GENERALIZED FEEDBACK DECODING OF CONVOLUTIONAL CODES

Wai-Hung Ng
The Aerospace Corporation
Los Angeles, California

ABSTRACT

The use of convolutional codes with feedback decoding is the most common error-correction technique in simple communication systems. A drawback of conventional feedback decoding is the limitation to a class of self-orthogonal codes which, in general, are non-optimum. Based on distance properties of the utilized code and test-error pattern analysis, we propose generalized feedback decoding that does not have the above mentioned limitation. This is minimum distance decoding and can be applied to any convolutional code while still maintaining its simplicity. Therefore, it has the advantage of being easily adopted in the existing systems. We can use complicated Viterbi or sequential decoder in large terminals and, with the same code, use the proposed decoder in small terminals; otherwise, both large and small terminals must utilize the same type decoder. Also, we may use the proposed decoding scheme to simplify and accelerate sequential type decoding. In addition, by means of the special recovery property of convolutional codes, advanced ARQ retransmission systems could be much improved; several practical applications are suggested and discussed in the last section of this paper.

1. INTRODUCTION

Various decoding approaches to convolutional codes have resulted in significant improvements in practical digital communication applications. In general, feedback decoding is utilized in simple systems, Viterbi or sequential decoding in relatively complicated systems, and stop-and-wait or continous retransmission (ARQ for detection only or hybrid forward error-correction/ARQ) in computer communication systems over noisy channel. Thus, it is desirable that, with appropriate decoders, all these different systems could use a common code to minimize their interface problem. However, conventional feedback decoding requires self-orthogonal codes [1] , which are not suitable for other decoders, and therefore obstructs such an approach.

In this paper, we propose generalized feedback decoding which can be applied to any convolutional code, but still maintains its simplicity. In order to give a clear illustration, the

discussion will be concentrated on binary rate 1/2 single generator convolutional codes. The approach presented, however, can be extended to other code rates.

The basic decoding concept is to derive two small sets of one-to-one corresponding sequences: a set of special test-error patterns, and a set of paths selected from the code tree. Both are stored in a small memory for decoding use. During the real time decoding process, whenever a code path at minimum distance from the received sequence is found, the decoder carries out the basic branch operation (BBO). That is, the decoder shifts out the earliest branch (or segment) in the tentatively decoded sequence $\underset{\sim}{w}$ as a correct representation of the corresponding transmitted branch. Since the previous branches of $\underset{\sim}{w}$ are tentatively accepted, there are only two possible choices for the newly decoded branch, namely, the two branches connected to the last branch previously selected. The decoder selects the one which is closer to the newly received branch. Then, the test-error sequence $\underset{\sim}{t}$ (i.e., $\underset{\sim}{t} = \underset{\sim}{w} \oplus \underset{\sim}{v}$, the modulo-2 sum of the tentatively decoded sequence $\underset{\sim}{w}$ and the received sequence $\underset{\sim}{v}$) is checked by the memory for an exact match to one of the stored special patterns. If an exact match is found, the corresponding code path stored in the memory is modulo-2 added to both $\underset{\sim}{w}$ and $\underset{\sim}{t}$ to replace the original $\underset{\sim}{w}$ and $\underset{\sim}{t}$, then returns to BBO.

This algorithm is minimum distance decoding, which guarantees better performance than conventional feedback decoding. Examples are given for both theoretical analysis and system design to show its simplicity in hardware implementation.

Although the main purpose of this study is to introduce the generalized feedback decoding, ways of utilizing properties of convolutional codes to further simplify and speed up sequential type decoding and suggestions for improving ARQ retransmission system will also be discussed.

2. CONVOLUTIONAL CODE TREE

General descriptions of single generator, binary, convolutional codes are available in the literature [2, 3, 4]. These codes are group codes and can be represented in a convenient way by a code tree which extends indefinitely from any node. For a rate one-half code, each message digit is encoded into two code digits which are determined by the generator sequence $\underset{\sim}{g}$, the present message digit, and the K-1 previous message digits, where K is the constraint length of the code in branches. For the generator sequence, we strongly recommend the notation $\underset{\sim}{g} = g(2^0)g(2^1)g(2^2) \ldots g(2^i) \ldots g(2^{K-1})$, where $g(2^i)$ is arbitrary code branch in $\underset{\sim}{g}$ and $0 \leq i \leq K-1$, for it can clearly show the inter-relationship among branches in the code tree. Development of a general initial code tree is given in Fig. 1. In this presentation, we use a rate 1/2 code generated by $\underset{\sim}{g}$ = 11 01 00 01 00 01 . . . $g(2^{K-1})$ to illustrate our decoding procedure. The development of this code tree is given in Fig. 2 for ready reference.

Without loss of generality, we shall state that two code branches stemming from the same node are always binary complements of each other, either 00 and 11 or 01 and 10. Thus, independent of the received code branch, two branches of test-error patterns stemming from the same node are either 00 and 11 or 01 and 10. Since choosing between 01 or 10 would make no difference in terms of distance, we could impose a rule that under BBO, the newly accepted test-error pattern must be either 00 or 01, and eliminate the possibilities of being equal to 10 and 11.

By utilizing code properties, the design of our decoder presented later is made very simple. However, to begin our explanation, we shall assume that the decoder has two K-branch shift registers to hold the tentatively decoded sequence $\underset{\sim}{w}$ and the test-error sequence $\underset{\sim}{t}$. For any of the sequence, say $\underset{\sim}{t}$, we let $\underset{\sim}{t}_b$ be the last b branches of $\underset{\sim}{t}$. Thus, $\underset{\sim}{t}_1$ is the last branch of $\underset{\sim}{t}$ and, under BBO, $\underset{\sim}{t}_1$ must be either 00 or 01.

The basic decoding strategy of the proposed approach is always to derive a path $\underset{\sim}{w}$ at minimum distance $|\underset{\sim}{t}|$ from the received sequence $\underset{\sim}{y}$, where $|\underset{\sim}{t}|$ represents the Hamming weight of $\underset{\sim}{t}$. Whenever a $\underset{\sim}{w}$ is found which is at minimum distance $|\underset{\sim}{t}|$ from $\underset{\sim}{y}$, the decoder returns to BBO. If the new $\underset{\sim}{w}$ results in $\underset{\sim}{t}_1 = 00$ (or $|\underset{\sim}{t}_1| = 0$), we can be sure that it has minimum test-error weight. But, if $\underset{\sim}{t}_1 = 01$ (or $|\underset{\sim}{t}_1| = 1$), we have the possibility that some other path may have smaller weight. Thus, when the BBO results in $|\underset{\sim}{t}_1| \neq 0$, we must check whether a backup search is needed or not. In the next section, we shall introduce a simple and efficient step to solve this problem.

3. GENERALIZED FEEDBACK DECODING

In this section, we shall begin with the analysis of coding properties and their utilization before introducing the implementation.

3.1 Test-Error Pattern Tree

In the last section, we explained that under BBO, $\underset{\sim}{t}_1$ must be either 00 or 01, and when $\underset{\sim}{t}_1 = 01$, a backup search may be needed. Here, we shall concentrate on the case of a tentatively decoded sequence $\underset{\sim}{w}$ resulting in $\underset{\sim}{t}_1 = 01$.

3.1.1 Determination of Exact Backup Distance b_t^*

We let d(b) be the minimum Hamming distance between pairs of paths, one from each of opposite half-trees of b branches in length [3]. If there is a $\underset{\sim}{w}'$ of the same length as $\underset{\sim}{w}$ and diverging from $\underset{\sim}{w}$ at $b_t^* = b$ branches back but having less test-error weight, it implies $|\underset{\sim}{w}'_b \oplus \underset{\sim}{y}_b| = |\underset{\sim}{t}'_b| < |\underset{\sim}{w}_b \oplus \underset{\sim}{y}_b| = |\underset{\sim}{t}_b|$. By analyzing this statement, we found that b must satisfy the necessary condition, denoted as $T^*(b)$ where $T^*(b) \leq |\underset{\sim}{t}_b|$: (i) $T^*(b) = \lceil (d(b) + 1)/2 \rceil$ if d(b) is odd and $d(b-1) < d(b) \leq d(b+1)$, (ii) $T^*(b) = \lceil (d(b) + 3)/2 \rceil$ if d(b) is odd and $d(b-1) = d(b)$, (iii) $T^*(b) = \lceil (d(b) + 2)/2 \rceil$ if d(b) is even (Proof of these assertions is given in Appendix A). Therefore, whenever BBO results in a $\underset{\sim}{t}_1 = 01$, a subsearch would

be needed if the backup distance b falls into one of the above three categories of $|\underset{\sim}{t}_b| \geq T^*(b)$. The necessary threshold condition $T^*(b)$ on exact backup distance $b_t^* = b$ for the rate 1/2 code shown in Figure 2 is tabulated in Table 1.

3.1.2 Development of Minimum Test-Error Pattern Tree

By utilizing the initial code tree and based on the basic property of BBO, the minimum test-error pattern tree is developed branch-by-branch prior to the real time decoding process. At each branch, if backup search is required and b_t^* is determined, a $\underset{\sim}{t}'_b$ is searched for from the code tree to replace the $\underset{\sim}{t}_b$ with greater test-error weight. Since convolutional codes are group codes, the $\underset{\sim}{t}$ derived from the decoding process is solely affected by the channel noise sequence and therefore the searched $\underset{\sim}{t}'_b$ is independent of the transmitted code sequence [8]. For this reason, we have the option to take any path in the code tree to be $\underset{\sim}{w}$ in developing the minimum test-error pattern tree. Here, the top path in the initial code tree, the all-zero digit sequence is chosen to be $\underset{\sim}{w}$ from which we derive $\underset{\sim}{v} = \underset{\sim}{w} \oplus \underset{\sim}{t} = \underset{\sim}{t}$. Knowing $\underset{\sim}{v} = \underset{\sim}{t}$, we will then be able to search for a $\underset{\sim}{w}'$ at the minimum distance from $\underset{\sim}{v}$.

When the decoder starts BBO from the first branch, independent of what the received branch $\underset{\sim}{v}_1$ is, there are only two possible $\underset{\sim}{t}_1$ s : 00 and 01. Since both $\underset{\sim}{t}_1$ s have weight $|\underset{\sim}{t}_1| \leq 1 < T^*(1) = 2$ (refer to Table 1), the decoder returns to BBO which again results in two new pairs of possible $\underset{\sim}{t}_1$ s, 00 and 01, to add to each of the first branches. Therefore, there are four possible $\underset{\sim}{t}_2$ s : 00 00, 00 01, 01 00, and 01 01. The first three $\underset{\sim}{t}_2$ s have weight $|\underset{\sim}{t}_2| \leq 1 < T^*(2) = 2$, but the last $\underset{\sim}{t}_2$ has weight $|\underset{\sim}{t}_2| = 2 = T^*(2)$ which implies that a backup search would be needed at $b_t^* = 2$.

As can be seen in Figure 2, if $\underset{\sim}{w}_2 = 00\ 00$ and $\underset{\sim}{v}_2 = \underset{\sim}{t}_2 = 01\ 01$, there is a $\underset{\sim}{w}'_2 = 11\ 01$ with $\underset{\sim}{t}'_2 = 10\ 00$ such that $|\underset{\sim}{t}'_2| = 1 < |\underset{\sim}{t}_2| = 2$. Thus we replace $\underset{\sim}{t}_2 = 01\ 01$ by the minimum test-error pattern $\underset{\sim}{t}'_2 = 10\ 00$ before continuing BBO for the third branch, and we also store this pair of $\underset{\sim}{t}_2$ and $\underset{\sim}{t}'_2$ in a small memory for future decoding use: whenever BBO results in a $\underset{\sim}{t}$ whose $\underset{\sim}{t}_2 = 01\ 01$, the decoder will replace this $\underset{\sim}{t}_2$ by the minimum test-error pattern $\underset{\sim}{t}'_2 = 10\ 00$, then returns to BBO.

By utilizing the same procedure of adding a pair of $\underset{\sim}{t}_1$s, 00 and 01, to each of the minimum test-error patterns, we could branch-by-branch build up a tree which has the following two properties: (i) each $\underset{\sim}{t}_b$ in the tree represents one of the minimum test-error patterns, and all $\underset{\sim}{t}_b$ patterns in the tree represent all possible minimum test-error patterns; (ii) during the process of either building the tree or in real time decoding, if BBO results in a $\underset{\sim}{t}$ whose tail sequence $\underset{\sim}{t}_b$ is the same as one of the $\underset{\sim}{t}_b$s previously stored in the memory, we shall directly replace this $\underset{\sim}{t}_b$ by its corresponding $\underset{\sim}{t}'_b$ in the memory. The resultant $\underset{\sim}{t}'$ is guaranteed for having minimum weight and thus the decoder can return to BBO without further backup search (Proof of this statement is given in Appendix B).

Figure 3 demonstrates the minimum test-error pattern tree up to five branches deep; the underlines indicate where a $\underset{\sim}{t}_b$ has been replaced by a $\underset{\sim}{t}'_b$. There are totally six pairs of corresponding $\underset{\sim}{t}_b$s and $\underset{\sim}{t}'_b$s required for $b \leq 6$, and they are tabulated in Table 2. It is therefore guaranteed to correct double errors (as $d(6) = 5$) with search length of six branches. Further simplification of this decoding process is described in the next subsection.

3.2 Permissible Path Decoding

The modulo-2 sum of each corresponding pair of $\underset{\sim}{t}_b$ and $\underset{\sim}{t}'_b$ described in the last subsection is the modulo-2 sum of the corresponding pairs of $\underset{\sim}{w}_b$ and $\underset{\sim}{w}'_b$ $((\underset{\sim}{t}_b \oplus \underset{\sim}{t}'_b) = (\underset{\sim}{w}_b \oplus \underset{\sim}{v}_b) \oplus (\underset{\sim}{w}'_b \oplus \underset{\sim}{v}_b) = (\underset{\sim}{w}_b \oplus \underset{\sim}{w}'_b))$ which according to the closure property of the group code is a special truncated path in the lower-half initial code tree. We denote this path to be permissable path, $\underset{\sim}{P}_{(i)} = \underset{\sim}{t}_b \oplus \underset{\sim}{t}'_b$, where i is the sequential order of the permissible path. Detailed properties of permissible paths are available in the reference list [3, 6]. We found that storing a corresponding pair of $\underset{\sim}{t}_b$ and $\underset{\sim}{P}_{(i)}$ instead of a pair of $\underset{\sim}{t}_b$ and $\underset{\sim}{t}'_b$ in the memory could minimize the hardware complexity because the relationship between a pair of $\underset{\sim}{t}_b$ and $\underset{\sim}{t}'_b$ is one-to-one corresponding while the relationship between $\underset{\sim}{t}_b$ and $\underset{\sim}{P}_{(i)}$ is many-one corresponding (Refer to Table 2).

The basic approach of utilizing permissible path decoding is more readily envisioned when organized into a schematic diagram. Fig. 4 shows the detailed functional and operational diagram of generalized feedback decoding with search length $b \leq 6$. The decoder consists of three shift registers each having six branches long to hold the $\underset{\sim}{v}$, $\underset{\sim}{w}$, $\underset{\sim}{t}$, and a small memory device to store a set of six $\underset{\sim}{t}_{(i)}$ s and a corresponding set of $\underset{\sim}{P}_{(i)}$ s, where $1 \leq i \leq 3$. The decoding procedure is the following:

(i) We pre-hardwire the set of $\underset{\sim}{t}_{(i)}$ and $\underset{\sim}{P}_{(i)}$ to the shift registers holding $\underset{\sim}{t}$ and $\underset{\sim}{w}$. (ii) Consider that each message digit x_t is encoded into $x_t c_t$, and received as $x'_t c'_t$, and received as $x'_t c'_t$ where $x'_t = x_t \oplus n'_t$, $c'_t = c_t \oplus n''_t$, and n'_t, n''_t are corresponding channel noise digits. Therefore, a message sequence $x_{t-5}x_{t-4}x_{t-3}x_{t-2}x_{t-1}x_t \ldots$ is encoded into $x_{t-5}c_{t-5}x_{t-4}c_{t-4}x_{t-3}c_{t-3}x_{t-2}c_{t-2}x_{t-1}c_{t-1}$ $x_t c_t \ldots$ and received as $x'_{t-5}c'_{t-5}x'_{t-4}c'_{t-4}x'_{t-3}c'_{t-3}x'_{t-2}c'_{t-2}x'_{t-1}c'_{t-1}x'_t c'_t \ldots$ When the decoder shifts in a new branch $\underset{\sim}{v}_1 = x'_t c'_t$, it uses the x'_t in $\underset{\sim}{v}_1$ and the tentatively decoded message digits in $\underset{\sim}{w}$ to encode $\underset{\sim}{w}_1$, from which we obtain $\underset{\sim}{t}_1$ and $\underset{\sim}{t}$. (iii) If the tail part of $\underset{\sim}{t}$ exactly matches one of the stored patterns $\underset{\sim}{t}_{(i)}$, we replace $\underset{\sim}{t}$ and $\underset{\sim}{w}$ by $\underset{\sim}{t} \oplus \underset{\sim}{P}_{(i)}$ and $\underset{\sim}{w} \oplus \underset{\sim}{P}_{(i)}$, respectively. Then go to (iv). Otherwise, just directly go to (iv). (iv) Return to BBO. That is, we shift out the oldest branch from each of the three registers, $\underset{\sim}{v}$, $\underset{\sim}{t}$, and $\underset{\sim}{w}$, and accept the first digit shifted out from $\underset{\sim}{w}$ register to be the decoded message digit. Then return to (ii).

The decoding configuration presented in Figure 4 is intended to explain the principle of operational procedure. In real processing, the implementation can be greatly simplified by hardware designer. For example, one could utilize an AND gate to operate the total function of each pair of $\underset{\sim}{t}_{(i)}$ and $\underset{\sim}{P}_{(i)}$. That is, the corresponding positions of non-zero digits of each $\underset{\sim}{t}_{(i)}$ are fed to the inputs of an AND gate, and the output of the AND gate is fed back to the $\underset{\sim}{t}$ register and the $\underset{\sim}{w}$ purpose of performing the operations of $\underset{\sim}{P}_{(i)} \oplus \underset{\sim}{t}$ and $\underset{\sim}{P}_{(i)} \oplus w$. Figure 5 illustrates a hardware design of the generalized decoder described in Figure 4.

4. DISCUSSION

It is believed that the only class of common codes could be utilized for different decoders is convolutional codes. Since the probability of decoding error decreases exponentially with constraint length K, it is preferable to utilize a long code, especially when extremely low probability of data transmission error is one of the system requirements.

The performance of a system using a long code depends on computational and buffer capabilities of the decoder (i.e., the performance is a function of the average and maximum number of decoding operations required per search).Here, we shall discuss several

different approaches to minimize the decoding effort and thus to achieve a much higher decoding speed, based on both the proposed generalized feedback decoding approach and the special recovery property of convolutional codes [9].

4.1 Utilization of Generalized Feedback Decoding

In previous sections, we have introduced utilizing generalized feedback decoding in small terminals for systems using a common code. This approach can also be utilized in sequential type decoding to accelerate the decoding speed in the following two ways: (i) With very simple hardware, this approach can eliminate all short searches up to the maximum backup distance equal to the search length of the generalized feedback decoder. For example, when the maximum backup distance is 10 branches long, with a small memory device to store 28 $\underset{\sim}{t}_{(i)}$ and 9 $\underset{\sim}{P}_{(i)}$ [6], we can eliminate all the backup searches while a maximum of $(2^{11} - 2)$ operations in branch search is required by sequential type decoding. (ii) If the necessary threshold condition $T^*(b)$ is applied to the backup search procedure, we would be able to minimize the subsearches to a very small range of required backup distances b_t^* [5]. When we carry out subsearches at these backup distances, a common set of $\underset{\sim}{t}_{(i)}$ and $\underset{\sim}{P}_{(i)}$ could be utilized to carry out parallel backup subsearches. Therefore, we could further minimize the decoding effort, buffer size, and complexity requirement.

It has been suggested that multiple stack algorithm (MSA) can achieve lower error probability and higher decoding speed than soft decision Viterbi decoding[7]. One could see that the decoding procedure described above[6] would be simpler and require less decoding effort than MSA, effort in utilizing this decoding approach to simplify MSA would be highly desirable.

4.2 Suggestions for Improving Advanced ARQ Retransmission Systems

Recently, study on computer communication systems with ARQ retransmission techniques has been increased[8]. Most of these studies are limited to either error detection only or detection with simple forward error correction, and because of the high retransmission rate, these systems are quite inefficient[8] . Therefore, it is necessary to utilize powerful error correction code to minimize the required retransmission process.

In general, sequential decoding would be a very powerful error-correction technique. However, the difficulty of employing the usual sequential type decoding in ARQ systems lies in the fact that its Paretian computational distribution problem often causes buffer overflow.

It is fully understood that the Paretian computational distribution problem arises from sequential decoding algorithm, not from the basic properties of convolutional codes, and the recovery property of convolutional codes (the ability to recover to correct path after decoder accepting errors[9]) could be utilized in the following way to overcome the existing difficulties of buffer overflow and to achieve maximum likelihood decoding of using long codes. By carefully selecting the coding parameters (such as employing

systematic code, soft-decision with longer search length), the length of recovery decoding error bursts could be minimized. The recovery decoding error bursts are then detected from the forward decoded sequence, and redecoded by a bidirectional search procedure to achieve a maximum likelihood decoding without the hazard of buffer overflow[10]. Study in this direction to improve advanced ARQ retransmission has been initiated, and results are anticipated in the near future.

REFERENCES

1. Massey, J. L. : Threshold Decoding, The MIT Press, 196Z.

2. Wozencraft, J. M. and Reiffen, B. : Sequential Decoding , The MIT Press, 1961.

3. Ng, W. H., and Pfeiffer, P. E. : "Minirnum-Hamming-distance decoding of single generator binary convolutional codes", Information & Control, October, 1968, pp. 295-315.

4. Wozencraft, J. M. and Jacobs, I. M. : Principles of Communication Engineering, John Wiley & Sons, 1965.

5. Ng, W. H. : "An upper bound on the back-up depth for maximum likelihood decoding of convolutional codes", IEEE Trans. on Information Theory, May 1976, pp. 354-357.

6. Ng, W. H. and Goodman, R. M. F. : "An efficient minimum-distance decoding algorithm for convolutional error-correcting codes", Proc. IEE, February 1978, pp. 97-103.

7. Chevillat, P. R. and Costello, D. J., Jr. : "A multiple stack algorithm for erasurefree decoding of convolutional codes", IEEE Trans. on Communications, December 1977, pp. 1460-1470.

8. Towsley, D. and Wolf, J. K. : "On the statistical analysis of Queue lengths and waiting times for statistical multiplexers with ARQ retransmission schemes", IEEE Trans. on Communications, April 1979, ppl 693-702.

9. Ng, W. H. : "Study on decoding recovery behavior for convolutional codes", IEEE Trans. on Information Theory, Nov. 1970, pp. 795-797.

10. Ng, W. H. : "Bidirectional search for convolutional codes", Proc. IEE, June 1978, pp. 495-500.

APPENDIX

A. Proof of the statement that $\underset{\sim}{t}'$ resulting from replacing $\underset{\sim}{t}_b$ of $\underset{\sim}{t}$ by $\underset{\sim}{t}'_b$, where $|\underset{\sim}{t}'_b| < |\underset{\sim}{t}_b|$, is a minimum test-error pattern.

Since $\underset{\sim}{w}$ is an extension by BBO of an accepted path which had minimum test-error weight, we have $|\underset{\sim}{t}_b| - |\underset{\sim}{t}_1| \leq |\underset{\sim}{t}'_b| - |\underset{\sim}{t}'_1|$.
Now since $|\underset{\sim}{t}_1| = 1$, $|\underset{\sim}{t}'_b| < |\underset{\sim}{t}_b|$ exists only if $|\underset{\sim}{t}_b| - |\underset{\sim}{t}_1| = |\underset{\sim}{t}'_b| - |\underset{\sim}{t}'_1|$ and $|\underset{\sim}{t}'_1| = 0$.
Thus $|\underset{\sim}{t}'_b| = |\underset{\sim}{t}_b| - 1$ and $\underset{\sim}{t}'$ is the minimum test-error pattern.

B. Proof of the necessary condition $T^*(b)$ stated in 3.1.1 that $|\underset{\sim}{t}'_b| < |\underset{\sim}{t}_b|$ only when $|\underset{\sim}{t}_b| \geq T^*(b)$.

a. If $d(b)$ is odd and $d(b-1) < d(b) \leq d(b+1)$, then $T^*(b) = [d(b) + 1]/2$.

From the distance property of the code, we know that $|\underset{\sim}{w}_b \oplus \underset{\sim}{w}'_b| \geq d(b)$ implies $|\underset{\sim}{w}'_b| \geq d(b) - |\underset{\sim}{w}_b|$ and $|\underset{\sim}{t}'_b| \geq d(b) - |\underset{\sim}{t}_b|$. If $|\underset{\sim}{t}_b| \leq T^*(b) = [d(b) - 1]/2$ were true, then $|\underset{\sim}{t}'_b| \geq d(b) - T^*(b) > |\underset{\sim}{t}_b|$ which is a contradiction to $|\underset{\sim}{t}'_b| < |\underset{\sim}{t}_b|$.

b. If $d(b)$ is odd and $d(b) = d(b-1)$, then $T^*(b) = [d(b) + 3]/2$.

We know that $\left[|\underset{\sim}{t}_b| - |\underset{\sim}{t}_1|\right] + \left[|\underset{\sim}{t}'_b| - |\underset{\sim}{t}'_1|\right] \geq d(b-1) = d(b)$; also $|\underset{\sim}{t}_1| = 1$ and $|\underset{\sim}{t}'_1| = 0$ imply $\left[|\underset{\sim}{t}_b| + |\underset{\sim}{t}'_b|\right] \geq d(b) + 1$. Therefore, if $|\underset{\sim}{t}_b| \leq T^*(b) = [d(b) + 1]/2$ were true, it implies $|\underset{\sim}{t}'_b| \geq [d(b) + 1] - |\underset{\sim}{t}_b| = |\underset{\sim}{t}_b|$ which is a contradiction to $|\underset{\sim}{t}'_b| < |\underset{\sim}{t}_b|$.

c. If $d(b)$ is even, then $T^*(b) = [d(b) + 2]/2$.

If $|\underset{\sim}{t}_b| \leq T^*(b) = d(b)/2$ were true, then $|\underset{\sim}{t}'_b| \geq d(b) - |\underset{\sim}{t}_b|$ implies $|\underset{\sim}{t}'_b| \geq |\underset{\sim}{t}_b|$ which is a contradiction to $|\underset{\sim}{t}'_b| < |\underset{\sim}{t}_b|$.

TABLE 1 DISTANCE FUNCTION $d(\cdot)$ AND THRESHOLD CONDITION $T^*(b)$ FOR RATE ONE-HALF CODE

b	$\underset{\sim}{g}$	d (b)	T* (b)
1	1 1	2	2
2	0 1	3	2
3	0 0	3	3
4	0 1	4	3
5	0 0	4	3
6	0 1	5	3
7	0 0	5	4
8	0 1	5	4
9	0 1	6	4
10	0 0	6	4

Table 2 Total $\underset{\sim}{t}_b$ and $\underset{\sim}{t}'_b$ for $b \leq 6$

	$\underset{\sim}{t}_b$						i of $\underset{\sim}{P}_{(i)}$	$\underset{\sim}{t}'_b = \underset{\sim}{t}_b \oplus \underset{\sim}{P}_{(i)}$					
b	6	5	4	3	2	1		6	5	4	3	2	1
2					01	01	1					10	00
5		01	00	10	00	01	2		10	10	00	00	00
5		10	00	10	00	01	2		01	10	00	00	00
5		01	00	01	00	01	3	11	00	00	00	00	00
6	01	00	00	01	00	01	3	10	01	00	00	00	00
6	10	00	00	01	00	01	3	01	01	00	00	00	00

WHERE: $\underset{\sim}{P}_{(1)}$ = 11 01

$\underset{\sim}{P}_{(2)}$ = 11 10 10 00 01

$\underset{\sim}{P}_{(3)}$ = 11 01 00 01 00 01

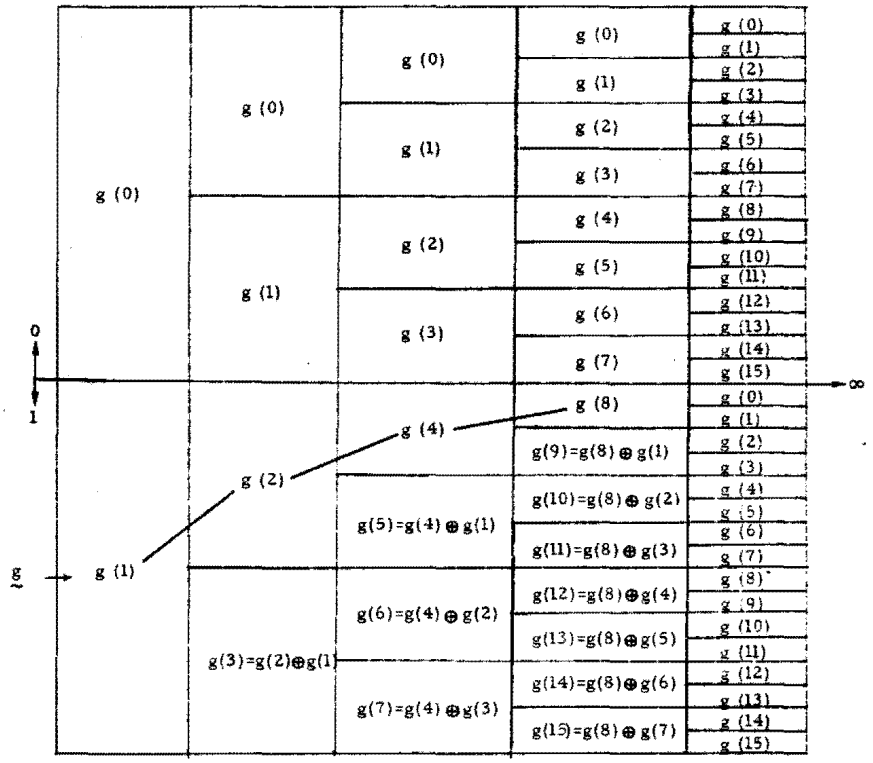

Figure 1 The development of a single-generator initial code tree
where $\underset{\sim}{g} = g(1)g(2)g(4) \cdots g(2^{K-1})$
$= g(1)g(2)g(4)g(8)$

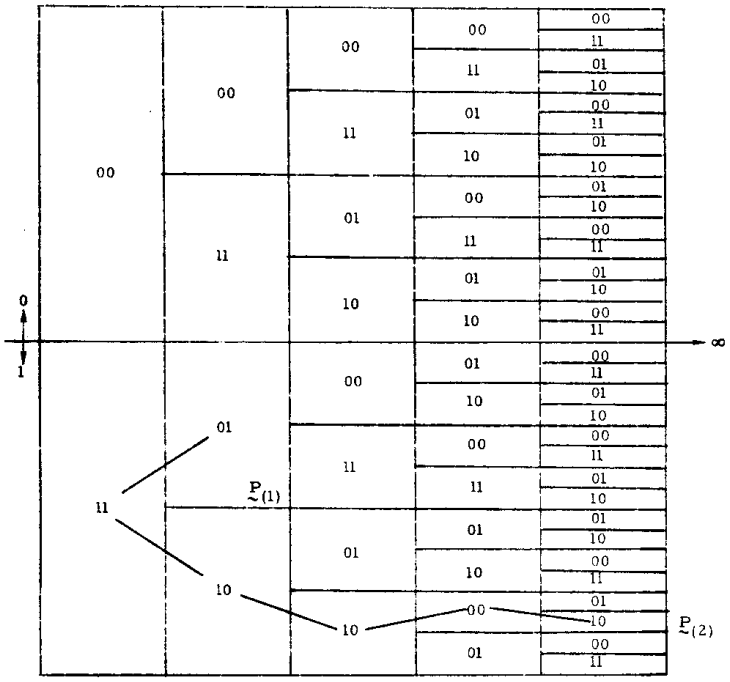

Figure 2 The development of the initial code tree for the half-rate code with $\underset{\sim}{g} = 11\ 01\ 00\ 01\ 00\ 01 \cdots g(2^{K-1})$

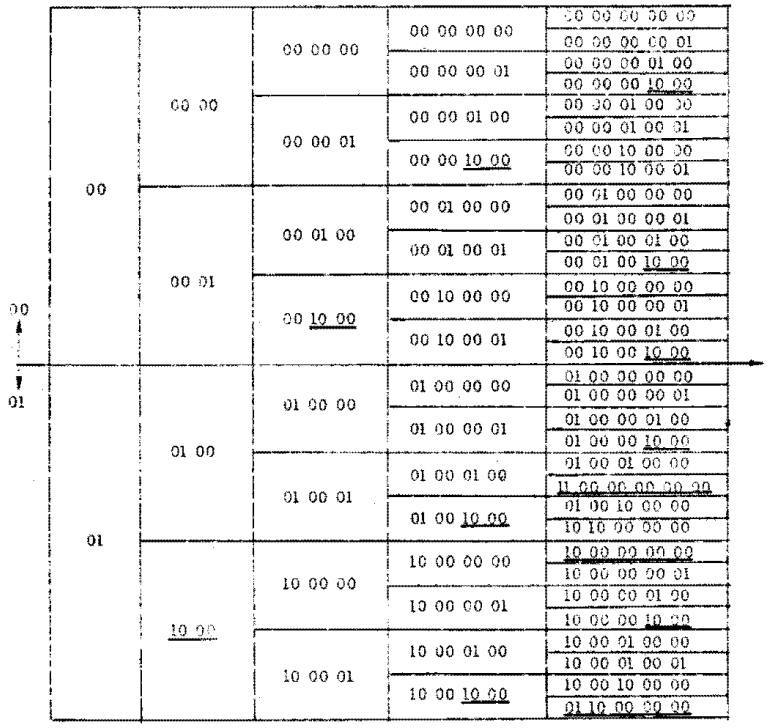

Figure 3 The minimum test-error pattern tree up to 5 branches deep.

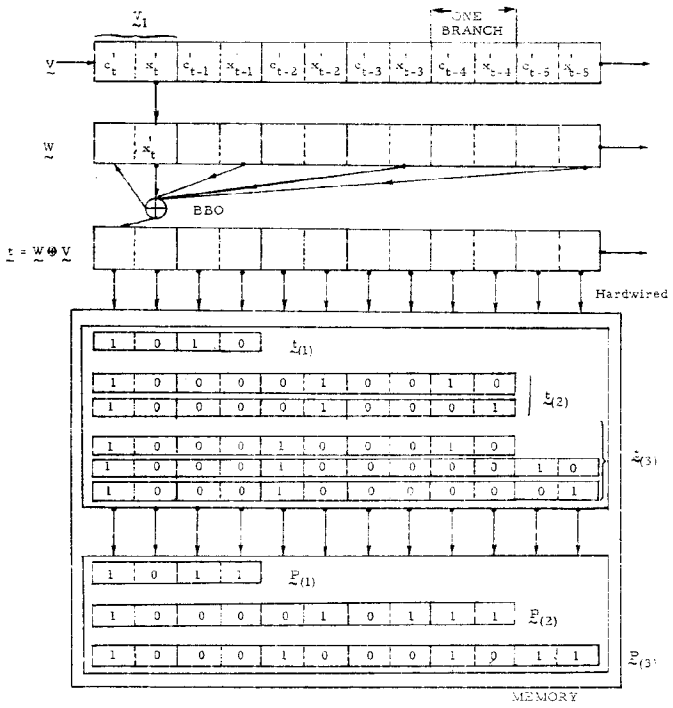

Figure 4 Generalized Feedback Decoding Procedure
when $\underset{\sim}{t}_2$ matches a pattern in $\underset{\sim}{P}_{(i)}$, change $\underset{\sim}{t}$ to $\underset{\sim}{t} \oplus \underset{\sim}{P}_{(i)}$ and $\underset{\sim}{W}$ to $\underset{\sim}{W} \oplus \underset{\sim}{P}_{(i)}$, $1 \le i \le 3$, and return to BBO. Otherwise, directly go to BPO.

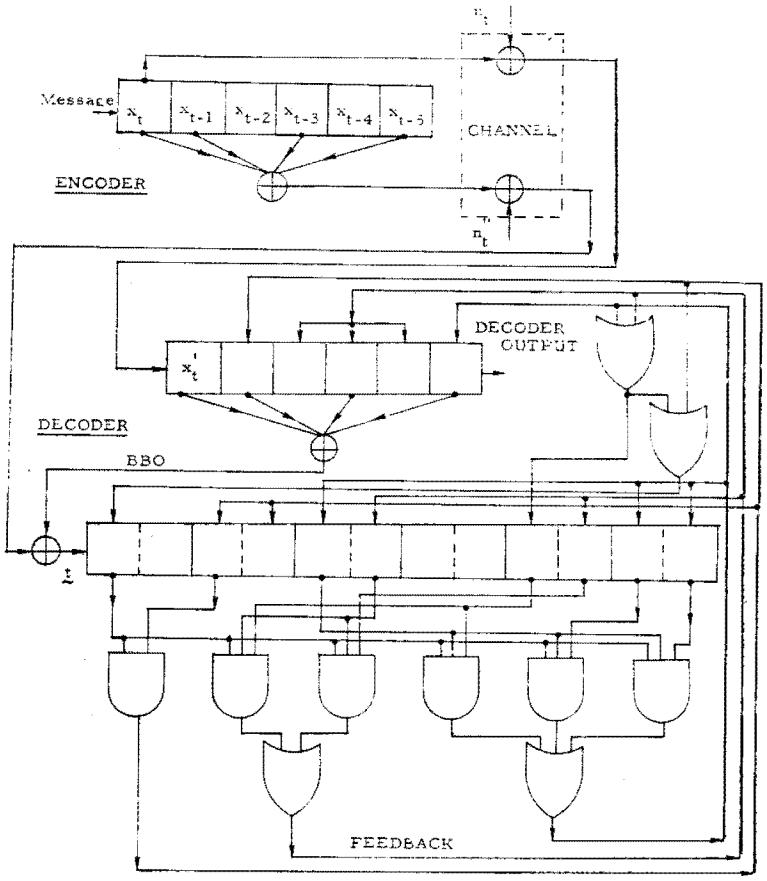

Figure 5 Complete circuitry design for generalized feedback decoder described in Figure 4.