow of a buffer of length $(N + 1)$. If we have a $(+)$ merge then

$$\Delta = y_0 + 1 \tag{12}$$

and for a $(-)$ merge

$$\Delta = y_0 - 1 \tag{13}$$

where the "0" subscript refers to the merge position. Because y_0 is $N(-2, \sigma^2)$ for a $+-$ transition, $N(0, \sigma^2)$ for a $++$ or $--$ transition, and $N(+2, \sigma^2)$ for a $-+$ transition (see Fig. 3) we have, for a correct decision, substituting into (12) and (13), that Δ is $N(1, \sigma^2)$ for a transition *leading to* a $+$ node and Δ is $N(-1, \sigma^2)$ for a transition *leading to* a $-$ node for both types *of* previous merges. We know we remain

unmerged for N transitions if

$$-1 \leqq \Delta - y_l \leqq 1 \qquad l = 1, \cdots, N. \tag{14}$$

Since the $++$, $+-$, $-+$, $--$ transitions are all equally likely, we have $\Delta - y_l$ with the densities

$$\Delta - y_l \sim N(+3, \sigma^2) \text{ with probability } 1/4 \tag{15a}$$

$$\Delta - y_l \sim N(+1, \sigma^2) \text{ with probability } 1/4 \tag{15b}$$

$$\Delta - y_l \sim N(-1, \sigma^2) \text{ with probability } 1/4 \tag{15c}$$

$$\Delta - y_l \sim N(-3, \sigma^2) \text{ with probability } 1/4. \tag{15d}$$

For small σ^2 (large signal-to-noise ratio) both (15a) and (15d) lead to (14) *not* being satisfied with a very high probability. We can ignore these two events. (15b) and (15c) both correspond to $y_l \sim N(0, \sigma^2)$ and occur together with probability 1/2. Because of the symmetry of (14), (15b), (15c), and y_l we can write the probability of no merge for at least N nodes as

$$\begin{aligned} P(N) &\cong Pr(-1 \leqq \Delta - y_l \leqq 1, l = 1, \cdots, N) \\ &\cong \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\,\sigma} \exp - \frac{(\Delta - 1)^2}{2\sigma^2} \left[\frac{1}{2} \int_{-1-\Delta}^{1-\Delta} \frac{1}{\sqrt{2\pi}\,\sigma} e^{-y^2/2\sigma^2}\, dy\right]^N d\Delta. \end{aligned}$$

Now if σ^2 is small then Δ is concentrated about 1 and the limit $-1 - \Delta$ can be replaced by $-\infty$. We then have

$$P(N) \cong \left(\frac{1}{2}\right)^N \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\,\sigma} e^{-(x^2/2\sigma^2)} \left[\int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}\,\sigma} e^{-y^2/2\sigma^2}\, dy\right]^N dx. \tag{16}$$

Letting

$$Q(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}\,\sigma} e^{-y^2/2\sigma^2}$$

and noting

$$dQ(x) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-x^2/2\sigma^2},$$

we write (16) as

$$\begin{aligned} P(N) &= \left(\frac{1}{2}\right)^N \int_{-\infty}^{\infty} [Q(x)]^N\, dQ(x) \\ &= \frac{1}{N+1} \left(\frac{1}{2}\right)^N. \end{aligned} \tag{17}$$

To obtain (17) we have really used only the fact that the distributions of Δ and y are translates of each other and that they are symmetric about their mean value. Equation (17) was also derived independently by Kobayashi.[8]

The Forney scheme[1] has a probability of buffer overflow of 2^{-N}. Equation (17) indicates a factor of N improvement in this case. For $N = 20$, (17) gives $P(N) = 4.5 \times 10^{-8}$.

VII. GENERALIZATIONS

The most obvious generalization we would like is to four-level signaling. We can obtain a signal trellis in the same way as in the binary case but now we have four nodes at each time instant. Again we can write, using the same arguments as before, equations equivalent to (7). However, all the special structure which led to the exceptionally simple results of (11) seems to be missing. Instead of only one set of no-merge paths as indicated by Fig. 5a, we have many. Instead of only one possible way for either a + or − merge to occur, we have several. It also appears that all four possible paths through the trellis must be kept. In short, for a four-level signaling, Forney's scheme seems to be the simplest but for binary signaling, the one described here is best.

VIII. CONCLUSIONS

The system here is applicable to partial response signaling with binary data of the form

$$x_k = a_k \pm a_{k-l} \qquad \text{for all } l \geqq 1.$$

Differential encoding of the data is helpful to reduce the effects of buffer overflow. The extension to multilevel signaling destroys the beauty and simplicity of the binary scheme.

IX. ACKNOWLEDGMENTS

The author wishes to thank R. Gitlin and J. Salz for stimulating discussion.

REFERENCES

1. Forney, G. D., "Error Correction for Partial Response Modems," talk presented at the 1970 International Symposium on Information Theory, Noordwijk, the Netherlands, June 1970.
2. Falconer, D. D., "Forney's Error Correction Scheme for Partial Response Signals," unpublished work.

3. Kretzmer, E. R., "Generalization of a Technique for Binary Data Communication," IEEE Trans. Commun. Technology, *COM-14*, February 1966.
4. Lender, A., "The Duobinary Technique for High-Speed Data Transmission," 1963 IEEE Winter General Meeting.
5. Lucky, R. W., Salz, J., and Weldon, E. J., Jr., *Principles of Data Communication*, New York: McGraw-Hill, 1968, pp. 83–92.
6. Bellman, R., *Dynamic Programming*, Princeton, N. J.: Princeton University Press, 1957.
7. Omura, J. K., "On Optimum Receivers for Channels with Intersymbol Interference," talk presented at the 1970 International Symposium on Information Theory, Noordwijk, the Netherlands, June 1970.
8. Kobayashi, H., "Correlative Level Coding and Maximum-Likelihood Decoding," IEEE Trans. Inf. Theory, *IT-17*, No. 5 (September 1971), pp. 586–594.