

CONVOLUTIONAL CODING

The present invention is concerned with the coding of digital signals using a convolutional code, and decoding such coded signals.

According to the present invention there is provided a method of coding digital data for transmission comprising combining sequences of bits to be coded with groups of one or more consecutive bits having a predetermined value or values and coding the combined sequence using a systematic convolutional code; characterised in that the said bit or bits having a predetermined value or values are omitted from the transmitted sequence.

In another aspect, the invention provides an apparatus for coding digital data comprising:

- (i) means (103) for combining a sequence of bits to be coded with at least one group comprising one or more consecutive bits having a predetermined value or values to form a modified sequence;
- (ii) a convolutional coder for coding the modified sequence using a systematic code to produce parity; and
- (iii) means for formatting an output sequence containing the parity bits and the bits of the input sequence but not containing the said bit or bits having a predetermined value or values.

In a further aspect the invention provides an apparatus for decoding such coded data, comprising:

- (i) timing means for gaining frame synchronisation;
- (ii) means for inserting into a received bit sequence one or more bits having a predetermined value or values at locations determined by the timing means to produce a second sequence;

- (iii) a decoder for decoding the second sequence in accordance with a convolutional code.

In a yet further aspect the invention provides an apparatus for decoding a digital signal which has a framing structure and has been coded by a systematic convolutional coder following insertion of one or more bits having a predetermined value or values at one or more predetermined positions, the apparatus including timing means to synchronise to the frame structure; means synchronised by the timing means for substituting, at the predetermined position(s), a bit or bits having the said predetermined value(s) in place of the bits received, and a decoder for decoding the bits in accordance with the convolutional code.

Some embodiments of the invention will now be described, by way of example, with reference to the accompanying drawings, in which:

Figures 1 and 2 are block diagrams of two known forms of convolutional coder;

Figure 3 is a state diagram for the coder of Figure 1;

Figure 4 is a trellis diagram illustrating the operation of a known Viterbi decoder;

Figure 5 is a block diagram of a coding apparatus in accordance with one embodiment of the invention;

Figure 6 is a timing diagram for the apparatus of Figure 5;

Figure 7 is a block diagram of a decoding apparatus in accordance with a second embodiment of the invention;

Figure 8 is a timing diagram for the apparatus of Figure 7;

Figure 9 is a block diagram of a decoding apparatus in accordance with a third embodiment of the invention; and

Figure 10 is a timing diagram for the apparatus of Figure 9.

First, the basic concepts of convolutional coding and Viterbi decoding will be explained.

A convolutional code is a type of error-correcting code; that is, the signals are coded in such a way that when the coded signals are received, with some of the bits in error, they can be decoded to produce decoded signals which are error-free, or at least have a lower occurrence of errors than would be the case without the use of an error-correcting code. Necessarily this process involves the introduction of redundancy. In convolutional coding, each bit (or each group of more than one bit) of the signal to be coded is coded by a convolutional coder to produce a larger number of bits; the coder has a memory of past events and the output thus depends not only on the current input bit(s) but also on the history of the input signal.

The rate of the code is the ratio of input bits to output bits; for example a coder which produces  $n$  coded bits for each  $k$  input bits has a rate  $R$  of  $k/n$ . The coder output may consist of  $k$  input bits and  $n-k$  bits each generated as a function of two or more input bits, or may consist entirely of such generated bits (referred to as parity bits). The former situation is referred to as a systematic code and the latter as a non-systematic code. In this specification the term systematic code will be used to include the situation where the output includes some but not all of the signal bits. The parity bits are commonly formed by modulo-2 addition of bits selected from the current data bit and earlier data bits, though non-linear elements and feedback paths may also be included. A typical  $1/2$  rate systematic coder with  $k=1$ ,  $n=2$  is shown in Figure 1. It receives serial data bits  $D_1$  into a two-stage delay line with delay elements 1, 2 and generates in a modulo-2 adder 3 a bit  $P_1$  which is  $D_1 \oplus D_{1,2}$ .

In general this function is defined by a subgenerator  $g$  which is a binary sequence indicating which of the delay

line taps do (1) or do not (0) participate in the addition. In Figure 1 the subgenerator for  $D_i$  is 100 and that for  $P$  is 101. These may be written in octal notation viz (4,5). The commutator 4 indicates that bits  $D_i$ ,  $P_i$ ,  $D_{i+1}$ ,  $P_{i+1}$ , etc are output serially in that order.

Figure 2 shows a convolutional coder for a (23,35) non-systematic code - i.e. with subgenerators 10011 and 11101.

Because of the coder memory, a given data bit affects the values of a number of successive generated bits. As the delay line in Figure 1 has one stage the data bit  $D_i$  contributes to two generated bits  $P_i$ ; it is said to have a constraint length of  $K=2$ . The coder of Figure 2 has a constraint length of  $K=5$ . We also define (for the purposes of this description) a coded constraint length i.e. the number of output bits which are a function of a given input bit. This length  $K_c = nK$ . If (unusually) the coder has two subgenerators with different constraint lengths then we take the larger value for computing  $K_c$ .

The process of decoding a signal coded by means of a convolutional code now needs to be considered. Because the code is a redundant code, not all output sequences are possible. One method of decoding with a systematic code is to use an algebraic decoder to feed the data bits into a local coder and compare the locally generated parity bits with the received parity bits. If these are the same, the received data bits are deemed to be valid. If they are different, then this shows that one or more of the received bits is in error. If the level of errors does not exceed the correction capability of the code, the erroneous bit can be located and hence corrected. Looking at the problem more generally, consider for the moment a complete message which has been received after coding and transmission over a channel which is subject to errors. If the received sequence is an allowable sequence then it has been received without error (or with errors such as to transform it from

the original sequence into another allowable sequence); either way, errors are assumed not to be present and the message can be readily decoded. If however the received message is not an allowable sequence, the decoding problem becomes one of identifying that allowable sequence which has the maximum correlation with the received sequence and decoding that. It can be shown that dealing with the whole message in this way produces the maximum likelihood of decoding the message correctly. It is not at once instinctively obvious why, when determining the value of a given data bit, one would wish to look at coded bits more than the coded constraint length. For example, in the following sequence, generated by the coder of Figure 2,

|              |          |              |              |              |              |              |     |
|--------------|----------|--------------|--------------|--------------|--------------|--------------|-----|
| $D_{i-1}$    | $D_i$    | $D_{i+1}$    | $D_{i+2}$    | $D_{i+3}$    | $D_{i+4}$    | $D_{i+5}$    | ... |
| $P_{i-1}(1)$ | $P_i(1)$ | $P_{i+1}(1)$ | $P_{i+2}(1)$ | $P_{i+3}(1)$ | $P_{i+4}(1)$ | $P_{i+5}(1)$ | ... |
| $P_{i-1}(2)$ | $P_i(2)$ | $P_{i+1}(2)$ | $P_{i+2}(2)$ | $P_{i+3}(2)$ | $P_{i+4}(2)$ | $P_{i+5}(2)$ | ... |

$D_{i+5}$ ,  $P_{i+5}(1)$  and  $P_{i+5}(2)$  clearly are independent of  $D_i$  and ostensibly are of no use in ascertaining its correct value in the presence of transmission errors. However, supposing that parity bit  $P_{i+3}(1)$  (for example) has been incorrectly received along with other unspecified errors which together mean that  $D_i$  cannot be correctly decoded. It may be that the information carried by  $P_{i+5}(1)$  (which is less than the coded constraint length distant from  $P_{i+3}(1)$  and, like it, is a function of  $D_{i+1}$ ,  $D_{i+2}$ ,  $D_{i+3}$  is of value in correcting the error in  $P_{i+3}(1)$  and thus permits a resolution of the value of  $D_i$ .

In many cases it is not practical in terms of system delay or decoding complexity to look at the whole of a message, but rather to look at bits within a time window of limited duration. The algebraic decoder discussed above and the maximum likelihood case correspond to window durations of the coded constraint length and infinity

respectively. The error performance of a decoder approaches asymptotically that of the maximum likelihood case as the window length increases.

As the size of window increases, the complexity of performing the required correlation increases, and it is therefore common to use a Viterbi decoder which in effect deals with each received n-bit group in succession and updates partial correlations but reduces the complexity by continually discarding correlation values (and hence candidate sequences) judged to be poor - although in fact the Viterbi decoder is generally described in terms of accumulated signal distance.

In order to describe its operation, it is necessary to note that the contents of the coder delay line are referred to as the state of the coder at any time each time the coder receives a data bit (or, more generally, k data bits) it undergoes a "state transition" from that state to another (or to the same) state. If the decoder assumes that the coder was in a particular state when a particular bit group was transmitted it can compare the received bit group with the bit groups which the coder is capable of producing where a transition occurs from that state; the signal distance between them being a measure of the probability of that transition having actually occurred.

Figure 3 shows a state diagram for the coder of Figure 1, where the states 00, 01, 10, 11 (where the contents of the delay line states 1, 2 are shown as the most and least significant bits respectively) are represented by circles with the corresponding numbers within them. Arrows show all the possible transitions between any two states, the corresponding output bits D, P are shown adjacent each arrow. If, at some point in time, the decoder has ascribed to each state a signal distance value, then having carried out the above comparison to calculate a distance for each

of the eight possible transition paths one then adds this to the distance value for the state from which the transition proceeds, thereby obtaining two updated distance values for each destination state. This process may be illustrated by way of example. Suppose that the sequence 101101001 has been coded using the coder of Figure 1: its generator is {11 00 01 00 00 etc} so its output is

|    |    |    |    |    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|----|----|----|----|
| 11 | 00 | 01 |    |    |    |    |    |    |    |    |    |
|    | 00 | 00 | 00 |    |    |    |    |    |    |    |    |
|    |    | 11 | 00 | 01 |    |    |    |    |    |    |    |
|    |    |    | 11 | 00 | 01 |    |    |    |    |    |    |
|    |    |    |    | 00 | 00 | 00 |    |    |    |    |    |
|    |    |    |    |    | 11 | 00 | 01 |    |    |    |    |
|    |    |    |    |    |    | 00 | 00 | 00 |    |    |    |
|    |    |    |    |    |    |    | 00 | 00 | 00 |    |    |
|    |    |    |    |    |    |    |    | 00 | 00 | 00 |    |
|    |    |    |    |    |    |    |    |    | 11 | 00 | 01 |

---

11   00   10   11   01   10   00   01   11   00   01

Suppose, further, that the received sequence is

11 00 11\* 11 01 10 00 .....

with a single error in the asterisked bit.

The decoding process shown in Figure 4 assumes that the transmission starts with the coder in state (00) in column 1 (the first node). The first pair of bits 11 has a Hamming distance of 2 and 0 respectively from the paths to states (00) and (10) at node 2: these distances are written adjacent the arrowheads. The second pair of received bits 00 has distances 0, 0, 2, 2 from the next four possible paths to node 3: adjacent each state are written the transmitted data associated with the path to that point, and the accumulated Hamming distance. From the third pair of received bits one identifies eight possible paths to the four states at node 4; if this process continues, we will

have sixteen possible paths to the next node, then 32 and so on. However, each pair of paths converging on a given state in node 4 has a certain difference in accumulated Hamming distance - e.g. - a difference of 1 at state (00) - and extension of these paths over a common extension path to later nodes will not change this difference. Therefore we can at once discard the path having the larger distance, leaving one survivor path at each node. The discarded paths are struck through in Figure 4 (where two paths have the same distance, usually the one having the lower position in the diagram is arbitrarily chosen for deletion). Note that at node 4, all survivor paths imply that the first data bit is "1" despite the error in the sixth (parity) bit, although the point at which such lack of ambiguity is apparent (if at all) will, in general, depend on the particular error pattern.

Continuing the same process to node 5, we see that the correct data 1001 is identifiable as having the lowest accumulated distance (though of course a decision on all four bits would not be taken at this point because the decoder does not know that the error is in the sixth rather than seventh bit).

Assuming a finite decoding window is employed, then the usual procedure is to decide, on the basis of the results over the window, upon the earliest data bit within that window, either by observing an unambiguous result or by choosing that result having the lowest accumulated Hamming distance (e.g. 10 ... at state 11 in node 5). Note that such a decision may be inconsistent with earlier decisions - i.e. the final result may imply assumption of a non-allowable path sequence. For many applications this does not matter, but if it does then correction may be made.

The above example assumed that the coder starting state was known; although not essential this provides improved results if it can be done. At the end of a transmission,

it is usual to append "tailing bits" e.g. zeros to ensure the generation of more than one parity bit from the last few data bits. If the number of tailing bits is equal to (or greater than) the length of the coder memory, and the identity of the tailing bits is "known" to the decoder, then this has the added advantage that since the final coder state is known no decision needs to be taken in the decoder between conflicting results at the last node.

In the above description, it has been assumed that the data to be decoded are in binary (i.e. 1/0) form; if the received signal is derived from a modulated signal it is inherent in this assumption that a hard decision as to the value of the received bit has already been made. However a soft demodulator may be used, i.e. one which gives an analogue output (or more commonly a digital word) representing a value in the range 0 to 1 - for example an FSK (frequency shift keying) demodulator may indicate the relative position of the received frequency between the nominal transmitted frequencies rather than indicating which it is closest to. In this case the above decoding can proceed as a soft decision decoder and actual signal distances are used.

In coding speech using LPC coding or other parametric representation of the speech, it is usual to code the speech on a frame by frame basis; i.e. the speech is divided into successive time frames and parameters (one or more of which hold for the whole frame) are generated for each frame.

It has been proposed, where such speech data are convolutionally coded, to insert tailing bits at the end of each frame, thereby improving error performance. This effectively decouples the frames from one another - i.e. decoding of a frame is not affected by errors in adjacent frames.

In the coding apparatus of Figure 5, speech signals received at an input 100 are coded into digital form on a frame-by-frame basis by a speech coder 101 of conventional construction. Any other digital data divided into frames could, be coded by the apparatus if desired (speech being merely one example). The output of the coder 101 is a serial bit stream synchronous with a bit clock  $\phi$  from a master timing unit 102. A timing diagram is shown in Figure 6. This bit stream is assumed to contain inter-frame gaps to accommodate tailing bits and a framing code. At the end of each frame of data the output is forced to logic zero for two bit periods by a pulse  $\phi_r$  operating on a selector 103, which has the effect of concatenating the data with tailing bits of value zero. The output of the selector is fed to a half-rate convolutional coder 104 employing a systematic code: in this example the (20, 35) code is employed. The coder 104 produces, as well as the data D, parity bits P. These are interleaved by a commutator 105, driven by clock pulses  $2\phi$  at twice the rate of pulses  $\phi$ . Finally a framing pulse of  $\phi_r$  switches a selector 106 to append a framing code (shown arbitrarily as 101001 in Figure 6) from a framing code generator 107. Apart from the fact that the 'tailing bit' pulse  $\phi_r$  lasts for only 2 bit periods whilst the convolutional coder 104 has four delay stages operation of the apparatus as described so far is conventional.

However, during the period that  $\phi_r$  is active (high) the supply of timing pulses to the convolutional coder is switched by a selector 108 so that it receives pulses  $2\phi$  instead of  $\phi$ . Thus the zero data input during  $\phi_r$  is interpreted as four tailing bits and the convolutional coder produces four parity bits and is then restored to state zero. These four parity bits (marked P1 ... P4 in the timing diagram) are not however intercalated with data - the latter (known to be zero) are discarded by virtue of

the fact that a selector 109 controlled by the pulse  $\phi_T$  bypasses the commutator and passes P1 ... P4 directly to the selector 106.

Figure 7 shows a decoder for decoding the output of the coder; Figure 8 is a timing diagram for the decoder.

Signals input at an input 200 are conducted to a timing unit 201 which establishes bit synchronisation and (by recognising the framing code) frame synchronisation.

It produces pulses  $\phi_D$ ,  $\phi_P$  which serve to clock alternate received bits into flip-flops 202, 203 providing data and parity inputs to a Viterbi decoder 204.

The parity bits P1 ... P4 are not passed to the flip-flop 202; instead its input is forced to zero for four periods of the clock  $\phi_D$  by a selector 205 controlled by pulses  $\phi_{TR}$  from the timing unit 201, thereby re-creating the tailing bits which were discarded at the coding end. The bits P1 ... P4 are re-timed to be synchronous with  $\phi_P$  by a four stage delay line 207 (clocked by pulses  $2\phi$ ) and the relevant bits selected by AND-gates 208-212 enabled by pulses  $\bar{\phi}_{TR}$  (via an inverter 213),  $\phi_1$ ,  $\phi_2$ ,  $\phi_3$ ,  $\phi_4$  and an OR-gate 214 before passing to the flip-flop 203. The Viterbi decoder is supplied with pulses  $\phi_{TR}$  which (as previously described) identify the point at which the convolutional coder 204 has reached state 0000 and causes it to restrict its search accordingly.

Figure 9 is a block diagram of a second embodiment of decoder, for use with a conventional coder. This time the tailing bits are transmitted; but when they are received, zeros are substituted irrespective of the value actually received - thereby correcting any transmission errors on those bits.

Received data at an input 300, consisting of alternate data and parity bits, are clocked alternately into flip-flops 302, 303, feeding a Viterbi decoder 304 as before, by pulses  $\phi_D$ ,  $\phi_P$  from a timing unit 301. However during the

period during which the tailing bits (shown in a timing diagram Fig. 10 as T1 ... T4) are received the input to the flip-flop 302 is forced to zero by a pulse  $\phi_{TR}$  acting on a selector 305. As before,  $\phi_{TR}$  is also supplied to the Viterbi decoder 304.

The above decoders shown in Figures 7 and 9 have been described on the basis that they receive as input a binary sequence - i.e. a hard decision has already been taken as to whether the bit is a '0' or a '1'.

It has already been mentioned that a soft demodulator can be used in systems of this kind; and this may be done if desired in the case of Figures 7 and 9, in which case the information routed to the Viterbi decoder is not a single bit but a value in the range 0 and 1 (or a single bit plus a confidence measure between 0.5 and 1). In this case the reinserted (Fig. 7) or substituted (Fig. 9) bits may be indicated to the Viterbi decoder as having the value zero (or zero with maximum confidence).

CLAIMS

1. A method of coding digital data for transmission comprising combining sequences of bits to be coded with groups of one or more consecutive bits have a predetermined value or values and coding the combined sequence using a systematic convolutional code; characterised in that the said bit or bits having a predetermined value or values are omitted from the transmitted sequence.
2. An apparatus for coding digital data comprising:
  - (i) means (103) for combining a sequence of bits to be coded with at least one group comprising one or more consecutive bits having a predetermined value or values to form a modified sequence;
  - (ii) a convolutional coder for coding the modified sequence using a systematic code to produce parity bits; and
  - (iii) means for formatting an output sequence containing the parity bits and the bits of the input sequence but not containing the said bit or bits having a predetermined value or values.
3. An apparatus for decoding data coded by the method of Claim 1, comprising:
  - (i) timing means for gaining frame synchronisation;
  - (ii) means for inserting into a received bit sequence one or more bits having a predetermined value or values at locations determined by the timing means to produce a second sequence;
  - (iii) a decoder for decoding the second sequence in accordance with a convolutional code.



4. An apparatus for decoding a digital signal which has a framing structure and has been coded by a systematic convolutional coder following insertion of one or more bits having a predetermined value or values at one or more predetermined positions, the apparatus including timing means to synchronise to the frame structure; means synchronised by the timing means for substituting, at the predetermined position(s), a bit or bits having the said predetermined value(s) in place of the bits received, and a decoder for decoding the bits in accordance with the convolutional code.

5. An apparatus according to Claim 3 or 4 including demodulation means to supply a received bit sequence accompanied by or embodying confidence measures for the bits thereof, wherein the insertion means or, as the case may be, substitution means is arranged to supply to the decoder a measure indicative of maximum confidence in respect of the inserted or substituted bits.

6. An apparatus for coding data substantially as herein described with reference to Figures 5 and 6 of the accompanying drawings.

7. An apparatus for decoding data substantially as herein described with reference to Figures 7 and 8 of the accompanying drawings.

8. An apparatus for decoding data substantially as herein described with reference to Figures 9 and 10 of the accompanying drawings.

