Titre: Title:	Error control techniques by convolutional coding
Auteurs: Authors:	David Haccoun
Date:	1984
Type:	Rapport / Report
Référence: Citation:	Haccoun, D. (1984). Error control techniques by convolutional coding. (Rapport technique n° EP-R-84-10a). https://publications.polymtl.ca/9566/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/9566/
Version:	Version officielle de l'éditeur / Published version
Conditions d'utilisation: Terms of Use:	Tous droits réservés / All rights reserved

Institution:	École Polytechnique de Montréal
Numéro de rapport: Report number:	EP-R-84-10a
URL officiel: Official URL:	
Mention légale: Legal notice:	

ERROR CONTROL TECHNIQUES BY CONVOLUTIONAL CODING*

by

David Haccoun, Ph.D.
École Polytechnique de Montréal

April 1984

* This research was supported in part by the Natural Sciences and Engineering Research Council of Canada, under Grant No. A4966.

ERROR CONTROL TECHNIQUES BY CONVOLUTIONAL CODING

by

David Haccoun

ABSTRACT

This paper presents the error control technique of convolutional coding, and some of its most powerful methods of decoding, especially sequential decoding and some of the new developments of the basic stack algorithm. These developments and variants attempt to circumvent some of the severe shortcomings of sequential decoding (i.e. computational variability and storage requirements), and open new possibilities of applications.

1. INTRODUCTION

In a digital communication system, information or data originated at the source must be transmitted to a distant user through a noisy channel. Because of the channel noise, the transmitted signals do not arrive at the receiver exactly as transmitted and hence errors are made in conveying the source message to the user. The performance for a digital communication system is the probability of error stated in terms of the probability of message, word, or bit error. The choice of the most appropriate unit will depend mainly on the application of the system. Whenever the data is organized in blocks, then a good measure for the system's performance will be the probability of block error; on the other hand, if the information to be sent is a continuous stream of data, then the probability of bit error might be the appropriate measure of performance. In certain applications such as computer-to-computer communication, a repeat for the erroneous block of information is requested through a feedback channel whenever an error is detected. For these systems, the performance criterion is the probability of undetected block error which is required to be very small, often smaller than 10^{-10}.

Additional requirements on the data itself may concern the tolerable delay in the processing and delivering of the data. Although some systems such as deep space telemetry can tolerate delays of several days in processing the data, many communication systems demand essentially real-time processing. The requirements of data accuracy, data rates and delays together with systems constraints on available transmitting power and bandwidth are key factors to be considered in the implementation of the system. A good design consists often in making sensible trade offs between complexity and performance in the most economical manner.

Error control techniques are usually divided into three types, namely, forward error control (FEC) schemes, error detection with retransmission (referred to as ARQ or automatic repeat request), and hybrid systems that employ both FEC and ARQ techniques. There are two large classes of FEC coding techniques, namely block coding and convolutional coding [1]. Block coding is used when the digital information is organized in distinct blocks, whereas convolutional coding is particularly suitable when the digital information to be transmitted arrive serially in long sequences rather than in blocks. Block coding is treated at length in [2]. In this paper, we will only examine convolutional coding, especially some of the most promising decoding techniques for convolutional codes.

2. BASIC MODEL

In a digital communication system employing FEC techniques such as shown in Figure 1, the data source generates binary information symbols at the rate R_s bits/s. These information symbols are encoded for error protection purposes and the encoder output is another binary sequence of rate Rc symbols/s. The code rate R in bits/symbols is given by the ratio $R = R_s/R_c$. Since R_c is larger than R_s, then over the channel the transmission speed is higher than the data rate delivered by the source. Equivalently, the introduction of an error control coding requires a bandwidth expansion.

At the receiver, let the received power be P and let N_o be the spectral density of the channel noise. Then the signal-to-noise ratio per information bit is given by $E_b/N_o = P/(N_o R_s)$. This E_b/N_o ratio serves as a figure of merit for different combinations of coding and modulation schemes. It is clear that a coding or modulation scheme which reduces the E_b/N_o required for a given error probability, leads to an increase in the allowable data rate and/or a decrease in the necessary transmission power. The basic problem is thus the determining of a system that will operate at the lowest E_b/N_o with a given error performance.

Figure 2 shows the performance curves of several coding schemes together with that of the perfect coherent PSK modulation. The coding gain of any coding scheme is measured as the difference in E_b/N_o between that coding scheme and PSK for a given error probability. For example, at a bit error probability of 10^{-5}, the block coding BCH (128,112) provides a 2 dB coding gain whereas soft decision Viterbi decoding (K=7, R=½) provides over 5 dB, and hard quantized sequential decoding can provide 5.2 dB. This 5.2 dB coding gain can be translated as either a 5.2 dB reduction in the transmitting power for the same data rate and error performance as the PSK system, or as an increase data rate equal to about 3.3 times the uncoded data rate. Depending on the application, both of these alternatives may be quite attractive in improving a system design.

For well-behaved channels, such as the space channel, systems using convolutional encoding and Viterbi or sequential decoding are among the most powerful and the most interesting : they provide substantial coding gains while being readily implementable. For convolutional codes, there also exist much simpler decoding schemes such as threshold decoding which provide more modest coding gains but which are far simpler to implement. For example, the rate 3/4 threshold decoder provides a coding gain of 1.2 dB. For satellite transmission of data over regular voice channels and SCPC systems, this relatively simple coding scheme is quite suitable and is widely applied [3,4,5].

We now briefly introduce convolutional coding and present the powerful Viterbi and sequential decoding techniques together with recent developments and extensions.

3. CONVOLUTIONAL ENCODING

A binary convolutional code of rate 1/v may be generated by a linear finite-state machine consisting of a K-stage shift register, V modulo-2 adders connected to some of the shift register stages, and a commutator that scans the output of the modulo-2 adders. The set of connexions between the mod-2 adders and the shift register specifies the code. An example of K=3, rate ½ convolutional encoder is given in Figure 3.

Information bits are shifted in at the left, and following each shift, the modulo-2 adders are sampled in sequence by the commutator, providing 2 code symbols that are transmitted through the channel. Hence, a coding rate R=½. For such simple encoders, the length of the shift register K is called the constraint length of the code, and the rate is 1/v. One can generalize the binary convolutional encoder by allowing more than one bit at a time to enter the encoder. If u bits are shifted in at a time, then the coding rate is u/v, and the constraint length is taken to be a multiple of the integer u, say K=ku.

Tree and Trellis Structure

Convolutional codes exhibit a tree and trellis structure that are essential to illustrate the powerful decoding methods of Viterbi and sequential decoding [4]. Consider a rate ½ encoder. The fact that at any time the input to the encoder may take 2 values, suggests representing the output of the encoder by a binary tree. The tree has two branches per node, each having v coded symbols corresponding to either a "zero" or a "one" input bit. Hence, an ℓ-bit input sequence that entered the encoder will trace a specific path in the encoded tree, and the corresponding code symbols on the branches of the path are the symbols transmitted through the channel. For example, the encoder of Figure 3 is described by the tree of Figure 4, and the input sequence 1011 traces a path with a coded sequence 11100001.

The state of an encoder is the contents of the first (K-1) positions of the encoder shift register, and the V symbols corresponding to a given input bit are specified by that input bit and the state of the encoder. Hence to each node of the tree, there corresponds an encoder state; there are 2^{K-1} possible distinct states. Hence, whenever the input sequence is longer than (K-1) bits, there are more nodes in the tree than states; that is several nodes correspond to the same encoder state and are thus identical. The tree contains redundant information which can be eliminated by merging together, at any same tree depth, all nodes corresponding to the same encoder state. The redrawing of the tree with merging paths and redundancy eliminated is called a trellis. Tree and trellis structures carry the same information, and an input sequence will trace the same path in either the tree or the trellis. Figure 5 is the trellis corresponding to the tree of Figure 4. For general rate U/V codes, the same principles apply, but instead of two branches per node, tree and trellis will have 2^U branches emerging from each node.

4. DECODING FOR CONVOLUTIONAL CODES

Decoding may be seen as the operation for determining the most likely information sequence given the received sequence. This received sequence is the transmitted sequence with possibly several symbols in error. The suboptimum tree search of sequential decoding and the optimal trellis search of Viterbi decoding are two techniques which attempt to find the most likely information sequence, or "best" path through a graph (tree or trellis) in which the branches are assigned likelihood or "metric" values.

Viterbi and sequential decoding have developed independently and appear to be the opposite for determining the most likely information sequence given the received sequence. The Viterbi algorithm uses the trellis structure of the code and examines all distinct paths at every trellis level, whereas a sequential decoder uses the tree structure of the code and follows only that path in the tree that appears to be the most likely. As a consequence, the computational effort is constant but large for Viterbi decoding, whereas it is on the average typically very small but variable for sequential decoding.

We now examine each of these techniques, emphasizing sequential decoding and some of its variants.

VITERBI DECODING :

The Viterbi decoding algorithm is a simple decoding procedure which determines the path having the largest accumulated metric of all possible distinct paths. It uses the trellis structure of the code and retains at each depth only the best path that terminates at each of the 2^{K-1} states. At each decoding step, the 2^{K-1} remaining or "surviving" paths are extended, and their total metrics compared pair-wise so that for each 2 paths merging at each state, only the path with the largest metric is retained. With this procedure, none of the discarded paths can ever be the most likely path, that is the procedure is optimum [4,6].

The error performance of Viterbi decoding decreases exponentially with the constraint length K of the code. Hence, it may be desirable to use a code with K as large as possible. However, since the Viterbi decoding operations are identical from level to level, and since they must be performed at every state, the complexity of the decoder and the number of computations per decoded bit grow exponentially with the constraint length of the code. This exponential growth limits practical Viberbi decoding to short constraint length codes ($K < 8$).

Viterbi decoding is widely used over a variety of channels, and can provide substantial coding gains, exceeding 5 dB at an error probability of 10^{-5}. It can easily operate on soft-decisioned data, providing an additional 2dB gain. Typical error probability curves are given in Figure 6, indicating approximately 0.5 dB of coding gain for each unit increase of the constraint length K. Although loss of synchronization of the decoder entails a gradual performance degradation, over bursty channels Viterbi decoding does not perform very well. In those channels interleaving of the data may have to be considered to decorrelate consecutive noise samples. However, the ensuing delay may not be acceptable in certain applications.

Finally, we should mention that Viterbi decoding is a very mature technique, widely used, with Viterbi decoders readily available for a variety of rates and a variety of decoding speeds, which may run in the tens of Megabits/-second [7].

SEQUENTIAL DECODING :

Whenever the desired performance requires using codes with large coding gain, hence large values of K, a somewhat suboptimal decoding technique must be used instead of Viterbi decoding. One such technique is sequential decoding [3].

The central idea of sequential decoding is the decoding of the received message one branch at a time, without searching the entire tree. Starting from the origin of the tree, the path selected to be explored one step further is the path whose metric is the largest among those already examined. The path that first reaches the end of the tree with the highest metric is accepted as the decoded path. As the decoding proceeds, the decoder occasionally retreats in the tree and extends earlier and possibly incorrect paths. In order to minimize this backing-up and extension of unlikely paths, the metric is biased in such a way that on the average it increases along the correct path and decreases along all incorrect paths [8].

There are two main sequential decoding algorithms : the Fano algorithm [9] and the Zigangirov-Jelinek (Z-J) algorithm [10]. In this paper, we will consider only the Z-J algorithm and some of its variants and extensions.

In the Z-J or stack algorithm, the decoder consists of a list or stack of the already searched paths, ordered in decreasing order of their metric values. The "top" of the stack has the largest accumulated metric and will be searched further, i.e. extended one level further along both branches emerging from its end node. The operations of the stack decoder are thus the finding of the top node, the extension and storage of its successors, and the proper reordering of the stack. As a node is extended, it is removed from the stack.

The algorithm is then :

1) Compute the metrics of the successors of the top node and enter them in their proper place in the stack.
2) Remove from the stack this top node that was just extended.
3) Find the new top node. If it is the final node, stop. Otherwise go to 1).

Sequential decoding involves a random motion in the tree, leading to a variable number of computations to decode a given block of information. The number of computations necessary to decode one information symbol has a Pareto distribution, that is a distribution whose tail decreases only algebraically [11].

Figure 7 shows a typical distribution curve. This variability of the computational effort constitutes one of the main drawbacks of sequential decoding. However, provided the coding rate R is smaller than some rate R_{comp} ,typically the average number of computations per decoded information bit, C_{AV}, is very small, much smaller than the constant number 2^{K-1} required for Viterbi decoding. At rates beyond R_{comp}, C_{AV} becomes unbounded; R_{comp} is called the computational cut-off rate and constitutes the limiting rate of operation for sequential decoders. Furthermore, unlike Viterbi decoding, this average decoding effort of sequential decoding is practically independent of the constraint length of the code which may thus be chosen at will. Therefore, sequential decoding tends to be used in conjunction with long constraint length codes, $K \geqslant 20$. However, because of the variable decoding delay, the analysis of sequential decoding is concerned not only with the error probability but also with the distribution of the computational effort.

In practical sequential decoders, the variable decoding delay is accommodated by the use of input and output buffers. The output buffer smoothes out the output rate of the decoded data, whereas the input buffer is used to store the incoming data waiting to be decoded. Regardless of the size of the buffer, there is a nonzero probability that it may fill-up, leading to an overflow and possibly, communication breakdown. This overflow problem as well as the restarting procedures constitute one of the most serious problems of sequential decoding. To overcome the difficulty of overflow, data are organized in blocks, usually on the order of 500 to 1000 branches, and known sequences called the "tail of the message" clear the shift register of the convolutional encoder after each block and resynchronize the system. In case of an overflow, the buffer clears the overflowed block and decoding can be resumed in the following block. A simplified block diagram of a sequential decoder using the stack algorithm is given in Figure 8.

5. VARIANTS OF SEQUENTIAL DECODING

Some new methods have been proposed to alleviate the computational difficulty of sequential decoding [12,13]. These methods are all based on the Z-J algorithm. First the concept of the Z-J algorithm can be generalized to allow a multiple-path extension instead of a single path extension. Furthermore, one can eliminate from the stack any undesired path, and therefore reduce the memory requirement of the stack. Finally, consideration has also been given to the decoding of high rate codes, such as rate 2/3 or rate 3/4 codes which are especially important in applications where the bandwidth is somewhat restricted. In these codes $2^2 = 4$ and $2^3 = 8$ branches emerge from each tree node respectively, overtaxing both the decoding speed and memory requirements of a straightforward sequential decoder[14,15]. We shall first consider the reduction of the computational variability by multiple path extensions.

5.1 Multiple-path Stack Algorithms :

This is a simple variation of the basic Z-J stack algorithm which consists in extending simultaneously several paths from the top of the stack [12,13].

Furthermore, remerging paths are exploited as in the Viterbi algorithm in order to eliminate redundant paths from the stack, and reduce the stack storage. With these modifications, it is shown that the variability of the computational distribution is reduced compared to that of the ordinary stack algorithm at a cost of a somewhat larger average number of computations. However, the error probability is improved and approaches closely that of the optimum Viterbi decoding. One could even modify at will the number of extended paths and adapt it to the needs imposed by the channel noise. When the channel is quiet, extend only one or two paths simultaneously, and whenever there is some burst of noise and the metric drops, extend a larger number of paths. The resulting "adaptive" sequential decoding yields an even smaller computational variability with a smaller average number of computations than the fixed multiple-path algorithm [16]. Distributions of computation obtained by simulation are given in Figure 9, showing the improvements obtained in both variability and average value of the computational effort.

By considering a decoding cycle as consisting of a path extension cycle (which extends some number of paths according to an extension rule), and a purging cycle (which eliminates from the stack any undesirable path according to a purging rule), then all the above algorithms, including the usual stack algorithm and the Viterbi algorithm are special cases of the "generalized stack algorithm" [12]. This generalized stack algorithm in essence closes the gap between the single-path sequential decoding and the all-path Viterbi algorithm.

Finally, another variant of the basic stack algorithm, called multiple stack algorithm has been recently proposed to circumvent the buffer overflow problem [17]. It uses several stacks, which are filled one after another, but in such a way that decoding is always completed, practically eliminating overflows, but at the expense of a substantial increase in stack and buffer storage.

5.2 High Rate Sequential Decoding :

As mentioned earlier, another variant of the basic stack algorithm of sequential decoding concerns the use of codes with coding rates $R = U/V$, rather than $1/V$. These codes require a bandwidth expansion of only V/U instead of V, and hence are of special importance when the available channel bandwidth is restricted.

We recall that for rate $1/V$ codes, there are two branches per node in the tree or trellis representation of the code. For binary U/V codes, there are 2^U branches stemming out of each node, and since an optimal decoder must examine them all, the complexity of a Viterbi decoder becomes further aggravated when used with these codes. On the other hand, since a sequential decoder explores only a fraction of the tree paths, then not all the 2^U branch extensions may need to be examined when exploring a tree path. Using the notion of a threshold to discard some of the least likely among the 2^U branch extensions, it has been shown that both the average number of computations and stack size of the decoder may be significantly reduced without impairing the error performance [15]. With such a modification of the stack algorithm, sequential decoding may thus present an attractive alternative to Viterbi decoding or to any other decoding scheme for rate U/V codes.

Threshold Determination

The basic idea of using a discarding threshold for the decoding of rate U/V codes is to obtain substantial savings in the required stack memory of the decoder at a minimal cost in error performance and computational effort. After extending a node into all of its 2^U successors, the metric of each of these branch extensions is compared to the threshold value. Only those extensions whose metric exceed the threshold value are stored in the stack; the others are discarded. Hence, in setting up the threshold value, the underlying desired property is that any discarded node be very unlikely a part of the correct path.

The determination of the threshold value depends on the code and the channel, which may be hard or soft quantized. For soft quantized channels, each symbol metric may take many values, and thus the number of different branch metric values is quite large, spanning a wide range. Each of these metric values occurs with a given probability.

Therefore, in such channels, one can set a threshold value that will be exceeded with some given probability. From simulations with rate 2/3 and rate 3/4 codes and 3-bit quantized channels, it was established that the probability of not exceeding the threshold to be between 10^{-4} and 10^{-6}.

As for hard quantized channels (the binary symmetric channels), the problem is considerably simplified because, due to special features of the codes suitable for sequential decoding, the number of different branch metrics turns out to be very small. For example, as shown in Figure 10, for rate 2/3 codes, there are only two error patterns possible for each node extensions. For every set of 4-branch extensions emerging from each node, there can be either one branch with no error, together with 3 branches with 2 errors (Pattern I), or 3 branches with a single error, together with one branch with 3 errors (Pattern II).

Consequently, for rate 2/3 codes, there are only 3 possible threshold values corresponding to discarding either single, double or triple-error patterns. An obvious choice for the threshold is to set it to a value which corresponds to discarding all 2 and 3 error patterns on a branch.

Similarly, for the rate 3/4 codes used in our simulation, there are only 2 possible error patterns corresponding to either 1 error-free branch, 6 double-error branches and 1 four-error branch (Pattern I), or 4 single-error branches together with 4 three-error branches (Pattern II). As shown in Figure 11, a reasonable choice for the threshold is the value which corresponds to the discarding of all branches bearing two or more errors.

Computer simulation results with the rate 2/3 and 3/4 codes for both hard and soft quantization have shown that very substantial savings (over 50% and up to 90%) in the required stack memory of the decoder can be achieved at practically no degradation of the error performance [15]. Furthermore, the variability of the computational effort remains unchanged compared to a straightforward sequential decoder, indicating that all the discarded branch extensions were indeed unnecessary for the decoding process. An example of results obtained for rate 3/4 codes over the BSC is shown in Figure 12.

Therefore, by a judicious choice of a discarding threshold that eliminates unnecessary paths that a standard sequential decoder must otherwise examine, large savings in the decoder memory may be achieved at hardly any cost in error performance or computational variability. Since the decoding process is practically independent of the constraint length of the code, such a variant of the stack algorithm may make sequential decoding a very attractive and practical alternative for the decoding of high rate codes over very noisy channels.

INTERSYMBOL INTERFERENCE

Sequential decoding, in particular the multiple-path stack algorithm was recently applied to a problem somewhat different from coding, that is the resolution of intersymbol interference problem in bandwith-constrained channels [18,19].

When transmitting high speed digital data over band limited channels, in addition to the ever present channel noise, a major impairment is the intersymbol interference (ISI) between neighboring pulses due to an insufficient channel bandwidth.

Although effective channel equalization techniques have been used to minimize the effects of ISI, these techniques are not optimum. A major breakthrough was achieved by considering ISI as a convolutional encoding of the data, where the memory of the channel is associated with the memory of the convolutional encoder. Hence, optimal decoding techniques for convolutional codes, i.e. Viterbi decoding, can be directly applied to the ISI problem. However, the computational complexity of the Viterbi decoding algorithm grows linearly with the length of the transmitted sequence, and also grows exponentially with the memory of the channel. Hence, its application is limited to channels with small memory, that is with an ISI extending only over very few pulses (typically less than 3) [20].

For long memory channels, since Viterbi decoding becomes impractical due to the excessive amount of computations required, an alternative approach is to use sequential decoding where the computational complexity, especially the average decoding effort is independent of the memory of the code. Hence, it can be applied to combat ISI over long memory channels. As for the computational variability, this drawback can be circumvented by the multiple-path stack algorithm [18,19].

Simulation results with channels having memory lengths up to 6 symbols have shown that the performance of the 7-path sequential decoding fall within the theoretical upper and lower performance bounds of an optimal decoder (see Fig. 13). As for the overall decoding effort, it is only a very small fraction of the decoding effort that would be required of an optimum Viterbi decoder. Furthermore, for the high signal to noise ratios of interest (> 15 dB), the results show hardly any computational variability. Consequently, a much simplified form of a multiple path sequential decoder structure appears possible, making the procedure a very attractive alternative to existing ISI methods. This simplified form of the decoder is presently under investigation [21].

6. CONCLUSIONS

We have presented the error control technique of convolutional coding and some of its important methods of decoding, especially Viberbi and Sequential decoding. In particular, we have presented some of the new developments of the basic stack algorithm of sequential decoding. These developments and variants attempt to circumvent some of the computational difficulties of sequential decoding, and to reduce the required storage in practical implementations. These new developments make sequential decoding techniques attractive alternatives and open new domains of applications.

REFERENCES

[1] V.K. BHARGAVA, "Forward Error Correction Schemes for Digital Communications", IEEE Communications Magazine, pp. 11-19, Jan. 1983.

[2] E.R. BERLEKAMP, "The Technology of Error-Correcting Codes", Proc. IEEE, Vol. 68, pp. 564-593, May 1980.

[3] J.L. MASSEY, "Threshold Decoding", MIT Press, Mass. 1983.

[4] V.K. BHARGAVA, D. HACCOUN, R. MATYAS, P. NUSPL, "Digital Communications by Satellite", John Wiley, New York, 1981.

[5] A. SEWARDS, L. BEAUDET, H. AHMED, "Forward Error-Correction for the Aeronautical Satellite Communications Channel", CRC Note No. 698, Depart. Communications, Ottawa, 1979.

[6] A.J. VITERBI, "Convolutional Codes and their Performance in Communication Systems", IEEE Trans. Commun. Technol., Vol. COM-19, Oct. 1971.

[7] I.M. JACOBS, "Practical Applications of Coding", IEEE Trans. Information Theory, Vol. IT-20, May 1974.

[8] D. HACCOUN, "A Markov Chain Analysis of the Sequential Decoding Metric", IEEE Trans. on Inf. Th., Vol. IT-26, Jan. 1980.

[9] R.M. FANO, "A Heuristic Discussion of Probabilistic Decoding", IEEE Trans. Inf. Theory, Vol. IT-9, April 1962.

[10] F. JELINEK, "A Fast Sequential Decoding Algorithm Using a Stack", IBM Journal of R & D, Nov. 1969.

[11] I.M. JACOBS and E.R. BERLEKAMP, "A Lower Bound to the Distribution of Computation for Sequential Decoding", IEEE Trans. on Inf. Th., Vol. IT-13, April 1967.

[12] D. HACCOUN and M.J. FERGUSON, "Generalized Stack Algorithms for Decoding Convolutional Codes", IEEE Trans. on Inf. Th., Vol. IT-21, pp. 638-651, Nov. 1975.

[13] D. HACCOUN, M. DUFOUR, "Stack and Input Buffers Overflow of Stack Decoding Algorithms", IEEE Intern. Symposium on Information Theory, Santa Monica, Feb. 1981.

[14] D. HACCOUN, CHEN NAIYUN, "Empirical Investigation of High Rate Sequential Decoding", IEEE International Symposium on Information Theory, Les Arcs, France, June 1982.

[15] D. HACCOUN, CHEN NAIYUN, "Variants of the Stack Algorithm for the Decoding of High-Rate Codes by Sequential Decoding", Satellite Communications Conference, Ottawa, pp 21.4.1-21.4.5, June 1983.

[16] D. HACCOUN, A. JANELLE, "Adaptive Sequential Decoding by Stack Algorithms", 22nd Midwest Symposium on Circuits and Systems, Philadelphia, pp. 171-175, June 1979.

[17] P.R. CHEVILLAT and D.J. COSTELLO, "A Multiple Stack Algorithm for Erasurefree Decoding of Convolutional Codes", IEEE Trans. on Comm., Vol. COM-25, Dec. 1977.

[18] D. HACCOUN, P. HAURIE, "Techniques de décodage séquentiel appliquées au problème de l'interférence entre symboles", Rapport technique EP-82-R-5, fév. 1982, Ecole Polytechnique de Montréal, 125 pages.

[19] D. HACCOUN, P. HAURIE, "Application of Multiple-Path Sequential Decoding to the Intersymbol Interference Problem", IEEE International Symposium on Information Theory, Les Arcs, France, June 1982.

[20] S. CROZIER, M. WILSON, W. MORELAND, J. CAMELON, P. McLANE, "Microprocessor Based Implementation and Testing of a Simple Viterbi Detector", Can. Elect. Eng. Journal, Vol. 6, No. 3, pp. 308, Jan. 1981.

[21] S. KALLEL, D. HACCOUN, "Simplified Multiple-Path Sequential Decoding for Intersymbol Interference Problems", Technical Report, Ecole Polytechnique, in preparation.

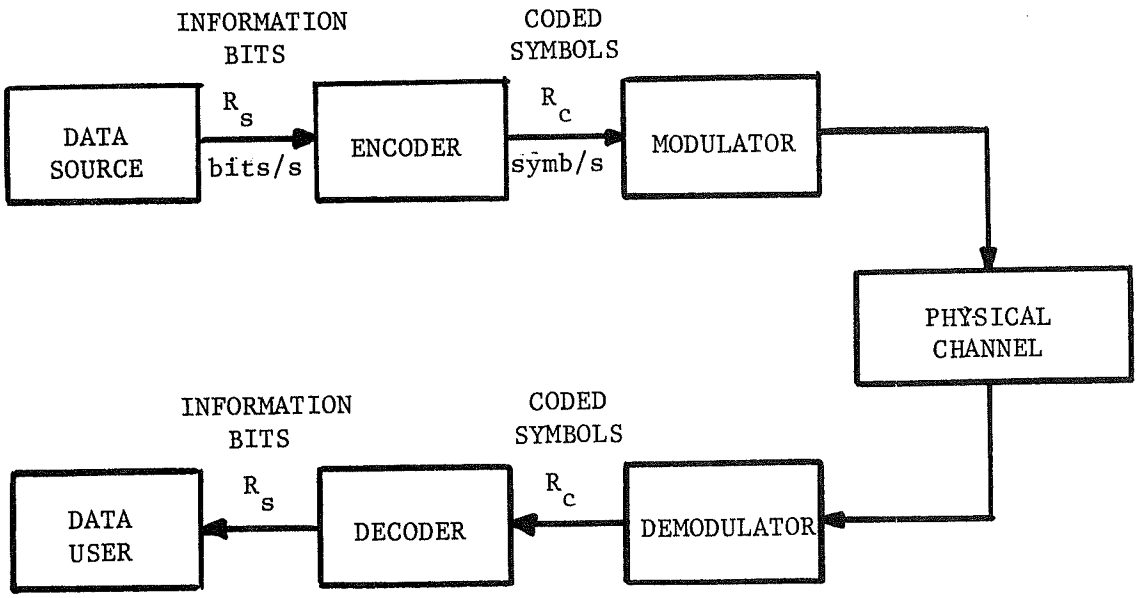

Figure 1: Model of Digital Communication System Using FEC.

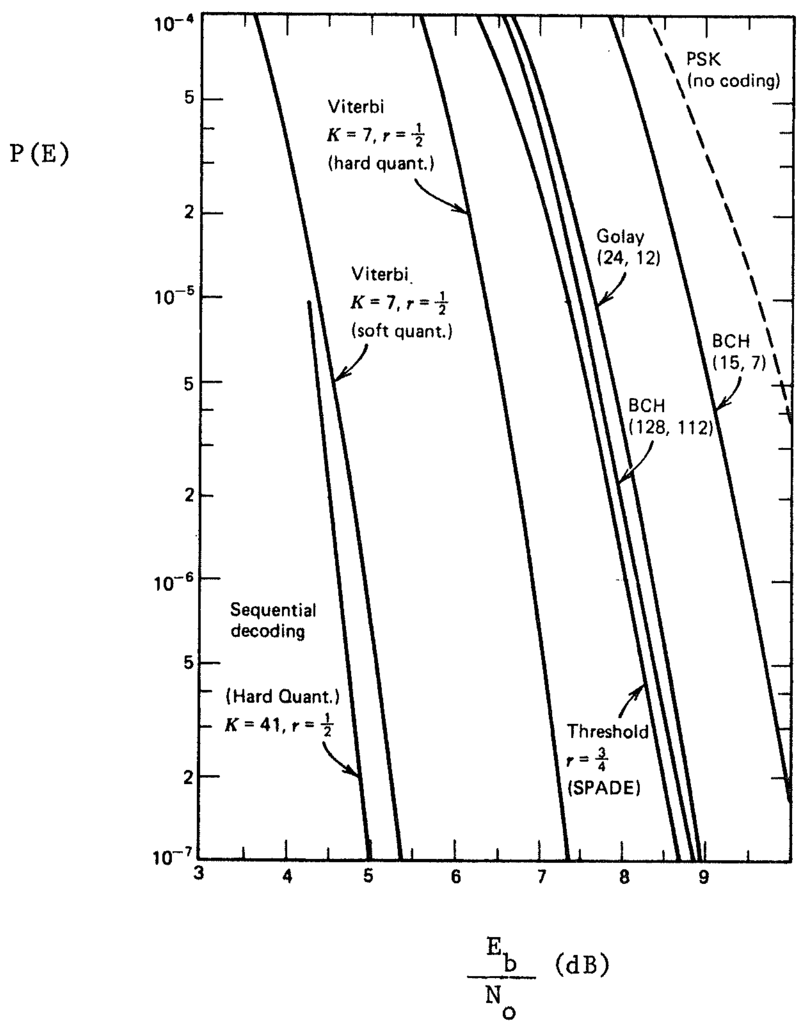

Figure 2: Performance curves of several coded systems.

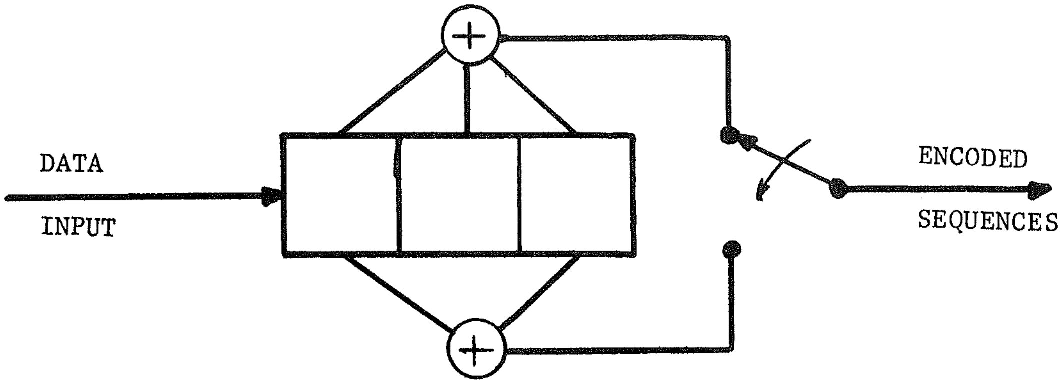

Figure 3: Convolutional encoder K=3, R=1/2.

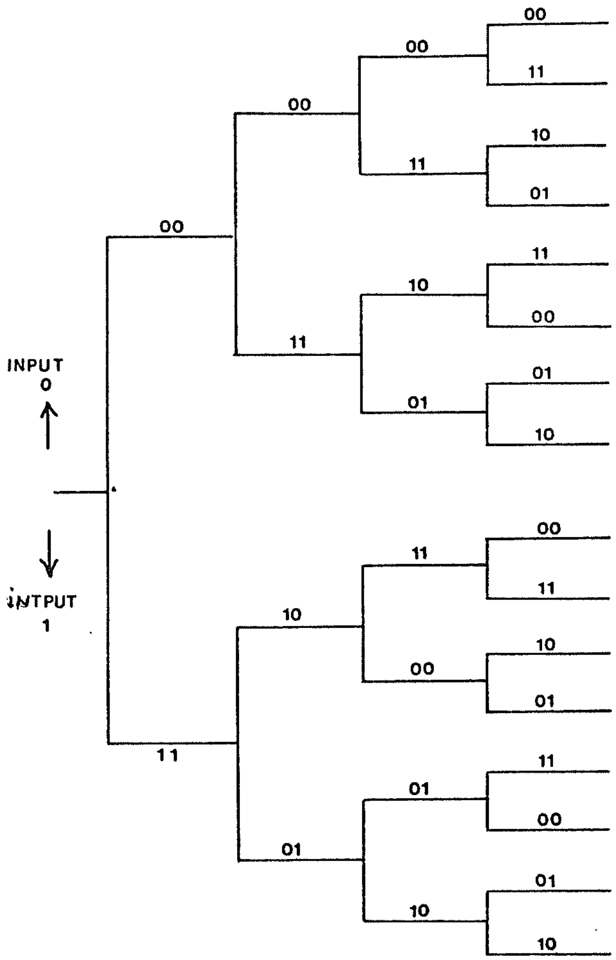

Figure 4: Tree corresponding to encoder of Fig. 3.

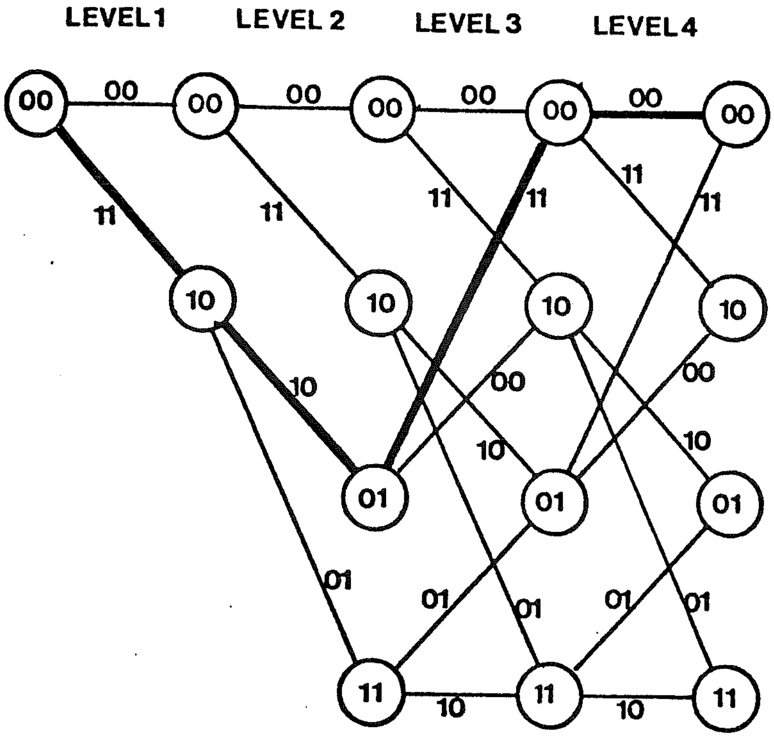

Figure 5: Trellis corresponding to tree of Fig. 4.

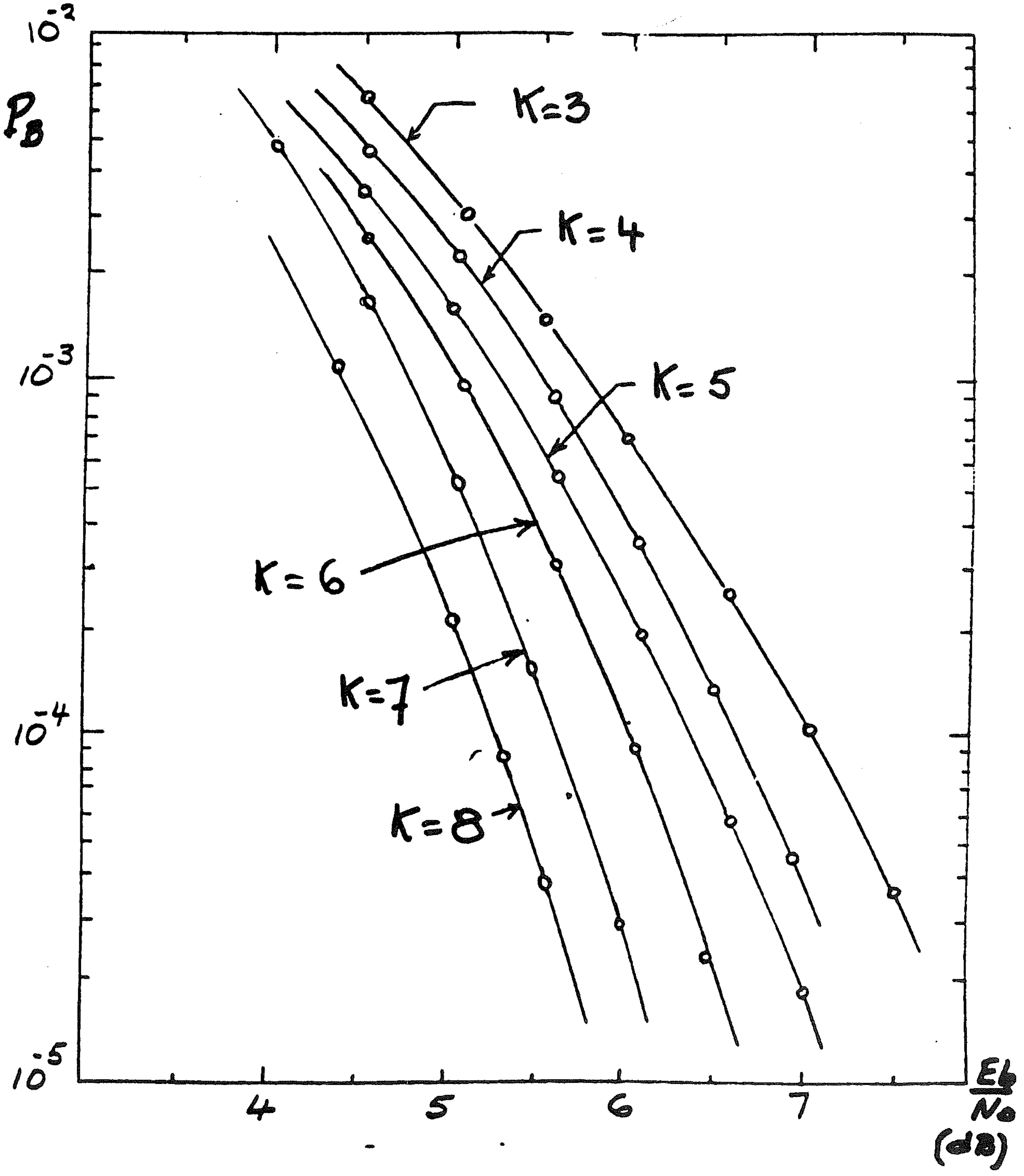

Figure 6: Performance curves for rate $\frac{1}{2}$ Viterbi decoding over BSC channel.

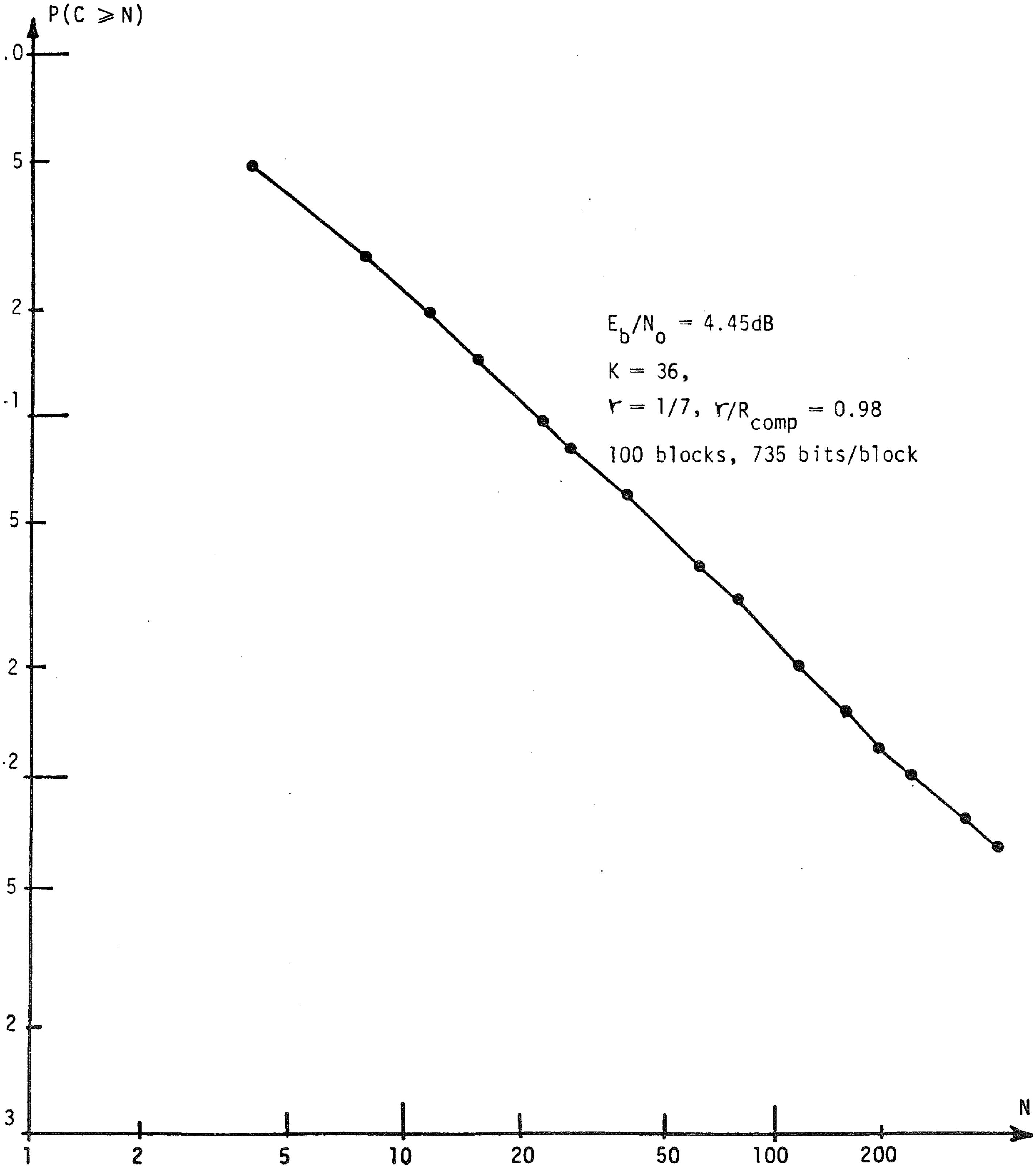

Figure 7: Distribution of the computational effort of sequential decoding.

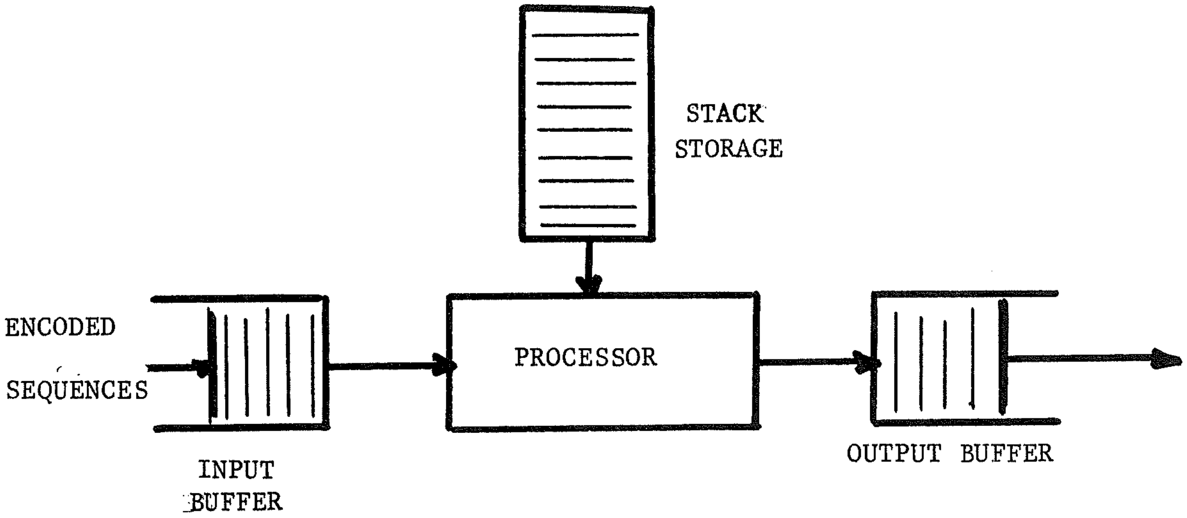

Figure 8: Block Diagram of a Sequential Decoder Using the Stack Algorithm.

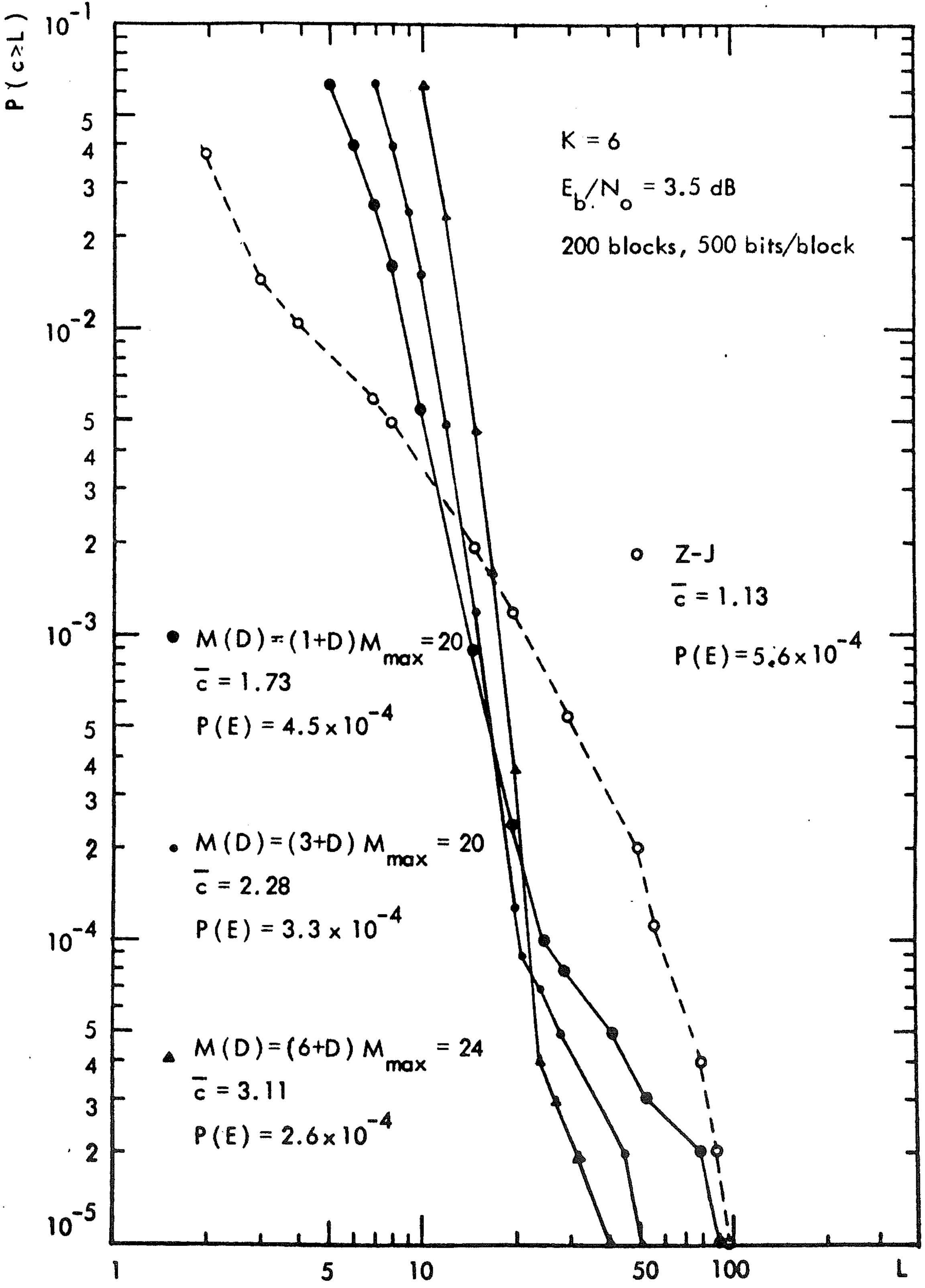

Figure 9 Empirical distribution of computations per search for the Adaptive algorithm.

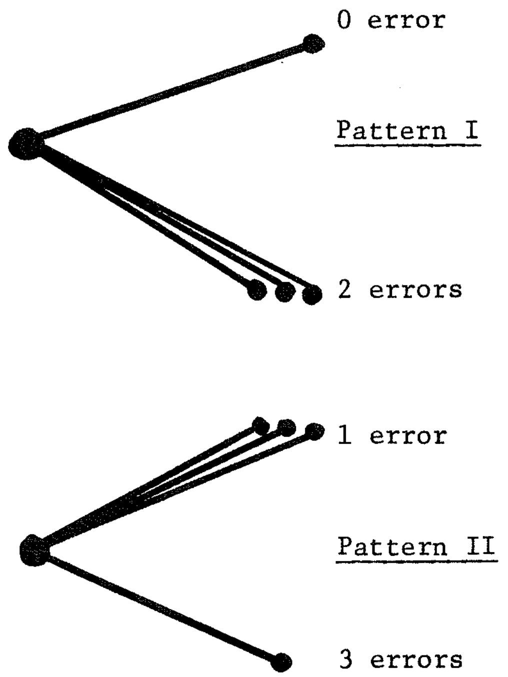

Fig. 10 Rate 2/3 code

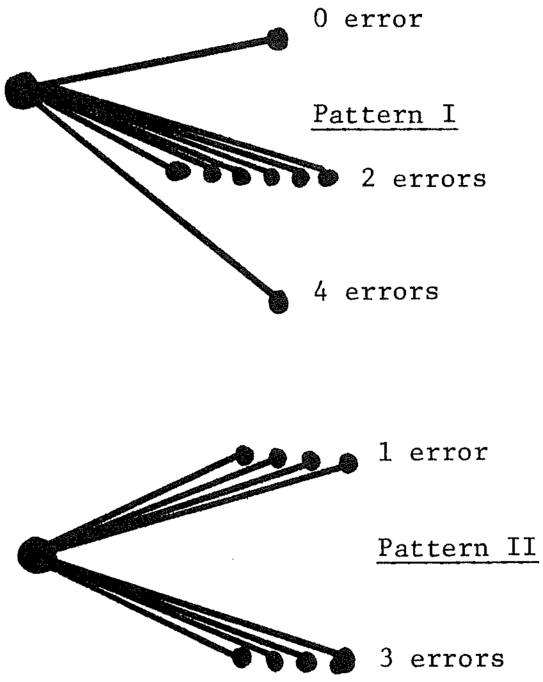

Fig. 11 Rate 3/4 code

Error patterns for rates 2/3 and 3/4 codes over BSC.

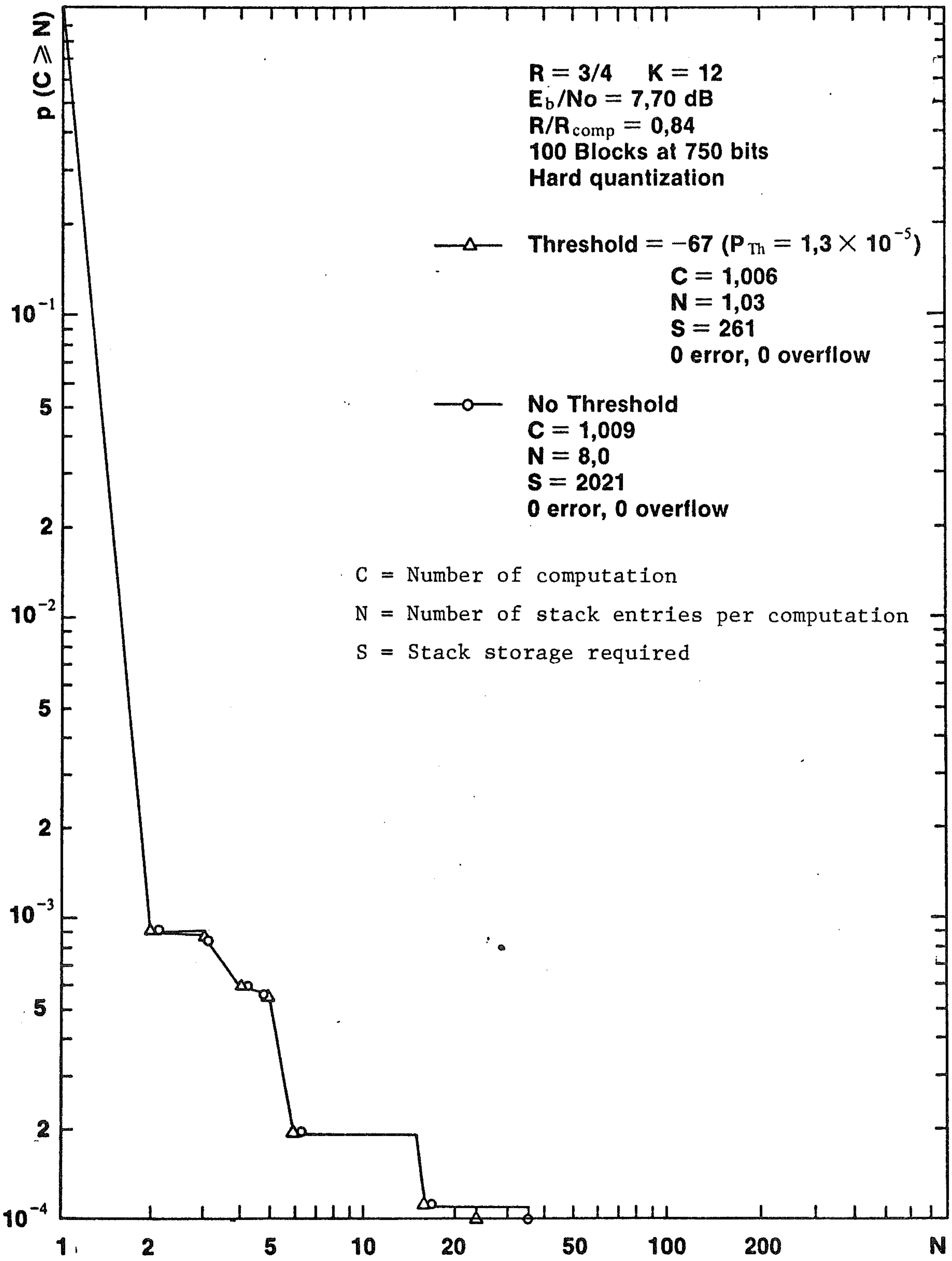

Figure 12: Distribution of computation for rate 3/4 code over the BSC using a discarding threshold.

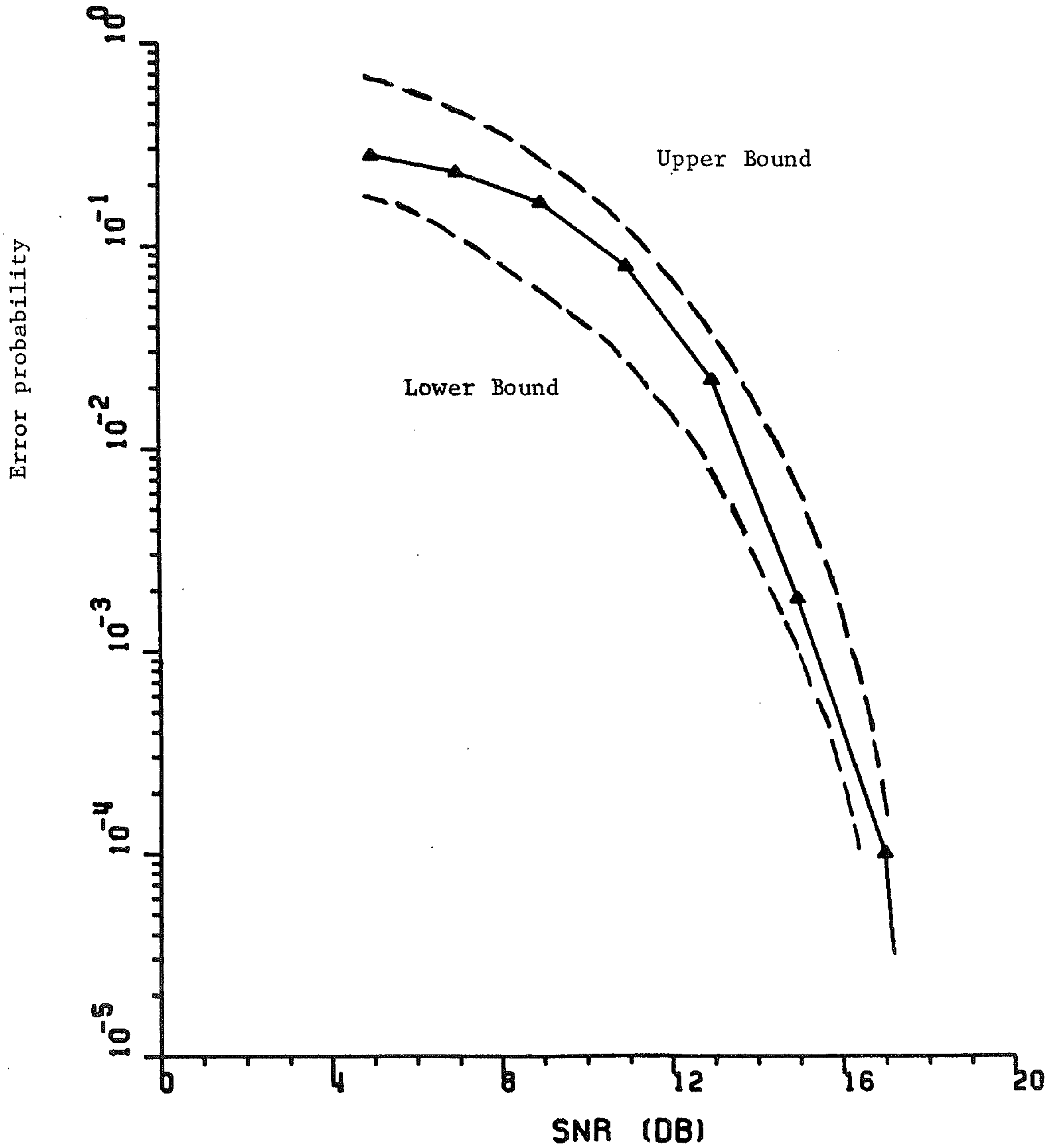

Figure 13: Performance of 7-path sequential decoding over channel of memory 6.
(I.S.I extending over 6 bits).