**Peter A. Franaszek**

# A General Method for Channel Coding

*A procedure is described for constructing minimum delay codes for discrete noiseless channels. The method is based on a simple recursive algorithm for finding a set of coding paths.*

## Introduction

Consider the problem of formulating fixed-rate (synchronous) codes for discrete noiseless channels of the type first considered by Shannon [1]. These codes, sometimes referred to as constrained or channel codes, are of interest in such applications as signal conditioning for digital transmission or magnetic recording as well as protocol design. A recent paper by Franaszek [2] (which may be used as an entry into the literature) develops necessary conditions for constructing block codes with minimum look-ahead. That is, for a coding rate of $\alpha$ bits per $N$ channel symbols, a look-ahead or delay bound of $M$ requires that, at each time $t$, $t = 1, 2, \cdots$, the encoder emits a word of length $N$ as a function of the next $M\alpha$ bits to be transmitted. Such codes are somewhat less general than the bounded delay codes of [3], where words are permitted to be of variable length. Loosely speaking, a variable-length code has the property that choice of a string of $N$ channel symbols may be a function of past as well as future bits. This can yield a lower value of $M$ than the look-ahead block coding case.

A recursive algorithm, given in [3], was shown to yield an outcome, which, if negative, precludes the possibility of constructing a bounded delay code. We show here a simplified version of the algorithm that provides a similar result for the block coding case. Moreover, a positive outcome is also sufficient for code construction. The possibility of such an encoding is ensured by having the coding process be a function of sufficient information concerning previous states and code words. The construction technique is shown only for the special case of $M = 2$, but can readily be extended to the case of general $M$. Also described is a method for reducing the coding delay $M$ at the expense of storing additional previous information, a new approach to constructing bounded delay codes.

## The method

We first consider the problem of minimum look-ahead block coding. The procedure yields encodings such that the $r$th word $W_r$ of $N$ channel symbols, transmitted at time $r$, is chosen given knowledge of the next $M\alpha$ bits to be transmitted as well as (at most)

1. The preceding $(M - 1)$ words;
2. The state occupied at time $r - M + 1$.

Decoding requires the above information as well as

3. $V$ succeeding words.

The decoding delay $V$ is a function of the channel constraints and the code mapping. This as well as the information represented by (1) and (2) may often be dispensed with if the assignment of information symbols to coding paths is done properly [2, 3].

Following [2], let $S = \{\sigma_i\}$, $i = 1, 2, \cdots, E$ denote the states for the channel transition model, and let

$\mathbf{D} = \{d_{ij}\}$

be the channel skeleton transition matrix, where $d_{ij}$ represents the number of distinct sequences of $N$ channel symbols leading from $\sigma_i$ to $\sigma_j$. Let $\phi_i$ be the *weight* of state $\sigma_i$, where $2^{-\alpha(M-1)} \le \phi_i \le 1$, and, loosely speaking, represents the number of $\alpha$-bit messages that can be transmitted from state $\sigma_i$. Reference [3] gives a dynamic programming procedure for obtaining a set of $\phi^*_i$ which upper bound the $\phi_i$. In fact, if variable-length codes [3] are not of specific interest, the following simpler recursive algorithm may be used to obtain the $\phi^*_i$.

- *Algorithm 1*

Let $\phi^*_i(n)$ denote the state weights at the $n$th iteration. Then

$$\phi^v_i(n+1) = 2^{-\alpha}\left[\sum_{ij} \phi^*_j(n)\right],$$

$$\phi^*_i(n) = 2^{-(M-1)\alpha} \llcorner [2^{(M-1)\alpha}\phi^v_i(n)],$$

subject to the condition that

$$\phi^*_i \le 1, \forall_i,$$

$$\phi^*_i(0) = 1.$$

The procedure is continued until there is no change over one iteration [*i.e.*, $\phi^*_i(n+1) = \phi^*_i(n), \forall_i$]. The symbol $\llcorner$ denotes the floor function.

If all the $\phi^*_i$ are zero, then no code exists with the given parameter $M$. Otherwise, a code may be constructed as shown below. Henceforth, for simplicity, it is assumed that $M = 2$.

Consider a state $\sigma_i$ with weight $\phi^*_j$. The quantity $\phi^*_i$ represents an integral number of *independent paths* [2], where an independent path (corresponding to an integer or possibly fractional number of code words) may represent a specific sequence of $\alpha$ bits (*e.g.*, 00 for $\alpha = 2$), followed by any other bits. If $\sigma_i$ is entered by a word, then the specific sets of $\alpha$ bits to be represented by $\sigma_i$ may be determined by the previous state, but those that follow are unknown and thus may comprise any of the $2^\alpha$ possibilities.

A code may be constructed by assigning to each state $\sigma_i$ a sufficient number of independent paths $P_1, P_2, \cdots$, corresponding to specific code words, so as to meet the weight bound $\phi^*_i$. These lead to succeeding states, whose independent paths can be assigned $\alpha$-bit sequences as a function of $\sigma_i$. Coding is then performed as a function of the next $M\alpha$ bits, coupled with the information specified by (1) and (2). The procedure is perhaps best illustrated by an example.

*Example* Consider the channel model shown in Fig. 1. A code will be constructed with one bit per symbol. The channel capacity [1] $C \gtrsim 1$, so some path pruning is required. This is done automatically by the above method.

The $\phi^*_i$ for $M = 2$ are respectively 1, 1, 1/2, 1/2. A feasible set of path assignments is shown in Table 1(a).

The code is completed by assigning specific $\alpha$-bit vectors to independent paths associated with successor states. Note that the weight used from $\sigma_2$ when continuing path $P_2$ from $\sigma_1$ is 1/2, although the available weight is one. An alternative would be to use weight 1, thus eliminating the need for code word $c$ from $\sigma_1$.

Suppose a starting state of $\sigma_3$ is chosen for transmission, and that the message is 010. See Table 1(b). A preamble of one bit is required, since $\sigma_3$ is of weight 1/2. The encoding is *fabd*. The preamble corresponds to $x_1$ from $\sigma_3$, encoded as symbol $f$, since the next bit is a 0. This

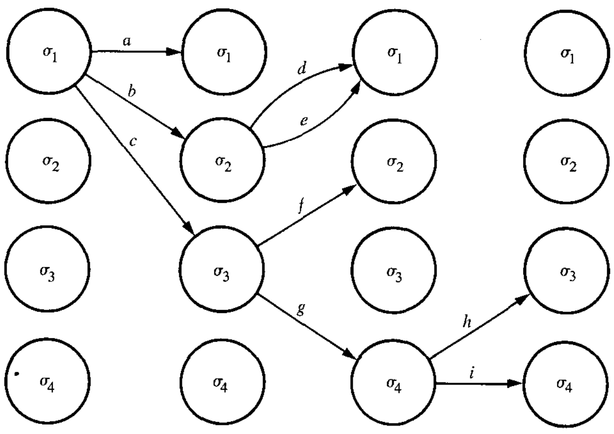


**Figure 1** Trellis of a restricted channel.

**Table 1(a)** Independent path assignments, Example 1.

| *State* $\sigma_i$ | *Code word* | *Next state* $\sigma_j$ | *Weight used from* $\sigma_j$ |
|---|---|---|---|
| $\sigma_1$ | $P_1\{a$ | $\sigma_1$ | 1 |
| | $P_2\{b$ | $\sigma_2$ | 1/2 |
| | $c$ | $\sigma_3$ | 1/2 |
| $\sigma_2$ | $P_1\{d$ | $\sigma_1$ | 1 |
| | $P_2\{e$ | $\sigma_1$ | 1 |
| $\sigma_3$ | $P_1\{f$ | $\sigma_2$ | 1/2 |
| | $g$ | $\sigma_4$ | 1/2 |
| $\sigma_4$ | $P_1\{h$ | $\sigma_3$ | 1/2 |
| | $i$ | $\sigma_4$ | 1/2 |

**Table 1(b)** Completed code, Example 1.

| *State* $\sigma_i$ | *Next* $M\alpha$ *bits* | *Code word* |
|---|---|---|
| $\sigma_i$ | $P_1\{x_1\,0$ | $a\,P_1(\sigma_1)$ |
| | $x_1\,1$ | $a\,P_2(\sigma_1)$ |
| | $P_2\{x_2\,0$ | $b\,P_1(\sigma_2)$ |
| | $x_2\,1$ | $c\,P_1(\sigma_3)$ |
| $\sigma_2$ | $P_1\{x_1\,0$ | $d\,P_1(\sigma_1)$ |
| | $x_1\,1$ | $d\,P_2(\sigma_1)$ |
| | $P_2\{x_2\,0$ | $e\,P_1(\sigma_2)$ |
| | $x_2\,1$ | $e\,P_2(\sigma_2)$ |
| $\sigma_3$ | $P_1\{x_1\,0$ | $f\,P_1(\sigma_2)$ |
| | $x_1\,1$ | $g\,P_1(\sigma_4)$ |
| $\sigma_4$ | $P_1\{x_1\,0$ | $h\,P_1(\sigma_3)$ |
| | $x_1\,1$ | $i\,P_1(\sigma_4)$ |

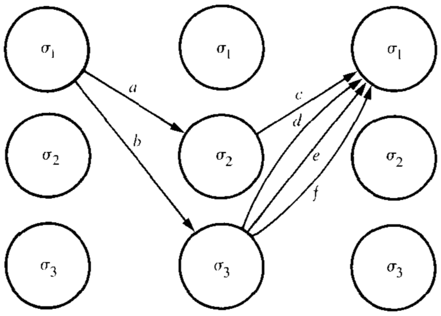


**Figure 2** Coding paths for a channel model.

**Table 2(a)** Independent path assignments, Example 2.

| *State* $\sigma_i$ | *Code word* | *Next state* $\sigma_j$ | *Weight used from* $\sigma_j$ |
|---|---|---|---|
| $\sigma_1$ | $P_1\{a$ | $\sigma_2$ | 1/2 |
| | $P_1\{b$ | $\sigma_3$ | 1/2 |
| | $P_2\{b$ | $\sigma_3$ | 1 |
| $\sigma_2$ | $P_1\{c$ | $\sigma_1$ | 1 |
| $\sigma_3$ | $P_1\{d$ | $\sigma_1$ | 1 |
| | $P_2\{e$ | $\sigma_1$ | 1 |
| | $P_3\{f$ | $\sigma_1$ | 1 |

**Table 2(b)** Completed code, Example 2.

| *State* $\sigma_i$ | *Next* $M\alpha$ *bits* | *Code word* |
|---|---|---|
| $\sigma_1$ | $P_1\{x_1\,0$ | $a\ P_1(\sigma_2)$ |
| | $P_1\{x_1\,1$ | $b\ P_2(\sigma_3)$ |
| | $P_2\{x_2\,0$ | $b\ P_2(\sigma_3)$ |
| | $P_2\{x_2\,1$ | $b\ P_3(\sigma_3)$ |
| $\sigma_2$ | $P_1\{x_1\,0$ | $c\ P_1(\sigma_1)$ |
| | $P_1\{x_1\,1$ | $c\ P_2(\sigma_1)$ |
| $\sigma_3$ | $P_1\{x_1\,0$ | $d\ P_1(\sigma_1)$ |
| | $P_1\{x_1\,1$ | $d\ P_1(\sigma_1)$ |
| | $P_2\{x_2\,0$ | $e\ P_1(\sigma_1)$ |
| | $P_2\{x_2\,1$ | $e\ P_1(\sigma_1)$ |
| | $P_3\{x_3\,0$ | $f\ P_1(\sigma_1)$ |
| | $P_3\{x_3\,1$ | $f\ P_2(\sigma_1)$ |

zero bit is encoded by independent path $P_1$ from $\sigma_1$, corresponding to symbol $a$. The following bit, 1, is encoded as $P_2(\sigma_1)$, which corresponds to either a $b$ or a $c$ as a function of the succeeding bit. This is a 0, so that $b$ is chosen. The last bit, 0, is encoded as a $d$.

This illustrates the coding procedure, but should not be taken as good coding practice, since a mapping exists for $M = 1$, with $\sigma_1$ and $\sigma_2$ as principal states [3]. □

The above was an example of a code with $\alpha = 1$. Here code words must end in states with weights of either 1/2 or 1. Each word corresponds to at most one independent path, so that decoding may be done given knowledge of only the preceding word and its associated channel state. For $\alpha > 1$, this is not necessarily the case. A single word may correspond to more than one independent path and decoding thus require knowledge of $V$ succeeding words, where $V$ is a function of the channel and chosen mapping. Design of a good code thus requires an attempt to minimize $V$, as well as such factors as state dependence [2, 3], in order to simplify the coder/decoder combination and minimize error propagation.

As mentioned in [2], $\phi_i \le 1$ results in a coding process where at each time $r$ the number of independent paths available is never greater than required to meet the rate of $\alpha$ bits per $N$ symbols. But a surplus of paths at state $\sigma_i$ may compensate for a deficit at some state $\sigma_j$. This is equivalent to permitting $\phi_i > 1$, which may be done by simply renormalizing the state weights, as illustrated by the following example.

*Example 2* Consider the sequence constraints shown in Fig. 2. The channel capacity is one bit per symbol. Algorithm 1 yields, for $M = 3$, $\phi_1$, $\phi_2$, $\phi_3$ equal to 1/2, 1/4, and 3/4, respectively. The weights $\phi_i$ are zero for $M = 2$.

Suppose the weights are multiplied by $2^\alpha = 2$, to obtain 1, 1/2, 3/2, respectively, and to reduce the delay to $M = 2$. Consider Table 2(a).

Note that $\sigma_3$ has associated with it three independent paths of length 1. A coding table similar to that of the previous example is given in Table 2(b).

Coding requires that, on entry to $\sigma_3$, information be kept concerning whether the independent path taken from $\sigma_1$ was $P_1$ or $P_2$. This is distinctly preferable to a code with $M = 3$, which has substantially more cumbersome coding and decoding tables. Note that the decoding delay $V = 1$ whenever path $b$ is taken from $\sigma_1$ to $\sigma_3$. □

## Conclusion

The techniques outlined here provide a means for coding once a set of nonzero state weights have been found via Algorithm 1. However, in many practical cases a fixed- or variable-length code with no look-ahead may yield a better implementation. It may be noted, for example, that the channel model shown in Fig. 2 admits a fixed-length code for $\alpha = 2$, $N = 2$, with no look-ahead or state dependency.

## References


1. C. E. Shannon, "A Mathematical Theory of Communication," *Bell Syst. Tech. J.* **47,** 143–157 (1948).
2. P. A. Franaszek, "On Future-Dependent Block Coding for Input-Restricted Channels," *IBM J. Res. Develop.* **23,** 75–81 (1979).
3. P. A. Franaszek, "Synchronous Bounded Delay Coding for Input Restricted Channels," *IBM J. Res. Develop.* **24,** 43–48 (1980).


*Received April 11, 1980*

*The author is located at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598.*