A HYBRID SEQUENTIAL AND ALGEBRAIC DECODING SCHEME

by

DAVID DUNCAN FALCONER

B.A.Sc. University of Toronto

(1962)

S.M. Massachusetts Institute of Technology

(1963)

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February, 1967

Signature of Author_____
Department of Electrical Engineering, September 19, 1966

Certified by_____
Thesis Supervisor

Accepted by_____
Chairman, Departmental Committee on Graduate Students

A HYBRID SEQUENTIAL AND ALGEBRAIC DECODING SCHEME

by

DAVID DUNCAN FALCONER

Submitted to the Department of Electrical Engineering on September 19, 1966 in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

ABSTRACT

The technique of convolutional encoding and sequential decoding permits reliable communication over discrete memoryless channels at relatively high information rates. However, the number of decoding operations per unit of information is a random variable, and hence the channel output must be buffered. The performance depends strongly on the probability of buffer overflow, which in turn strongly depends on the variability of the decoding effort. In practice, the maximum information rate for which the performance/compexity trade-off is reasonable may be considerably less than the channel capacity.

This paper considers a hybrid sequential and algebraic decoding scheme as a means of improving the computational performance of pure sequential decoding. Information to be transmitted via N separate sequential encoding/decoding systems is pre-coded with an algebraic code block length N. An algebraic decoder may utilize information decoded by the majority of the sequential decoders to assist those few sequential decoders which are currently undergoing the heaviest computational loads. It is shown that this hybrid technique improves the asymptotic distribution of computation to the extent that, for sufficiently large N, the average computation is finite for any rate less than channel capacity. Moreover, if a fixed buffer size is allotted to each sequential decoder, and the block length N is allowed to increase, the buffer overflow probability decreases exponentially with the total buffer storage.

Interesting performance parameters are calculated for two types of channels, and a tentative design procedure is proposed. The theoretical results are supplemented by the results of a computer simulation of the hybrid scheme.

Thesis Supervisor: Irwin M. Jacobs

Title: Associate Professor of Electrical Engineering

ACKNOWLEDGMENTS

I wish to express my sincere thanks to Professor Irwin M. Jacobs for his patient and encouraging guidance of this research. My association with him has been a rewarding educational experience.

To Professor John M. Wozencraft I owe a debt of gratitude for the initial stimulus and for his continuing interest in the problem. I also wish to thank my thesis readers, Professor Robert G. Gallager and Professor Robert S. Kennedy for their stimulating comments and suggestions.

Many other staff members and graduate students are to be thanked for their helpful comments and stimulating discussions. I am grateful to Charles Niessen for his invaluable assistance during the programming for the simulation.

I gratefully acknowledge the support extended to me by the Research Laboratory of Electronics during this research. Financial support was provided by a research assistantship and by a Hughes Industrial Fellowship. I also wish to thank Project MAC for the use of the PDP-6 computer.

Finally, my wife Marcia deserves my humble and heartfelt thanks for her encouragement, patience, and understanding. Her cheerful inspiration brightened many a dark moment. As well, she did all the typing.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

CHAPTER 1

INTRODUCTION

Shannon's Noisy Channel Coding Theorem has continued to tantalize coding theorists and communication engineers with its promise of arbitrarily reliable communication at information rates up to the capacity of a noisy channel[1]. Finding coding schemes which closely approach this ideal performance with reasonable equipment complexity is of considerable theoretical and practical interest. Several classes of randomly-chosen codes are known which achieve error probabilities decreasing exponentially with encoder complexity[2,3,4]. A more challenging problem is the specification of powerful coding schemes for which the average number of operations to decode a code word is not considered exorbitant. A number of interesting schemes have been proposed for the important class of discrete memoryless channels; these include sequential decoding of convolutional codes[6,7], low-density parity-check codes[5], and several recent multi-stage or "concatenated" coding schemes[9,10,11]. In this report, another scheme is presented which is a hybridization of sequential and algebraic decoding.

1.1 The Source and Channel Models

We shall assume than an information source periodically produces symbols selected equi-probably and statistically independently from a u-letter alphabet. For later convenience, and with no loss of generality, we assume that the source alphabet is a finite field, so that u is a power of a prime.

We shall restrict our consideration to the discrete memoryless channel (DMC) with P inputs, Q outputs, and transition probabilities $\left\{ q_{ij} \quad i = 1,2,\ldots P, \quad j = 1,2,\ldots Q \right\}$. This channel is well-suited to analysis and is a reasonable model for many real channels. Transitions from symbols of the input alphabet to symbols of the output alphabet are statistically independent from one channel use to another. We introduce further notation for the DMC by referring to the coding theorem for this channel.

Block coding for the DMC is accomplished as follows. Source symbols are grouped together to form messages. Each of e^{KR} possible messages is identified with a unique code word which is a sequence of K channel input symbols. The number K is known as the block length. The transmission of each message requires K channel uses, and the information rate is R nats per channel use. The probability of decoding error is minimized by the adoption of a maximum-likelihood decoding rule: given a received code word $\bar{y}$, of K channel output symbols, choose message i, corresponding to code word $\bar{x}_i (i = 1,2,\ldots e^{KR})$ for which $pr(\bar{y} \mid \bar{x}_i)$ is largest.

Gallager[4] has shown that for a random ensemble of block codes with a probability distribution on the channel input symbols of

$\left\{ p_i \quad i = 1,2,\ldots P; \quad \sum_{i=1}^{P} p_i = 1 \right\}$, the probability of decoding error per code word p_e is upper-bounded by

$$p_e < \exp\left[-k E(R)\right] \tag{1.1.1.}$$

where $$E(R) = \max_{\substack{0<s\leq 1, \\ \{p_i\}}} \left[E_o(s) - sR \right]$$

and $$E_o(s) = -\ln \sum_{j=1}^{Q} \left(\sum_{i=1}^{P} p_i q_{ij}^{\frac{1}{1+s}} \right)^{1+s}$$

The quantity $E_0(S)$ is an important function of the channel which will be encountered again in later chapters. The so-called <u>maximum-likelihood error exponent</u> E(R), is positive for $0 \leq R < C$, where C is the capacity of the DMC. The capacity is given by the expression,

$$C = E_o'(o)$$

$$= \max_{\{p_i\}} \sum_{i=1}^{P} \sum_{j=1}^{Q} p_i q_{ij} \ln \frac{q_{ij}}{f_j} \quad \text{nats per channel use.} \tag{1.1.2.}$$

where f_j is the channel output probability:

$$f_j = \sum_{i=1}^{P} p_i q_{ij}$$

$$\sum_{j=1}^{Q} f_j = 1$$

This is the tightest known bound on the error exponent for all rates, except those which are much less than C (a tighter, <u>expurgated</u> bound[4] applies for these rates). The upper bound agrees exponentially

with a recently derived lower bound[12] for rates above a certain rate R_{crit}.

Maximum-likelihood decoding conceptually involves the comparison of a received code with each of e^{KR} possible transmitted code words. For interesting values of K and R, this entails an enormous number (for example, e^{100}) of decoding operations. Consequently, maximum-likelihood decoding is hardly practical; a non-optimum, but efficient, decoding scheme whose complexity grows slower than exponentially with K is necessary.

1.2 Outline of the Hybrid Scheme

Sequential decoding, to be described more fully in Chapter 2, applies to the general class of tree codes whose error-correcting properties are known to be at least as good as those of block codes. The number of operations to decode a given amount of information is a random variable whose mean is a small finite number for rates less than a rate R_{comp}, which is less than channel capacity[6]. The mean approaches infinity for rates above R_{comp}. In the hybrid scheme, information to be transmitted via N separate sequential encoding/decoding systems is pre-coded with an algebraic (block) code of block length N. The N sequential decoders operate in the usual fashion, except that the occasional received symbols which require very large numbers of decoding operations may be decoded much sooner with the aid of an appropriate algebraic decoder. It will be shown that the asymptotic distribution of computation per unit of information may be considerably improved with this technique, and furthermore, that satisfactory performance with finite average computation is possible for any rate less than channel capacity, if N is sufficiently large. This implies a considerable extension of the generality and range of usefulness of sequential decoding.

1.3 Outline of the Report

Chapter 2 reviews the essential concepts and bounds on performance of convolutional encoding and sequential decoding. The only new results here are a lower bound to the waiting line distribution for infinite buffer size, (which is derived in Appendix A), and is not used subsequently), a minor extension of error probability bounds to apply to convolutional codes with finite encoding and decoding constraint lengths, and an upper bound on the distribution of computation for information rates between R_{comp} and capacity. The chapter closes with a provocative discussion of the limitations on the computational performance of sequential decoding. With the exception of the last two sections, Chapter 2 can likely be skipped by readers whoare familiar with sequential decoding.

Chapter 3 contains a description, random-coding analysis, and generalizations of the hybrid scheme. The principle theoretical results on the computational performance of the hybrid scheme are given in sections 3.3, 3.4, 3.5, 3.6, and 3.8. Some possible generalizations are briefly discussed in section 3.7.

Chapter 4 contains curves of performance for two interesting types of channels: the additive white gaussian noise channel and the M-ary erasure channel. The effective pareto exponent is the criterion of performance used here.

In Chapter 5, the results of a simulation of the hybrid scheme on a PDP-6 computer are described.

Possible methods of implementing the hybrid scheme are described in Chapter 6. Some idea of the relative costs of the hybrid scheme

can be obtained from the suggested system configuration. Also in this chapter, a possible approach to system design is tentatively advanced, based on the theoretical results of Chapter 3 and the results obtained from the simulation.

Chapter 7 contains final conclusions and suggestions for future research. The kinship of the hybrid scheme to other multi-stage coding schemes is noted.

In Appendix B, lemmas are proved which are necessary to derive the distribution of computation for rates above R_{comp}. Appendix C contains a brief resume of essential facts about algebraic codes, in particular, the important Reed-Solomon codes. This appendix may be omitted by readers familiar with algebraic codes.

CHAPTER 2

SEQUENTIAL DECODING

Sequential decoding was introduced by J.M.Wozencraft[13] in 1957, and has since been extended and refined by others[14,15,7]. It is the only presently-known decoding scheme for which the average error probability for a random ensemble of codes can be made arbitratily small for rates up to a significant fraction of channel capacity, while the average number of computations per digit remains small. The code ensemble is the ensemble of tree codes.

2.1 Tree Codes and Their Implementation

The name "tree code" is derived from the code structure, which is typified in fig. 2-1. The code is represented as a tree with u branches stemming from each node. In fig. 2-1, $u = 3$. The information to be encoded is assumed to be arriving at the encoder as a sequence of symbols from a u-symbol alphabet. There is obviously a one-to-one correspondence between the set of all possible first ℓ information symbols and the set of all paths of length ℓ branches, stemming from the origin. Thus, the first ℓ information symbols can be considered to direct the encoder along a path ℓ branches

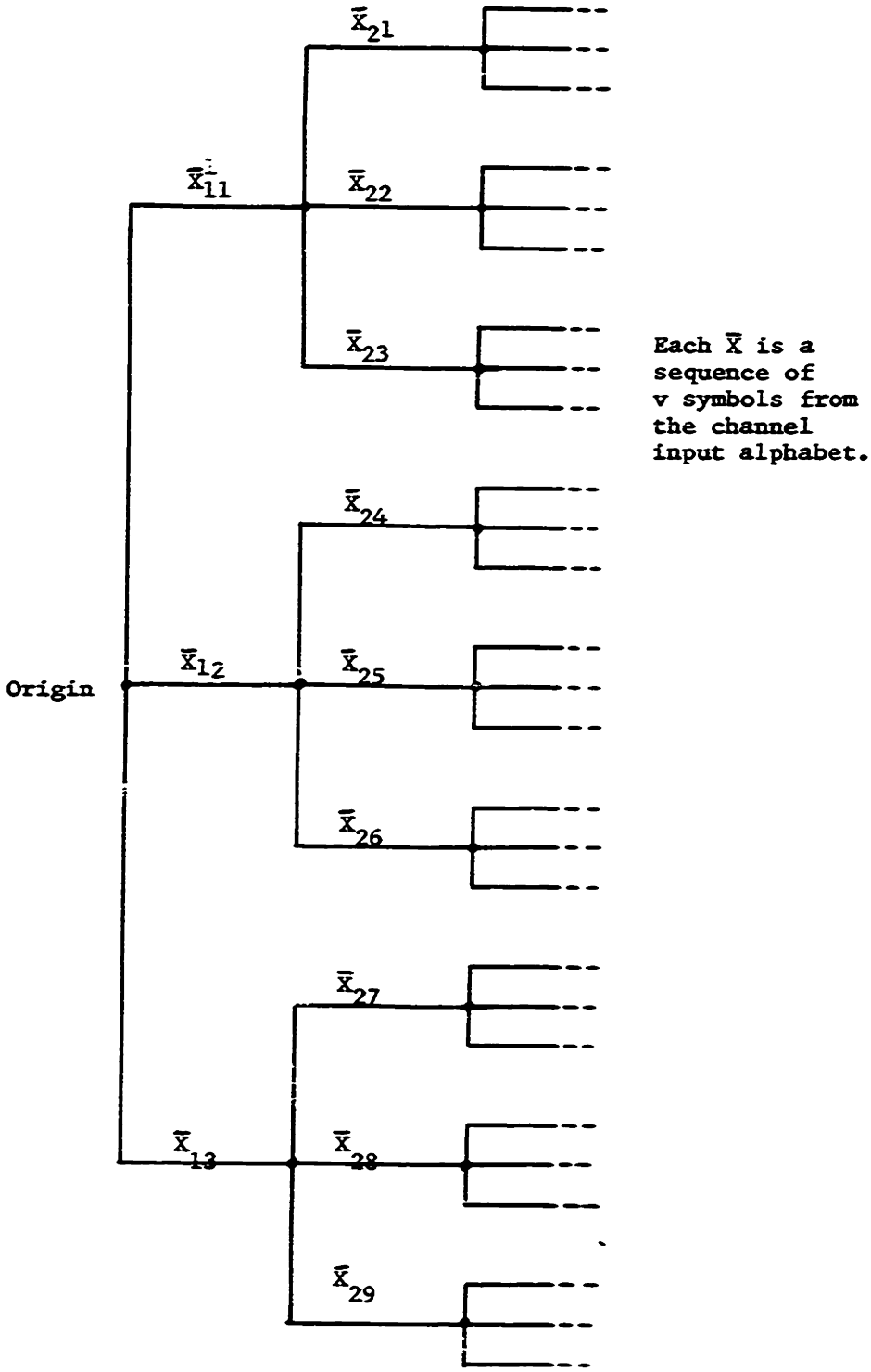

FIG. 2-1 ILLUSTRATION OF A TREE CODE

long. As the encoder traverses each branch of this path, it causes the v channel input symbols, which "label" the branch, to be transmitted over the channel. Thus, the rate of the tree code is $r = 1/v \ln u$ nats per channel use. Because there is no restriction on the encoder output alphabet, tree coding is applicable to any discrete memoryless channel.

The set of tree codes has a subset, called the convolutional codes, which are very easy to instrument with digital components. The channel input symbols are identified with the integers from 0 to P-1. With each of the u possible information symbols, there is associated a generator sequence of $k_e v$ channel input symbols. The integer k_e is the encoding constraint length. Each successive information symbol initiates the transmission of its associated generator sequence superimposed on the generator sequences initiated by previous information symbols. Two or more generator sequences, shifted by multiples of v integers, are superimposed by adding simultaneously-generated integers mod P. The result is a sequence of integers in the range (0, P-1) which is equivalent to a sequence of channel input symbols. Since each information symbol can only affect the present and future transmitted channel symbols

(up to $k_e v$ of them), a convolutional code is also a tree code. In the common case, where source and channel input alphabets are the same finite field, convolutional encoders are implemented with shift registers which store the k_e latest information symbols and v parity nets, each consisting of mod P adders and multipliers[6]. The complexity of such a device is proportional to $k_e v$.

Recently, Viterbi[5] has obtained upper and lower bounds on the probability of error obtainable with convolutional codes. The bounds show that when a suitable semi-optimum decoding procedure is used, convolutional codes with a constraint length of k_e exist which are capable of yielding a much lower probability than that for maximum likelihood decoding of an equivalent block code with block length $k_e v$. Stated more precisely, the error exponent for convolutional codes is greater than the unexpurgated error exponent for block code rates between zero and capacity. Fig. 2-2 compares the error exponents in the upper bounds for block and convolutional codes.

Viterbi's upper bound shows that convolutional codes are potentially more powerful than block codes for the correction of errors. The source of this extra power is the sequential nature of the encoding process. Each information symbol affects the $k_e v$ channel symbols which follow it. An incorrect decoding decision on a branch would affect the decoding of many more than k_e succeeding branches. In particular, all extended paths stemming from an incorrect path would generally appear improbable to the decoder. Thus, if the decoder is allowed to go back and change previous decisions, its <u>effective</u> constraint length, measured by the span of the decoder's past decisions which affect its present decisions, may be much larger than the code constraint length k_e.

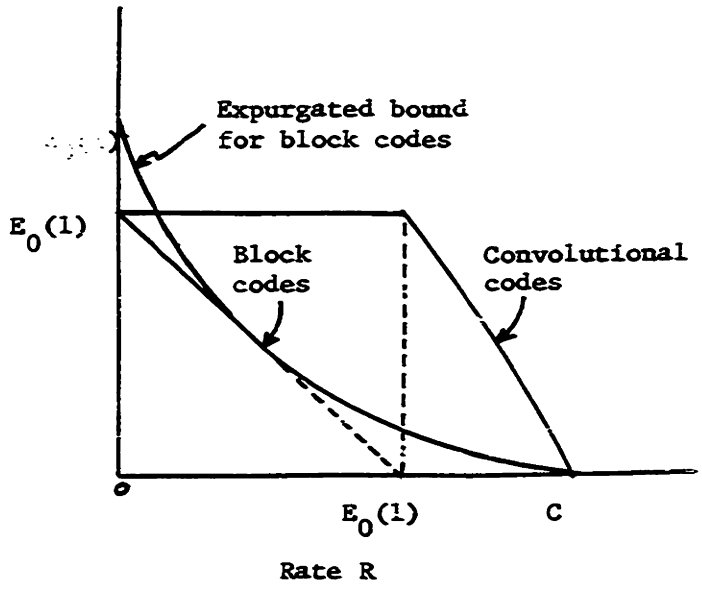

FIG. 2--2 COMPARISON OF BLOCK AND CONVOLUTIONAL CODE ERROR EXPONENTS

2.2 Sequential Decoding

The sequential decoding procedure for tree codes is similar in some respects to the encoding procedure. The decoder conceptually includes a copy of the code tree. Successive sequences of v channel output symbols, termed "received branches", are received from the channel. Each corresponds to a transmitted branch of the code tree. Starting with the first received branch, the sequential decoder compares it to each of the u possible first branches in the code tree, and picks or hypothesizes that one which appears "closest" to the received branch in terms of some appropriate distance criterion. Then it compares the first two received branches with the u two-branch paths whose first branch is already hypothesized. It may hypothesize the closest of these paths and proceed ahead, or, if none of the u paths appears close enough, it may go back and change its hypothesis for the first branch. Similarly, after penetrating the code tree to a distance of ℓ branches, the decoder measures the distance between the first $(\ell+1)$ received branches and each of the u $(\ell+1)$-branch paths whose first ℓ branches are already hypothesized. Depending on the measured distance, it then may either proceed, or backtrack, change the previously-

hypothesized branch, and then try to move forward again. The aim is eventually to trace a path which is closest in terms of an appropriate metric, to the sequence of received branches. A more detailed description of a sequential decoding algorithm and its distance criterion will appear in a later section.

The appealing feature of such a sequential decoding algorithm is that a hypothesis of any branch tentatively rejects an exponentially large number of possible paths initiated by the u-1 remaining branches which stem from the same node. If a wrong hypothesis is made for any branch, the probability is high that the decoder will find no path which contains that branch and which is "close" to the received path. Thus the decoder tends quickly to recognize when a wrong hypothesis has been made, and it backtracks to try and correct it.

2.3 Computation Variability in Sequential Decoding

Sequential decoding is a systematic procedure for finding the correct transmitted path, given the received sequence of branches. Since the hypotheses and details of the decoder's operation are based directly on successive channel outputs, which are random variables, we must expect that the number of times any branch is examined is a random variable also. Let us count each branch examination as a computation. Note that computations require machine time, and that the sequential nature of the decoding implies that computations are done one after another, instead of two or more simultaneously. Therefore, if many computations are required to trace some small section of the correct path, there must be a buffer provided to store the later branches that arrive while the decoder is searching through the corresponding section of the tree. The buffer obviously cannot be infinite, and therefore the "oldest" received branches are continually being pushed out to make room for the newest received branches. Thus, there is the possibility of buffer-overflow if one or more branches require so many computations that before the decoder makes its final decision on them, the

corresponding received branches are pushed out of the buffer. After this happens, there is a high probability that the sequential decoder will not get back onto the correct path, without either a prohibitively large number of computations or the aid of a feedback channel or "resynchronization" interval[6]. Naturally, the speed of the decoder must be large enough to keep up with at least the average computational load.

Jacobs and Berlekamp[16] have obtained lower bounds on statistics of the computational effort in sequential decoding for a DMC. These bounds are especially important since they apply to <u>any</u> purely sequential decoding algorithm. The only necessary assumptions are that:

(1) Branches are examined sequentially so that at least one computation is done for each node examined.

(2) A hypothesis for any branch is not based on any later received branches.

Consider a sequence of N channel output symbols which the decoder will try to map into a segment of the correct transmitted path. Suppose the previous portion of the correct path is known. Let the tree code rate be

$$R = \frac{1}{N} \ln M \qquad \text{nats per channel use,}$$

and define a rate

$$R_s = \frac{1}{s} E_o(s) \qquad (2.3.1.)$$

Designate by C the total number of computations associated with the N channel symbols. Jacobs and Berlekamp's first result is a lower bound on the cumulative probability distribution of C:

There is a length N satisfying

$$N[R_s - E_o'(s)] - o(\sqrt{N}) \gtrsim \ln x > (N-1)[R_s - E_o'(s)] - o(\sqrt{N-1}) \qquad (2.3.2.)$$

for which

$$pr(C \gtrsim x) > x^{-s} \exp\left[-o(\sqrt{\ln x})\right] \qquad (2.3.3.)$$

if rate $R \geq R_s$.

In these inequalities, $O(Z)$, denoting the "order of Z", is a function which for all positive Z is upper-bounded by KZ, where K is a fixed constant. For later use, we note that a little algebra will show that

$$\exp\left[-o(\sqrt{\ln x})\right] > \exp\left[-o\left(\frac{1}{\epsilon}\right)\right] x^{-\epsilon} \qquad (2.3.5.)$$

for any small positive value of ϵ. Thus (2.3.3.) can be re-written as

$$pr(C \geq X) > X^{-(s+\epsilon)} \exp\left[-O\left(\frac{1}{\epsilon}\right)\right] \qquad (2.3.6.)$$

For large values of X, the right hand side of (2.3.3.) or (2.3.6.) behaves as X^{-s}. A probability distribution of this type is known as a pareto distribution. The parameter s, for which $R = R_s$, is known as the pareto exponent. It is clear that $pr(C \geq X)$ decreases rather slowly with X, especially if s is small.

Jacobs and Berlekamp have shown that the m^{th} moment of computation for a sequence of Λ symbols increases exponentially with Λ if $R \geq R_m$. In particular, as Λ approaches infinity, the mean computation approaches infinity if the pareto exponent is less than one.

The rate for which the average computation becomes infinite is clearly an important parameter for a sequential decoding system. It is called the "computation cutoff rate" or R_{comp} and is given by:

$$R_{comp} = E_0(1) = \sum_{j=1}^{Q} \left(\sum_{i=1}^{P} p_i \sqrt{q_{ij}} \right)^2. \qquad (2.3.7.)$$

R_{comp} is always less than the capacity of a DMC; in most, but not all, channels of interest it is equal to or greater than one-half of capacity. Since the mean computation is unbounded for rates above R_{comp}, it would seem that R_{comp} is the maximum usable rate for a sequential decoding system. This statement will be qualified in a later section.

Fig. 2-3 shows the pareto exponent as a function of rate R for a typical DMC.

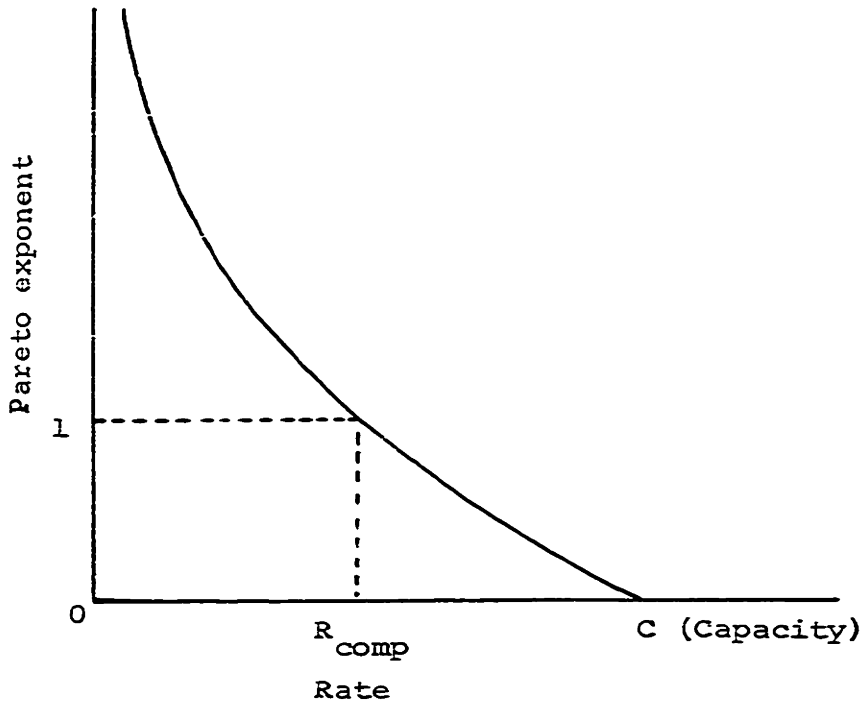

FIG. 2-3 PARETO EXPONENT FOR A TYPICAL DMC

Note that a large pareto exponent (and therefore a distribution $pr(C \geq X)$ with a fairly rapidly-decreasing tail) is achievable only by making the information rate small.

Jacobs and Berlekamp also obtained a lower bound on the probability that a finite buffer of length B overflows or that a decoding error is made during the decoding of Λ branches. The machine speed parameter is μ computations per received channel symbol. The bound is an algebraic function of B:

pr(buffer overflow or error before decoding Λ branches)

$$= P_{\Lambda}(B) > \Lambda (\mu B)^{-s} \exp\left[-O(\sqrt{\ln \mu B})\right] \qquad (2.3.8.)$$

where s is the pareto exponent.

Since B is a measure of the complexity of the decoder, we see that the probability of failure due to buffer overflow for a sequential decoder decreases only algebraically, rather than exponentially with system complexity. For example, if s = 1, at least a hundred-fold increase in B or μ is necessary to achieve a hundred-fold decrease in $pr_{\Lambda}(B)$. Since B and μ are usually limited by cost or size limitations, it is easy to see why buffer overflow is the dominating mode of failure in most practical systems utilizing sequential decoding. In practice, a worthwhile

improvement in performance can only be achieved by a decrease in information rate.

Overflow of a finite buffer occurs when the waiting line of channel output digits that are yet to be decoded exceeds the size of the buffer. A study of the queuing process is of interest for sequential decoding.

It should be clear that the waiting line will not possess a stationary limiting distribution unless μ, the number of computations that the sequential decoder can perform between successive received channel symbols, exceeds the average required number of computations required per received channel symbol. This notion is confirmed by a result due to Lindley[17], which states that a queuing process whose customer's (branches) inter-arrival times and whose service times (number of computations) are both statistically independent random variables with finite means, will have a stationary limiting waiting line distribution if and only if the mean inter-arrival time exceeds the mean service time. In particular, a stationary waiting line distribution is not possible if the pareto exponent is equal to less than one.

The behavior of the waiting line distribution when there is no finite buffer constraint can be shown by

intuitive arguments. The behavior of a typical waiting line as a function of the maximum penetration of the sequential decoder into the tree is shown in fig. 2-4.

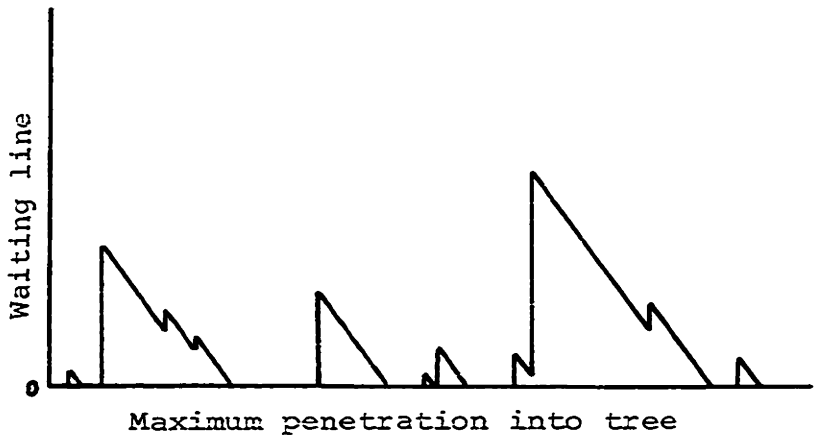

FIG. 2-4 TYPICAL WAITING LINE BEHAVIOR

The almost vertical segments of this plot correspond to long searches by the decoder in the vicinity of a few especially "noisy" received branches. When the noisy branches are finally passed successfully, the waiting line tends to decrease with a slope proportional to $-\mu$ until it drops to zero or the next span of "noisy" branches is encountered. Thus, the waiting line at some time t will certainly exceed a value X if the number of computations associated with the branch received just

previous to t exceeds μX, or if the number of computations associated with any previously-received branch is sufficiently greater than μX that the waiting line cannot possibly decay to a value less than X before time t. Thus one would expect that the stationary waiting line distribution would be given approximately by the integral of the computation distribution from X to infinity. The resulting distribution should be pareto, with exponent equal to (s-1). We substantiate this intuitive result with a rigorous lower bound on the asymptotic waiting line distribution when there is no finite buffer constraint.

Assume the following:

(1) The sequential decoder has been operating long enough so that at least X channel symbols have already been received.

(2) The buffer is infinite.

(3) The encoding constraint length is infinite so (anticipating a result to be stated later) the probability of error is zero.

(4) The average computation per received symbol is less than the speed parameter μ. (Hence the pareto exponent must be greater than one.)

Then we have the following

Theorem: Under the above assumptions, the waiting line distribution, $pr(w \geq X)$ is lower-bounded by a function that behaves as $\mu^{-s} X^{-(s-1)}$ for large values of X, where s is the pareto exponent. More precisely,

$$pr(w \geq X) > \mu^{-(s+\epsilon)} X^{-(s-1+\epsilon)} f(\epsilon, X)$$

where $f(\epsilon, X)$

$$= \exp\left[-o\left(\frac{1}{\epsilon}\right)\right] \cdot \left[o\left(\frac{1}{\ln \mu X}\right)\right]$$

This theorem is proven in Appendix A. Although it is of theoretical interest, it will not be used in the sequel. The result was first shown to be plausible by Gladstone[18], who modelled the waiting line process as a queuing system with Markov arrivals and independent pareto service times. The result shows that in the absence of a finite buffer constraint, the mean waiting line is infinite if the pareto exponent is less than two. This is a reflection of the fact illustrated in fig. 2-4, that if a large "burst" of computation ever causes the waiting

line to rise to a peak value of Y greater than X, the period during which it remains above X thereafter is at least proportional to (Y-X).

Of course, this asymptotic behavior of the waiting line distribution will not be observed in a practical sequential decodin g system with a finite buffer size. However, it might represent a good approximation to the waiting line distribution for systems with very large buffers (for values of X much less than the buffer size).

Up to now, we have mentioned sequential decoding in very general terms and have referred to lower bounds on the statistics of computation which apply to <u>any</u> purely sequential decoding algorithm. Let us now focus attention on the Fano algorithm, which is fairly straightforward to imp lement, and whose distribution of computation asymptotically behaves like the lower bound.

2.4 A practical Realization of Sequential Decoding - The Fano Algorithm

The version of sequential decoding that seems best suited for practical application at present is based on the Fano algorithm. This algorithm has been lucidly described elsewhere[7,20]. Our main aim here is to introduce terminology and to point out certain properties of the algorithm which will be exploited in later derivations. The Fano algorithm is applicable to any real channel that can be modelled as a DMC, and also can be adapted to dispersive channels[23,24,25].

At each step in the decoding procedure, the path which connects the origin to the latest hypothesized branch is the 'hypothesized path'. There must be a distance criterion, which is a function of the hypothesized and received paths. This distance function is commonly termed the path metric, even though it is not a metric in the mathematical sense. For a DMC, an appropriate[7] path metric is given as follows: Consider the i^{th} hypothesized and received branches, consisting repectively of the v channel input symbols $x_1, x_2, \ldots x_v$ and the v channel output symbols $y_1, y_2, \ldots y_v$. Each x_j assumes an integer value from 1 to P, and each y_j assumes an integer value from 1 to Q. For a given pair (x_j, y_j) there are

associated the channel transition probability $pr(y_j|x_j)$ and the channel output probability $pr(y_j)$. Then for the i^{th} received and hypothesized branches, the path metric increment is defined as

$$\lambda_i = \sum_{j=1}^{v} \left[\ln \frac{pr(y_j|x_j)}{pr(y_j)} - U \right] \qquad \text{(in nats)} \qquad (2.4.1.)$$

where U is a positive constant. For a DMC, λ_i increases linearly with the logarithm of the a posteriori probability of the branch,

$$\prod_{j=1}^{v} \frac{pr(y_j|x_j)}{pr(y_j)}$$

The constant U is chosen so that the average path metric along the correct path is positive, while the average path metric along any incorrect path is negative. A value of U equal to the code rate R is commonly used in practice, with some theoretical justification[7]. The path metric for received and hypothesized paths of length ℓ branches is

$$L_{\ell v} = \sum_{i=1}^{\ell} \lambda_i = \sum_{k=1}^{\ell v} \ln \frac{pr(y_k|x_k)}{pr(y_k)} - \ell v U \qquad (2.4.2.)$$

where the received and hypothesized channel symbols are now numbered from 1 to ℓv.

The distance criterion is provided in the Fano algorithm

by an infinite series of numbers or thresholds which are spaced at fixed intervals of Δ nats. The aim of the algorithm is to direct the decoder along a hypothesized path whose metric tends to cross a non-decreasing sequence of thresholds. With high probability, this will be the correct path. Any path whose metric is below a certain threshold is said to violate that threshold. The Fano algorithm continually compares the path metric of the current hypothesized path with the current or running threshold. Whether a path metric satisfies of violates the running threshold at any time determines the decoder's next move. The running threshold is continually being raised ("tightened") or lowered as the algorithm searches for a path whose metric tends to increase.

Fig. (2-5) is a word flow chart of the Fano algorithm. It was adapted from one shown by Wozencraft and Jacobs[20]. The algorithm should be self-explanatory with the aid of the following notes:

(1) Any hypothesized path is characterized by its last node. Thus, given a sequence of channel output symbols, each node has a path metric associated with it. The decoder is said to be "at" that node which is the end of the currently hypothesized path.

(2) The u branches which diverge from a node are ranked from "best" to "worst" in the order of their respective

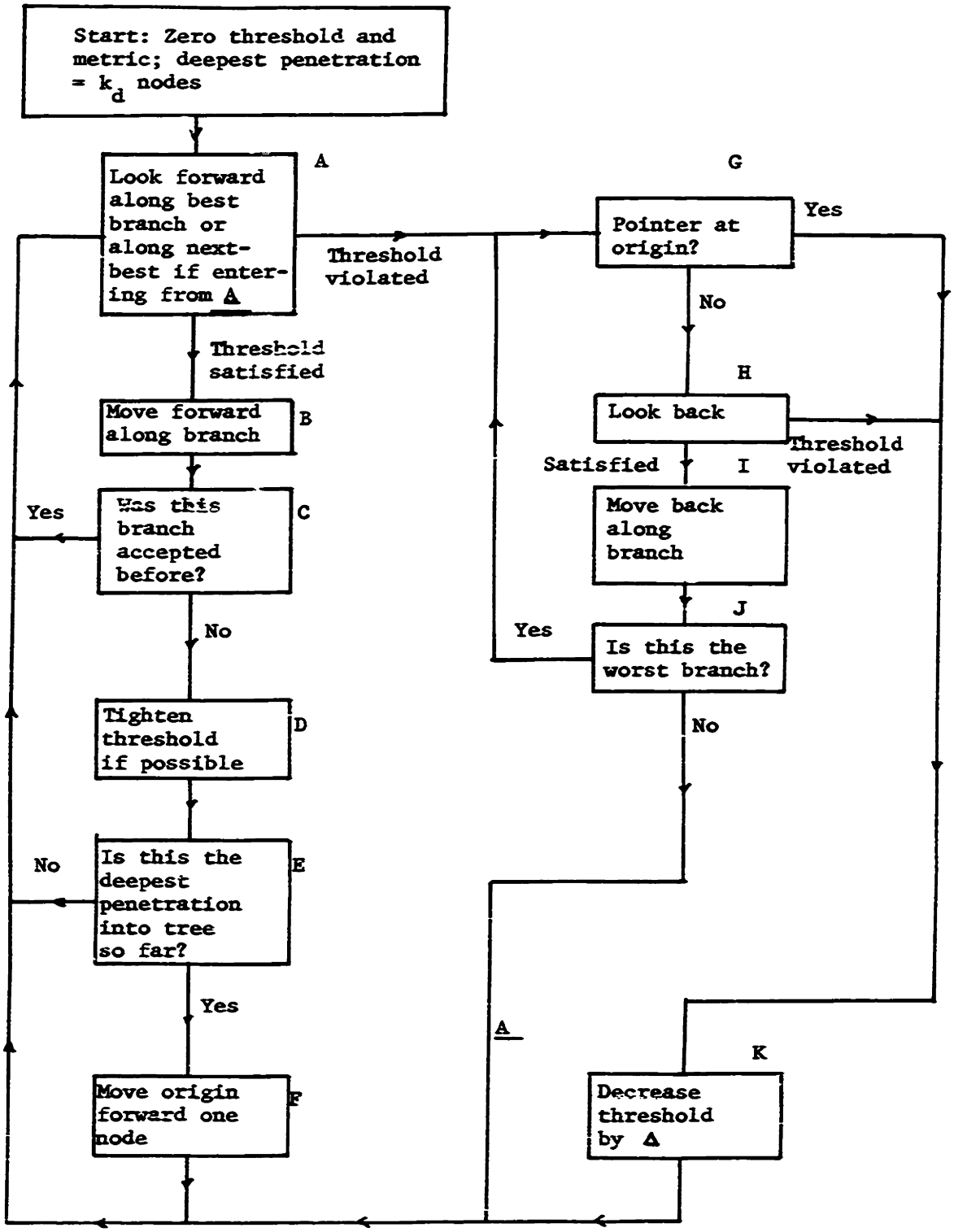

FIG. 2-5 WORD FLOW CHART OF FANO ALGORITHM

path metric increments.

(3) "Looking forward" (box A) along a branch means testing whether the running threshold is satisfied or violated by the node from which the branch diverges.

(4) Sequential decoders to be considered in this thesis will have a fixed decoding constraint length k_d. This means that searching back more than k_d nodes behind the maximum current penetration into the tree (box E) is not allowed. The farthest behind accessible node on the current hypothesized path is the origin (boxes F and G), which is k_d nodes behind the current deepest penetration. Initially, the origin is the first node, and the deepest penetration is set to the k_d^{th} node. We will discuss the consequences of a finite decoding constraint length later.

(5) Box C, asking whether a newly-hypothesized branch was hypothesized previously, is cleverly implemented by several logical instructions and a one-bit stored variable[7,20].

(6) Box D increases (tightens) the running threshold to the highest value that is not violated by the current node. This can only be done after the

decoder has moved forward along a branch that had never been previously hypothesized.

(7) Once a search starts, (algorithm visits box H after visiting boxes D and A) the algorithm tries to find a path which stems from an accessible* node and which does not violate the running threshold. The running threshold is reduced by Δ (box K) if and only if no such path extending to infinity exists. Thus the decoder must visit every accessible node before the threshold can be lowered.

(8) The algorithm is designed so that an endless loop is impossible. It cannot look at any branch more than once in the forward direction and once in the backward direction when a given running threshold is in effect. The prevention of endless loops is accomplished by boxes C and D. Thus, the decoder can not visit any node more than $(u+1)$ times (once for each branch connected to the node) while a given threshold

---

* An accessible node is one which is accessible without violation of the running threshold, i.e. the decoder can move to it through the tree without visiting any node which violates the running threshold.

is in effect. Properties (7) and (8) are useful in deriving upper bounds for the distribution of computation in sequential decoding.

Because of the random nature of the decoding process, a high-speed random-access buffer must be provided to store the incoming branches and the hypothesized information symbols. Fig. (2-6) shows the structure of a typical sequential decoder, with emphasis on the buffer. Interaction of the algorithm with the stored branches and hypotheses is conceptually through pointers which allow received branches to be referenced and hypotheses to be made or changed. The pointers tend to move to the left as the decoder progresses through the tree, but they move to the right when searching causes the decoder's forward progress to be slower than the rate of arriving branches.

The search pointer indicates the branch (and corresponding hypothesized information symbols) which the algorithm is currently examining. The extreme pointer marks the deepest current penetration into the tree. The origin pointer, k_d nodes behind the extreme pointer, marks the current origin. Information symbols to its right are considered irrevocably decoded. The search pointer is always between the other two pointers and tends to remain close to the extreme pointer. Occasionally, a large search

may force the search pointer towards the origin pointer. Juxtaposition of these two pointers constitutes a type of detectable error, since when this happens, some irrevocably decoded symbols are likely to be in error. In practice, for reasonable values of k_d, this event is usually preceded by buffer overflow, in which the origin pointer reaches the far right end of the buffer. As mentioned previously, the decoder must rely on auxiliary feedback[21,22] or resynchronization techniques[6] to restart after a buffer overflow occurs.

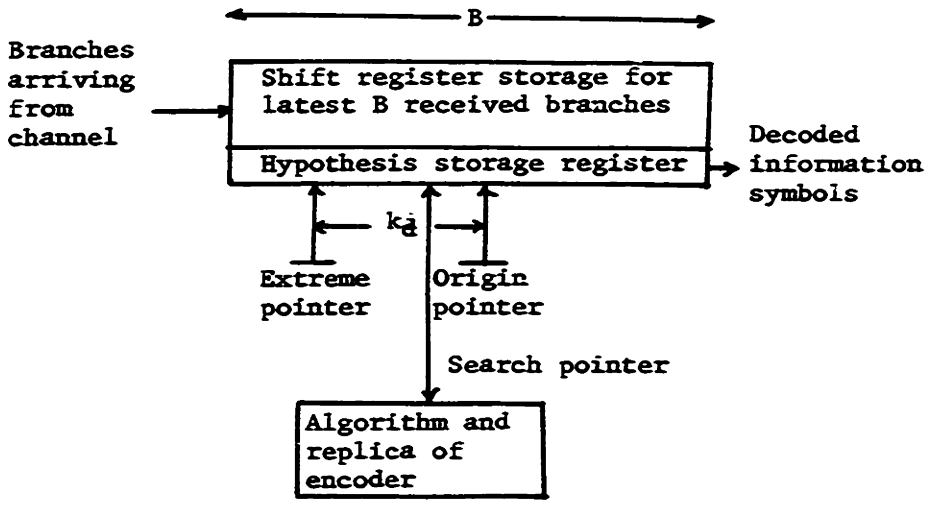

FIG. 2-6 STRUCTURE OF A TYPICAL SEQUENTIAL DECODER

2.5 Probability of Error for Sequential Decoders Which Use the Fano Algorithm

Most of the results quoted in this section were obtained independently by Stiglitz (unpublished) and Yudkin[23] by random coding arguments. They apply to any discretememoryless channel with P input letters, Q output letters, and transition probabilities $\{q_{ij}, i=1,2,\ldots P, j=1,2,\ldots Q\}$. The probabilities are averaged over an ensemble of tree codes with rate R. In this ensemble, each channel input symbol in the tree is picked independently of every other channel input symbol from a probability distribution $\{p_i, i=1,2,\ldots P\}$ The corresponding channel output distribution is $\{f_j = \sum_i p_i q_{ij}, j=1,2,\ldots Q\}$ with $\sum_i p_i = \sum_j f_j = 1$. By a simple modification of the derivations, it can be shown that the results also hold for the subset of this ensemble of tree codes for which all channel input symbols along a path are statistically independent, and there is pairwise statistical independence between any two paths with different first information symbols. This means that the bounds can apply to the ensemble of random convolution codes[6].

The path corresponding to the transmitted message sequence is called the correct path. Any particular node which we want to consider and which lies on the correct

path is called a reference node. A finite constraint length tree is an infinite tree in which any information digit affects no more than the first $k_e v$ channel input symbols on any path stemming from it. A finite constraint length tree may be realized by a convolution encoder which stores the latest k_e information digits and whose tap connections are changed at random at least once every k_e input digits. The random time variation of the encoder ensures that successive segments of the $k_e v$ channel symbols along any path are statistically independent, and facilitates the random code arguments.

For a sequential decoder with an infinite buffer, an infinite decoding constraint length, and a finite encoding constraint length k_e, the only kind of error that can occur is called an undetectable error. This happens when the decoding algorithm, after hypothesizing any information digits of the tree incorrectly, continues on without ever returning to correct those digits. Consideration of the Fano algorithm shows that this event can only happen (but does not always happen) if, stemming from a reference node, there is an "acceptable" incorrect path longer than k_e branches, which matches the correct path in k_e or more consecutive information digits. If the decoder follows such a path past the

span of matching k_e digits, it will, with high probability, continue on without ever returning to the incorrect portion of the path. An "acceptable" path of length L branches is one whose path metric at the L^{th} node exceeds the minimum threshold satisfied by all nodes on the correct path beyond the reference node. Yudkin and Stiglitz have derived a random-coding upper bound on the probability that such an incorrect path exists, which in turn is an upper-bound on the probability of undectable error, $p_u(e)$ for one reference node. Their bound is of the form

$$p_u(e) < \exp\left[-k_e v R_{comp}(U)\right] \qquad (2.5.1.)$$

where, for an appropriate choice of U (including U = R), $R_{comp}(U) > 0$ for $R < \text{Capacity}$.

For a given rate R, with U=R, the quantity $R_{comp}(U)$ is much greater than the unexpurgated random code exponent for block codes with the same rate. Thus for moderately large values of k_e, the probability of undetectable error is miniscule. For $U=R > R_{comp}$, $R_{comp}(U)$ equals Viterbi's upper bound exponent for convolutional codes at rates $R_{comp} < R < C$.

There is a possibility that the decoding algorithm will go more than k_d nodes along an incorrect path

stemming from a reference node thus causing another kind of error if the decoding constraint length k_d, is finite and less than k_e. We call this a detectable error, since in most cases it will be accompanied by buffer overflow. A necessary condition for the occurence of a detectable error is that there exist an acceptable incorrect path of length k_d stemming from a reference node.

Yudkin and Stiglitz have obtained an upper bound on the probability of this event, $p_d(e)$, which holds for the ensemble of randomly-picked tree codes. The bound is of the form

$$p_d(e) < A \exp\left[-k_d E_1(R)\right] \qquad (k_d \le k_e) \qquad (2.5.2.)$$

where A is a constant and

$$E_1(R) = \max_{\eta,\rho,u} \begin{cases} E_a(\eta,\rho) - \rho R & \text{for } u > \frac{1}{\eta\rho}\left[E_a(\eta,\rho) - E_b(\eta,\rho)\right] \\ E_b(\eta,\rho) + u\eta\rho - \rho R & \text{for } u \le \frac{1}{\eta\rho}\left[E_a(\eta,\rho) - E_b(\eta,\rho)\right] \end{cases}$$

$$E_a(\eta,\rho) = -\ln \sum_{j=1}^{Q} \left(\sum_{i=1}^{P} p_i q_{ij}^{1-\eta\rho} \right) \left(\sum_{i=1}^{P} p_i q_{ij}^{\eta} \right)^{\rho}$$

$$E_b(\eta,\rho) = -\ln \sum_{j=1}^{Q} f_j^{1-\eta\rho} \left(\sum_{i=1}^{P} p_i q_{ij}^{\eta} \right)^{\rho}$$

It can be shown that for an appropriate choice of U (including U=R), $E_1(R)$ is positive up to R = channel capacity.

A sketch of how $E_1(R)$ might appear for a typical channel is shown in fig. (2-7). For appropriately-chosen values of U, $E_1(R)$ equals the unexpurgated block random code exponent up to a rate between R_{crit} and R_{comp}. At $R = R_{comp}$, with $U = R_{comp}$, it is one half the block random code exponent. For $R < C$, $E_1(R)$ is usually much smaller than $R_{comp}(U)$. Thus we would expect that the second type of error would predominate in almost all cases of interest.

From these results we can also obtain an upper bound on the probability of error for a finite decoding constraint length k_d which is greater than k_e. By error probability here, we mean the probability that the sequential decoder goes at least k_d nodes along an incorrect path stemming from a reference node. Since $k_d > k_e$, this event includes undetectable errors.

To get this bound we consider a finite constraint length tree with a particular reference node. Incorrect paths (of length $k_d > k_e$) which stem from the reference node are of two types: those which include a succession of k_e or more information symbols which match the corresponding information symbols on the correct path, and

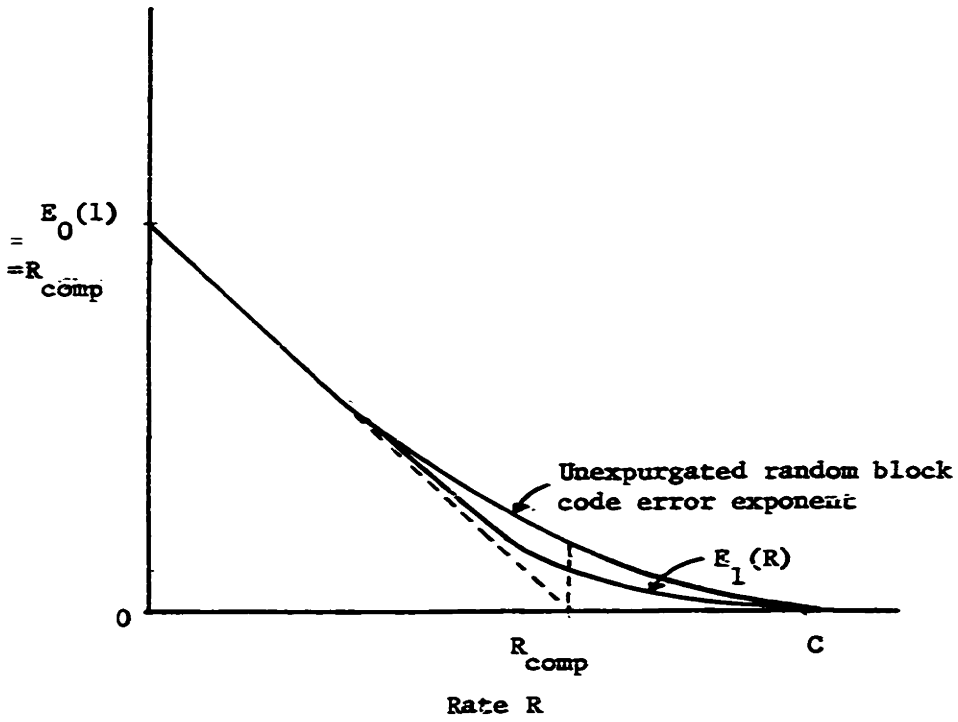

FIG. 2-7 BEHAVIOR OF $E_1(R)$

those which do not. Paths of the first type are included in the set of paths, which, if followed, can cause undetectable errors. Therefore the probability of the first type of incorrect path being followed out to a depth of k_d nodes is upper-bounded by $p_u(e)$.

Incorrect paths of the second type, because of the structure of the random finite constraint length tree, are statistically independent of the correct path along their entire length. The number of such paths is certainly upper-bounded by the total number of incorrect paths of length k_d nodes; therefore the probability that the second type of path is followed out to k_d nodes is less than the probability of detectable error for an infinite constraint length tree, $p_d(e)$. Therefore, for the ensemble of random tree (or convolution) codes with encoding constraint length k_e and decoding constraint length k_d, the probability of error, $p(e)$ is bounded by

$$p(e) \leq p_u(e) + p_d(e)$$

$$\leq A_1 \exp\left[-k_e \vee R_{comp}(U)\right] + A_2 \exp\left[-k_d \vee E_1(R)\right]$$

All of the above bounds involve the probability of error for a single reference node. It is straightforward to show[20] that the probability that one or more of the

first b successive nodes of a tree are decoded incorrectly is bounded by

$$b\,p(e) \leq b\left[A_1 \exp\left[-k_e \vee R_{comp}(U)\right] + A_2 \exp\left[-k_d \vee E_1(R)\right]\right]$$

Thus for rates below capacity the undetectable and detectable error probabilities decrease exponentially with the encoding and decoding constraint lengths respectively. Moreover, the undetectable error exponent is much greater than the detectable error exponent. This means that in most situations, undetected errors are orders-of-magnitude less probable than detected errors, and that the error probability of a sequential encoding and decoding system is governed primarily by the decoding constraint length (in the absence of buffer overflow).

Because of the algebraic dependence of the probability of buffer overflow on the size of the buffer, we see that reduction of the overflow probability by a given factor is much more expensive than a corresponding reduction in the probability of error. Consequently, we will place more emphasis on the computation variability and buffer overflow problems than on the error probability in sequential decoding systems.

2.6 The Distribution of Computation for the Fano Algorithm

In this section, we derive an upper bound on the probability distribution of computation at rates between R_{comp} and capacity, for the Fano Algorithm. First however, the computation variable will be defined and previous upper bounds for rates below R_{comp} will be reviewed briefly.

The only measure of computation that is tractable for upper bounds is the total number of computations (i.e. examinations of hypothesized branches) ever performed among branches of the incorrect subset of a reference node. The incorrect subset consists of the reference node plus all nodes on incorrect paths stemming from the reference node. This is a fairly realistic cost measure, since generation of branch hypotheses, retrieval of received branches from storage, and evaluation of the path metric increments have been found to take up most of the decoder's time in practice. The upper bounds in this section are all derived by random coding arguments for the same ensemble of tree codes as that mentioned in the previous section.

(a) Previous Upper Bounds

The first statistic to be upper-bounded was the average number of computations in the incorrect subset.

For sequential decoding for the DMC, the mean computation is bounded for all rates less than R_{comp}[6,14,22,23]. This result increases the significance of the rate R_{comp} (equation 2.3.7.), which we first mentioned in connection with the lower bounds.

The first analytical result for the probability distribution of computation in sequential decoding was obtained by Savage[26]. The bound is of the following form: If the code rate R is less than R_s, where s is a positive integer, and R_s is defined by equation (2.3.1.), then

$$pr(C > x) < \overline{C^s}\, x^{-s} \qquad (2.6.1.)$$

where $\overline{C^s}$, the s^{th} moment of computation, is bounded. For rates above R_s, the bound on $\overline{C^s}$ diverges. This upper bound, valid for the Fano algorithm, has the same asymptotic form as the lower bound of Jacobs and Berlekamp, when the rate $R \simeq R_s$ (s an integer). For such rates, s is the pareto exponent.

Savage also obtained a heuristic upper bound on the probability of buffer overflow during the decoding of successive branches for $R < R_{comp}$. This bound has the form $(\rho B)^{-s}$ where B is the buffer size and s is the pareto

exponent. This heuristic bound was obtained under the assumption that the decoder operates so fast relative to the average required computational effort that waiting line peaks are nearly disjoint and statistically independent. This assumption appears to be justified for the normal operation of sequential decoders.

Recently Yudkin[27] has shown that Savage's upper bound on the distribution of computation holds also for non-integral values of s greater than one. Thus for $R < R_{comp}$, an upper bound on the distribution of computation for the Fano Algorithm is

$$Pr(C > x) < A_o X^{-s} \quad \text{for rate} \quad R < R_s = \frac{1}{s} E_o(s) \qquad (2.6.2.)$$

where A_o is a constant.

This agrees asymptotically with Jacobs and Berlekamp's lower bound in the same range of rates. It should be noted that Savage's and Yudkin's upper bounds have been shown to apply to the ensemble of random tree codes but not to the ensemble of random convolutional codes. However, it is a very reasonable conjecture (which is supported by experimental evidence) that they do apply to the convolutional code ensemble. Our upper bound for rates above R_{comp} will be shown to apply to the ensemble of convolutional codes.

(b) A New Upper Bound for $R > R_{comp}$

In upper-bounding the distribution of computation for rates above R_{comp}, we follow a procedure similar to that of Savage: first we bound the s^{th} moment of computation, and then we use this with a Chebysheff inequality to bound the distribution. We are concerned with C, the total number of computations ever done in the incorrect subset of a reference node. One computation is said to be done on a node whenever the decoding algorithm visits that node to examine branches diverging from it or leading to it. We assume that the encoding constraint length is infinite so that no undetectable errors are possible. Then, distinct paths diverging from the reference node are statistically independent over their entire length, rather than just over a finite number of branches. We also assume that no detectable errors are made, either because the decoding constraint length is infinite or because errors are corrected by a magic genie, and hence do not affect the decoding of later branches.

Each node in the inncorrect subset will be labelled by indices (ℓ,n), where ℓ is its depth in the tree measured from the reference node and n denotes which of the $(u-1)u^{\ell-1}$

nodes at depth ℓ it is. Consider some node (ℓ,n) and the correct path metric shown in fig. (2-8)

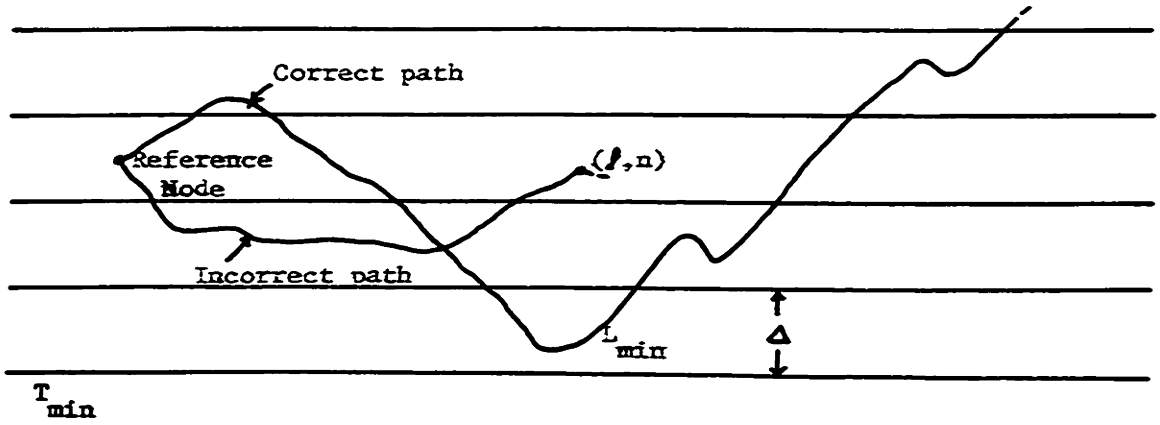

FIG. 2-8 PATH METRICS

Denote the path metric at node (ℓ,n) by

$$L'(\ell,n) = \sum_{k=1}^{\ell_v} \lambda_k'(n)$$

$\lambda_k'(n)$ is the k^{th} path metric increment on the path leading to node (ℓ,n). Denote the path metric at some depth h along the correct path (measured from the reference node) by

$$L(h) = \sum_{k=1}^{h_v} \lambda_k$$

Let $L_{min} = \min_{0 \le h < \infty} L(h)$

be the minimum correct path metric beyond the reference node. The highest threshold that is not violated by L_{min} is denoted by T_{min}.

Note that there is at least one path (the correct path) which remains above T_{min} beyond the reference node. This means (see note [7] of section (2.4)) that no threshold lower than T_{min} need be used by the Fano algorithm in searching the incorrect subset of the reference node. Then, if T is the highest threshold that is not violated by node (ℓ,n), the maximum number of thresholds with which the decoder can visit (ℓ,n) is the number between T and T_{min} inclusive. It can be seen from fig. (2-8) that this number is overbounded by

$$\frac{1}{\Delta}\left[L'(\ell,n) - L_{min} \right] + 2$$

Furthermore, node (ℓ,n) cannot be visited more than (u+1) times for each threshold (note [8] of section (2.4)). Thus, the number of computations $C(\ell,n)$, associated with node (ℓ,n) is overbounded by

$$C(\ell,n) \leq (u+1)\left[\frac{1}{\Delta}\left[L'(\ell,n) - L_{min} \right] + 2 \right] \qquad (2.6.3.)$$

We may re-write (2.6.3.) to make it more convenient for bounding purposes. Let $\varpi(\ )$ be the unit step function.

$\varpi(x) = 1$ if $x \geq 0$

$= 0$ if $x < 0$

We use $\varpi(\ )$ as a counting function in re-writing (2.6.3.):

$$C(\ell,n) \leq (u+1) \sum_{m=1}^{\infty} \phi\left[L'(\ell,n) - \min_{0 \leq h < \infty} L(h) + 2\Delta - m\Delta \right]$$

$$= (u+1) \sum_{m=-1}^{\infty} \phi\left[L'(\ell,n) - \min_{0 \leq h < \infty} L(h) - m\Delta \right] \qquad (2.6.4.)$$

The minimum in (2.6.4.) may be removed and $C(\ell,n)$ further overbounded by summing over h:

$$C(\ell,n) \leq (u+1) \sum_{h=0}^{\infty} \sum_{m=-1}^{\infty} \phi\left[L'(\ell,n) - L(h) - m\Delta \right] \qquad (2.6.5.)$$

Now the <u>total</u> computation in the incorrect subset of the reference node is overbounded by summing $C(\ell,n)$ over all nodes of the incorrect subset. After interchanging the order of the summations for convenience, we get

$$C \leq (u+1) \sum_{m=-1}^{\infty} \sum_{\ell=0}^{\infty} \sum_{h=0}^{\infty} \sum_{n=1}^{M_\ell} \phi\left[L'(\ell,n) - L(h) - m\Delta \right]$$

(2.6.6.)

where $M_\ell = (u-1)u^{\ell-1}$ = number of nodes at depth ℓ.
This expression is equivalent to that considered by others[20,23,26] in bounding moments of computation for the Fano algorithm. The derivation from this point will make use of a random tree code ensemble, the Fano path metric, and some familiar[3] bounding techniques.

The tree code ensemble is the same as that described in the previous section for the upper bounds on error probabilities. Channel input symbols along a path are picked independently from the distribution $\{p_i \quad i=1,2,\ldots P\}$. All possible correct paths are a priori equally probable. Also, there is pairwise independence between the correct path and any incorrect path stemming from the reference node.

We designate sequences of N channel input and output symbols by the N-component vectors $\bar{x}_N$ and $\bar{y}_N$ respectively. Then for N successive channel uses following the reference node, the probability that the correct path is $\bar{x}_N$, that the set of M_N incorrect paths is $\{\bar{x}'_N(1), \bar{x}'_N(2), \ldots \bar{x}'_N(M_N)\}$, and that the channel output is $\bar{y}_N$, is written as

$$p(\bar{x}_N)\, p(\bar{y}_N \mid \bar{x}_N)\, p(\bar{x}'_N(1) \cdots \bar{x}'_N(M_N) \mid \bar{x}_N) \qquad (2.6.7.)$$

with $$p(\bar{x}'_N(n) \mid \bar{x}_N) = p(\bar{x}'(n)) \qquad (2.6.8.)$$

for each individual $\bar{x}'_N(n)$.

We will also make use of the channel output probability $f(\bar{y}_N)$ given by *

$$f(\bar{y}_N) = \sum_{\bar{x}_N} p(\bar{x}_N)\, p(\bar{y}_N \mid \bar{x}_N) \qquad (2.6.9.)$$

$$\text{with} \qquad \sum_{\bar{y}_N} f(\bar{y}_N) = 1 \qquad (2.6.10.)$$

Since successive channel input symbols and successive channel transitions are statistically independent along any path, we note for later reference that

$$p(\bar{x}_N) = \prod_{k=1}^{N} p(x_k) \qquad (2.6.11.)$$

$$p(\bar{x}'_N(n)) = \prod_{k=1}^{N} p(x_k(n))$$

$$p(\bar{y}_N \mid \bar{x}_N) = \prod_{k=1}^{N} p(y_k \mid x_k) \qquad (2.6.12.)$$

$$f(\bar{y}_N) = \prod_{k=1}^{N} p(y_k) \qquad (2.6.13.)$$

where the $\{x_k\}$, $\{y_k\}$, and $\{x'_k(n)\}$, are respectively the

---

* A sum written as $\sum_{x}$ means "sum over all possible x".

components of the vectors $\bar{x}_N$, $\bar{y}_N$, and $\bar{x}'_N$ (n).

The Fano path metric, defined by equation (2.4.2.) may be substituted in the bracketted expression in (2.6.6.) to yield

$$L'(\ell,n) - L(h) - m\Delta$$

$$= \ln\left[\frac{p(\bar{y}_{\ell v}|\bar{x}'_{\ell v}(n))\, f(\bar{y}_{hv})}{f(\bar{y}_{\ell v})\, p(\bar{y}_{hv}|\bar{x}_{hv})}\right] - v(\ell-h)U - m\Delta \qquad (2.6.14.)$$

To bound $\overline{C^s}$ for $0 < s \leq 1$, we first invoke a lemma, which is proven in Appendix B.

Lemma 1

For a sequence of non-negative random variables $\{z_i\}$,

$$\overline{\left(\sum_i z_i\right)^s} \leq \sum_i \overline{z_i^s} \qquad \text{for } 0 \leq s \leq 1 \qquad (2.6.15.)$$

where the overhead bar denotes expectation. Then, since $\omega(\ )$ is a non-negative random variable, we can use (2.6.6.), (2.6.14.), and (2.6.15.) to write

$$\overline{C^s} \leq (U+1)^s \times \sum_{m=-1}^{\infty}\sum_{\ell=0}^{\infty}\sum_{h=0}^{\infty}\overline{\left[\sum_{n=1}^{M_\ell}\phi\left[\ln\left(\frac{p(\bar{y}_{\ell v}|\bar{x}'_{\ell v}(n))\, f(\bar{y}_{hv})}{p(\bar{y}_{hv}|\bar{x}_{hv})\, f(\bar{y}_{\ell v})}\right) - v(\ell-h)U - m\Delta\right]\right]^s} \qquad (2.6.16.)$$

for $0 < s \leq 1$.

We have used lemma 1 successively on all but the innermost sum of (2.6.16.).

Now, the unit step function, $\epsilon(\ )$ in (2.6.16.) may be overbounded by the exponential function $e^{\sigma(\ )}$ for any $\sigma \geq 0$. We will restrict σ so that $0 < \sigma \leq 1$. Making this substitution in (2.6.16.) yields

$$\overline{C^s} \leq (\mu+1)^s \sum_{m=-1}^{\infty} \sum_{l=0}^{\infty} \sum_{h=0}^{\infty} \exp\left[-v\sigma(ms\Delta + lsU - hsU)\right] F(h,l)$$

$$= a_1 \sum_{l=0}^{\infty} e^{-lvs\sigma U} \sum_{h=0}^{\infty} e^{hvs\sigma U} F(h,l) \qquad (2.6.17.)$$

where $$a_1 = (\mu+1)^s \frac{\exp[s\sigma\Delta]}{1-\exp[-s\sigma\Delta]} \qquad (2.6.18.)$$

and $$F(h,l) = \left[\sum_{n=1}^{M_l} \left(\frac{P(\bar{Y}_{lv} \mid \bar{x}'_{lv}(n))\, f(\bar{Y}_{hv})}{P(\bar{Y}_{hv} \mid \bar{x}_{hv})\, f(\bar{Y}_{lv})} \right)^{\sigma} \right]^s \qquad (2.6.19.)$$

Lemma 2

Over the ensemble of convolutional codes described earlier,

$$F(h,l) \leq \begin{cases} M_l^s \exp\left[-(l-h)v\, E_b(\sigma,s) - hv\, E_a(\sigma,s)\right] & (l > h) \\ M_l^s \exp\left[-(h-l)v\, E_c(\sigma,s) - lv\, E_a(\sigma,s)\right] & (l \leq h) \end{cases}$$

for $0 < s \leq 1$ and $0 < \sigma \leq 1$. (2.6.20.)

where

$$E_a(\sigma,s) = -\ell n \sum_{j=1}^{Q} \left(\sum_{i=1}^{P} P_i q_{ij}^{1-\sigma s} \right) \left(\sum_{i=1}^{P} P_i q_{ij}^{\sigma} \right)^{s} \quad (2.6.21.)$$

$$E_b(\sigma,s) = -\ell n \sum_{j=1}^{Q} f_j^{1-\sigma s} \left(\sum_{i=1}^{P} P_i q_{ij}^{\sigma} \right)^{s} \quad (2.6.22.)$$

$$E_c(\sigma,s) = -\ell n \sum_{j=1}^{Q} f_j^{\sigma s} \sum_{i=1}^{P} P_i q_{ij}^{1-\sigma s} \quad (2.6.23.)$$

and $$f_j = \sum_{i=1}^{P} P_i q_{ij} \quad (2.6.24.)$$

This lemma is proven in Appendix B. It is similar to one proven earlier by Yudkin[23].

Note that the code rate R is

$$R = \frac{1}{\nu} \ell n \; u \text{ nats per channel use.}$$

Therefore

$$M_\ell = (u-1)u^{\ell-1} < u^{\ell} = e^{\ell \nu R} \quad (2.6.25.)$$

Then we can re-write (2.6.17.) using lemma 2 and (2.6.25.):

$$\overline{C^s} < a_1 \sum_{\ell=0}^{\infty} e^{-\ell \nu E} \sum_{h=0}^{\ell} e^{-h\nu F} + a_1 \sum_{\ell=0}^{\infty} e^{-\ell \nu G} \sum_{h=\ell+1}^{\infty} e^{-h\nu H} \quad (2.6.26.)$$

where $E = \sigma s U - sR + E_b(\sigma, s)$ (2.6.27.)

$F = E_a(\sigma,s) - E_b(\sigma,s) - \sigma s U$ (2.6.28.)

$G = \sigma s U - sR - E_c(\sigma,s) + E_a(\sigma,s)$ (2.6.29.)

$H = E_c(\sigma, s) - \sigma s U$ (2.6.30.)

The four series in (2.6.26.) are all geometric. Four sufficient conditions for (2.6.26.) to converge are listed in table (2.1).

Conditions	Alternative Statements from (2.6.27.)...(2.6.30.)	
$E > 0$	$E_b(\sigma,s) > sR - \sigma s U$	(2.6.31)
$E+F > 0$	$E_a(\sigma,s) > sR$	(2.6.32)
$H > 0$	$E_c(\sigma,s) > \sigma s U$	(2.6.33)
$G+H > 0$	$E_a(\sigma,s) > sR$	(2.6.34)

Table (2.1) Sufficient Conditions for Convergence of (2.6.26.)

Conditions (2.6.32.) and (2.6.34.) turn out to be identical, so that (2.6.31.), (2.6.32.), and (2.6.33.) are sufficient conditions.

Note that the only present restrictions on σ and s are

$0 < \sigma \leq 1$

and $0 < s \leq 1$.

Let us now further restrict σ to

$$\sigma = \frac{1}{1+s}$$

This will lead to a convenient and tight bound on $\overline{C^s}$.

With this restriction, (2.6.21.), (2.6.22.) and (2.6.23.) may be re-written respectively as

$$E_a(\sigma, s) = -\ln \sum_{j=1}^{Q} \left(\sum_{i=1}^{P} p_i q_{ij}^{\frac{1}{1+s}} \right)^{1+s} = E_o(s)$$

$$E_b(\sigma, s) = -\ln \sum_{j=1}^{Q} f_j^{\frac{1}{1+s}} \left(\sum_{i=1}^{P} p_i q_{ij}^{\frac{1}{1+s}} \right)^{s} = E_b(s)$$

$$E_c(\sigma, s) = -\ln \sum_{j=1}^{Q} f_j^{\frac{s}{1+s}} \sum_{i=1}^{P} p_i q_{ij}^{\frac{1}{1+s}} = E_c(s)$$

The appearance of the function $E_0(S)$ here is reassuring. The three conditions for convergence will be re-written with the aid of the following lemmas:

Lemma 3

For $0 < s \leq 1$,

$$E_b(s) \geq \frac{s}{1+s} E_o(s) \qquad (2.6.35.)$$

Lemma 4

For $0 < s \leq 1$,

$$E_c(s) \geq \frac{1}{1+s} E_o(s) \tag{2.6.36.}$$

These lemmas are proven in Appendix B. With $\sigma = \frac{1}{1+s}$, sufficient condition (2.6.32.) becomes

$$E_o(s) > sR \tag{2.6.37.}$$

By virtue of lemma 3, a sufficient condition corresponding to (2.6.31.) is

$$\frac{s}{1+s} E_o(s) > sR - \frac{s}{1+s} U$$

or

$$E_o(s) > sR + (R - U) \tag{2.6.38.}$$

Also, by virtue of lemma 4, a sufficient condition corresponding to (2.6.33.) is

$$\frac{1}{1+s} E_o(s) > \frac{s}{1+s} U$$

or

$$E_o(s) > sU \tag{2.6.39.}$$

Our final step in establishing the convergence of (2.6.26.) is to set U, which has not hitherto been specified, to equal the rate R. Then all three conditions for convergence (2.6.37.), (2.6.38.) and (2.6.39.) become identical, namely

$$R < R_s = \frac{1}{s} E_o(s) \qquad (2.6.40.)$$

Thus, we have proven the following theorem:

Theorem:

Over the ensemble of random convolution codes, $\overline{C^s}$ $(0 < s \leq 1)$ is bounded by a finite constant, provided that $U = R$ and $R < R_s$.

This theorem is equivalent to Savage's result for integer values of s. Note that R_s is also the rate above which the lower bound on the s^{th} moment of computation diverges[16], as was mentioned previously. Although the measures of computation appearing in the upper and lower bounds are not strictly the same, the similarity in form of our bound to the lower bound strongly suggests that no asymptotically tighter bound is possible.

An upper bound on the distribution of computation for rates between R_{comp} and capacity is now easily obtained by using a generalization of Chebysheff's inequality. For generality we consider the total computations required in the incorrect subsets of Λ successive reference nodes which lie on the correct path. Denote the total computation by

$$C_\Lambda = \sum_{i=1}^{\Lambda} C_i \qquad (2.6.41.)$$

where C_i is the number of computations in the i^{th} incorrect subset. We can bound the s^{th} moment of C_Λ by using lemma 1 of the previous theorem for $0 < s \leq 1$, or Minkowski's inequality[28] for $s \geq 1$. In either case,

$$\overline{C_\Lambda^s} \leq \left[\sum_{i=1}^{\Lambda} \left(\overline{C_i^s} \right)^{\frac{1}{s'}} \right]^{s'} \qquad (2.6.42.)$$

where $s' = \max(1,s)$.

But $\overline{C_i^s}$ is independent of i. Consequently,

$$\overline{C_\Lambda^s} \leq \Lambda^{s'} \overline{C^s} \qquad (2.6.43.)$$

Now, following Savage[26], we prove and then invoke the generalization of Chebysheff's inequality. If Z is a positive random variable,

$$\overline{Z^s} = \sum_x x^s \, pr(z=x)$$

$$\geq \sum_{x \geq x_0} x^s \, pr(z=x) \qquad \text{for some } x_0 \geq 0$$

$$> x_0^s \sum_{x \geq x_0} pr(z=x)$$

$$= x_0^s \, pr(z \geq x_0)$$

Thus, the inequality is,

$$pr(z \geq x_0) < \overline{z^s} \, x_0^{-s} \tag{2.6.44.}$$

for values of s such that $\overline{Z^s}$ is bounded.

Applying this to C_Λ , we get

$$pr(C_\Lambda \geq x) < \Lambda^{s'} \, \overline{C^s} \, x^{-s} \tag{2.6.45.}$$

where $s' = \max(s,1)$

This is valid, where $\overline{C^s}$ is bounded, i.e. for rate $R < R_s = 1/s \; E_0(s)$. Thus, combining our bound with those of Savage[26] and Yudkin[27], the distribution of computation

in Λ successive incorrect subsets is bounded by

$$pr(C_\Lambda \geq X) < \Lambda^{\alpha'} A_0 X^{-(\alpha-\epsilon)} \qquad (2.6.46.)$$

where α, the pareto exponent, is defined by

$$R = R_\alpha = \frac{1}{\alpha} E_0(\alpha) \qquad (0 \leq R < \text{capacity})$$

and $\alpha' = \max(1,\alpha)$

ϵ is an arbitrarily small positive quantity, and A_0 (or $\overline{C^{\alpha-\epsilon}}$) is independent of Λ or X, but is finite only for $\epsilon>0$. Note that if the rate R is between R_{comp} and channel capacity, then the pareto exponent α must be between 0 and 1. It approaches 0 at $R = R_0$ = channel capacity. In using this bound from now on, we will ignore the quantity ϵ as trivial, since it would not affect our asymptotic results.

2.7 Discussion of the Distribution of Computation

The pareto exponent estimated by our upper bound for $R \geq R_{comp}$ and by the upper bounds of Savage and Yudkin for $R \leq R_{comp}$, agrees asymptotically with the pareto exponent estimated by the lower bound of Jacobs and Berlekamp. This asymptotic agreement between the lower bound for any pure sequential decoding algorithm and the upper bounds for the Fano algorithm should deter us from looking for any other pure sequential decoding algorithm which has significantly better computational characteristics than the Fano algorithm.

The theoretical results on the statistics of computation have been supported by observations of simulated and real sequential decoders utilizing the Fano algorithm. Jordan and Blustein have simulated Fano-sequential decoding for the binary symmetric channel[30] and later for more general channels[31]. Falconer and Niessen[32] have simulated Fano-sequential decoding for a proposed deep space telemetry system[33]. In addition, actual sequential decoders have been built and studied by Lincoln Laboratory[34]. The number of searches required to advance one node in the tree is always observed to have a pareto distribution, with a pareto exponent closely approximated by the theoretical

value s given by

$$\text{Rate } R = R_s = \frac{1}{s} E_o(s)$$

for $0 < R <$ capacity.

For rates below about 0.9 R_{comp}, the average number of computations per decoded information symbol is small, typically on the order of five. At a rate above R_{comp}, corresponding to a pareto exponent of about 0.8, the decoding was observed[32] to proceed in essentially the same fashion as below R_{comp}, but with more frequent and more severe searches. The observed sample mean number of computations per bit after about 50,000 bits had been decoded was roughly 20.

The "computational cutoff rate" R_{comp} is clearly an important parameter to consider in designing a sequential decoding system. The remarks which follow are motivated by the theoretical and experimental results on the statistics of computation which we have seen up to this point.

We have shown that the distribution of computation $pr(C \geq x)$, for rates between R_{comp} and capacity is upper-bounded by a pareto distribution, which is a finite monotonically-decreasing function of X which is arbitrarily

small for sufficiently large, but finite values of X. Thus one can define a finite number N so as to make the probability that an information symbol requires more than N computations arbitrarily small. It also means that the infinite expectation of the computation distribution arises solely from the behavior of the distribution for infinite values of X. R_{comp} is therefore the rate above which the mean computation is infinite for a pure sequential decoding algorithm when there is no constraint on the maximum number of computations which can be allotted to decode any branch or sequence of branches.

Such a 'pure' unconstrained sequential decoder never will be built. There are always constraints imposed by finite buffer size, maximum number of computations allowed between resynchronization periods or uses of a feedback channel, or other auxiliary procedures such as the one to be described in the next chapter. These constraints act to truncate the tail of the computation distribution. The average computation per decoded information symbol will always be finite (although perhaps large) if the sequential decoder is allowed to pass on as undecodable a small fraction of the incoming channel symbols and if it can be assisted by auxiliary procedures such as periodic resynchronization (see section 3.7(a)). Then, if the

decoding circuitry has sufficiently high, but finite speed, a stationary limiting waiting line distribution[17] will exist, whose mean is finite.

In summary then, there is no fundamental reason why a practical sequential decoding system incorporating the usual auxiliary resynchronization, feedback or other procedures, cannot operate at any information rate less than channel capacity*. Of course, the median of the distribution of computation will rise very rapidly with the rate, and machine speed or buffer size required to achieve a sufficiently low percentage of undecodable data may be exorbitant. In the next chapter we describe and analyze a scheme in which sequential decoding is augmented by algebraic decoding to yield a computational performance which can be asymptotically much superior to that obtainable by sequential decoding alone.

---

* Indeed, this lack of concern about infinite expectations is exemplified by present-day sequential decoders operating with pareto exponents between one and two, whose average waiting lines would be infinite but for their finite buffer constraint!

CHAPTER 3

A Hybrid Sequential and Algebraic Decoding Scheme

The algebraic behavior of the distribution of computation for sequential decoding clearly makes for a rather poor tradeoff between system performance and complexity at rates near or above R_{comp}, where the pareto exponent is small. In this chapter, we present a scheme which incorporates sequential decoding as its major component, but which has a pareto distribution of computation whose asymptotic behavior can be markedly improved without requiring a decrease in information rate.

The scheme essentially employs N parallel sequential encoding and decoding systems which operate almost independently. Not all N information streams entering the convolutional encoders are independent however. A small amount of redundancy is introduced among the input information streams by a block (algebraic) encoder. This redundancy can be exploited by a block (algebraic) decoder to assist those sequential decoders which are experiencing the heaviest computational load. Algebraic encoding and decoding is briefly discussed in Appendix C.

3.1 Encoding

The encoder structure is shown in fig. (3-1). Symbols generated by the N separate convolutional encoders are transmitted over the DMC by connecting the output of each encoder to the channel in turn. It will be convenient to assume that the output streams from the N encoders are transmitted over N separate, identical, statistically independent DMC's, each used $\frac{1}{N}$ times as often as the actual DMC.

A stream of information symbols which are selected equiprobably from a u-letter alphabet enters each convolutional encoder. Thus, u branches diverge from each node of the tree, and if there are v channel input symbols per tree branch, the convolutional code rate r, is $\frac{1}{v} \ell n\ u$ nats per channel use. For convenience, we shall assume that u is an integer power of a prime number, although this is not necessary. The N input information streams are not all independent however. From information supplied by a source, an algebraic encoder forms code words or "blocks"; each is a sequence of N symbols which are selected from a u^b-letter alphabet, where b is an integer ($b \geq 1$). [For several classes of algebraic codes, such as the Reed-Solomon codes, the required value of b increases with N.]

Output of the algebraic encoder is a succession of N-symbol code words or "blocks"

N convolutional encoders
Constraint length = k_e
Rate = r

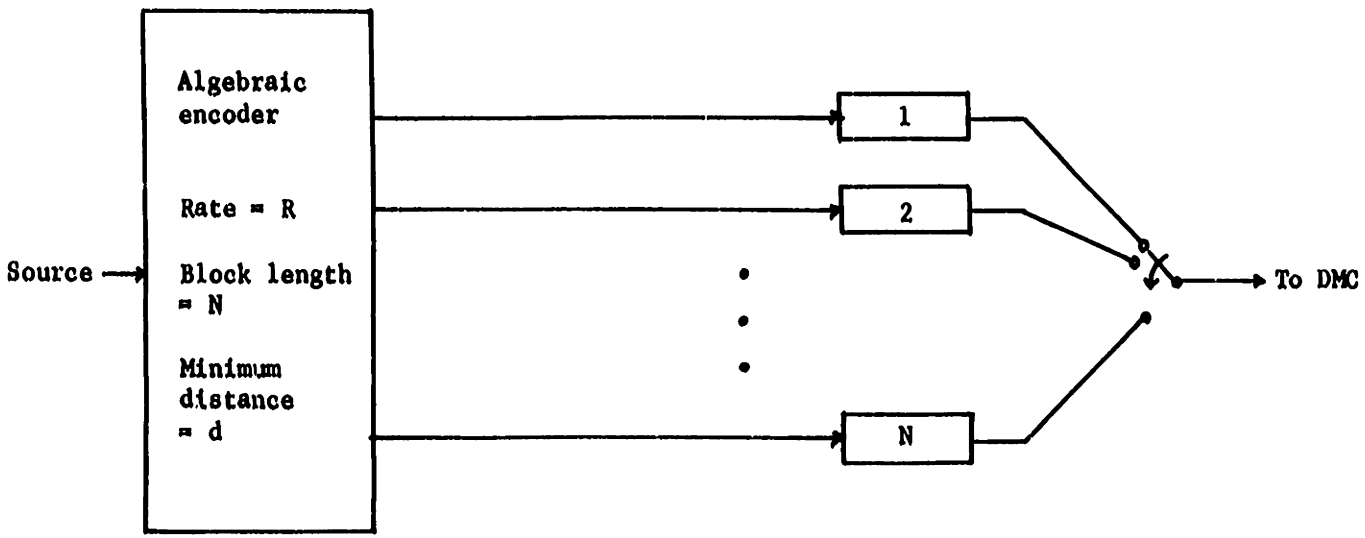

FIG. 3-1 ENCODER STRUCTURE

The algebraic code comprises $(u^b)^{NR}$ equiprobable words, where R is its dimensionless rate. Each of the N symbols in a block enters one of the convolutional encoders as a sequence of b symbols from a u-letter alphabet. In other words, the stream of information symbols entering the i^{th} convolutional encoder is derived from the i^{th} symbols of successive words generated by the algebraic encoder. The overall information rate is rR nats per channel use. If the algebraic code is in systematic form, NR of the input information streams can be considered to originate from NR independent sources, while the remaining N(1-R) information streams are formed from check symbols. Symbols from the u-letter alphabet which enter the convolutional encoders will still be called "information symbols", even though some are redundant. A symbol from the algebraic encoder's u^b-letter output alphabet will be referred to as a "b-symbol" to distinguish it from the b information symbols which comprise it.

3.2 Decoding

Not surprisingly, a decoder appropriate to the coding scheme described in the last section consists of two stages. The first stage comprises N sequential decoders, one corresponding to each of the convolutional encoders. The second stage is an algebraic decoder, whose function is to verify or correct hypotheses made by those sequential decoders which at any time are experiencing the most severe computational load.

Controlled by the Fano algorithm, all N sequential decoders simultaneously and independently attempt to proceed along the correct path in their own trees. Information symbols and their corresponding hypothesized branches which are k_d or more branches back of the farthest current penetration into the tree are considered irrevocably decoded. The current origin of each tree is the end of its current irrevocably-decoded path. The decoding constraint length k_d is made large enough to reduce the detectable error probability to a tolerably low level.

If the algebraic code has a minimum distance of d, then d-1 erased b-symbols can be corrected per block. Consequently, once N-(d-1) b-symbols of a block have been irrevocably decoded by the sequential decoders, the

remaining (d-1) b-symbols are immediately decoded within a fixed number of operations by the algebraic decoder. In this way, those (d-1) b-symbols of a block, which would normally be decoded last, are essentially all decoded as soon as the d^{th}-last b-symbol is decoded by sequential decoding. If an algebraically-decoded b-symbol is identical to the corresponding b information symbols hypothesized by a sequential decoder, the only effect on that sequential decoder will be the advancement of its current origin to the first branch that remains undecoded. However, if the algebraically-decoded b-symbol differs from the sequential decoder's b hypothesized information symbols, the sequential decoder's hypotheses are corrected, its current origin is advanced, and its search pointer is moved to the new current origin on the corrected path. Thus, the algebraic decoder's assistance tends to curtail the longest searches by continually informing the "farthest-behind" sequential decoders of their correct paths.

Fig. (3-2) shows a typical situation just after the algebraic decoder has been used. The tree for each sequential decoder is represented as a triangle with the apex representing the current origin, and the location

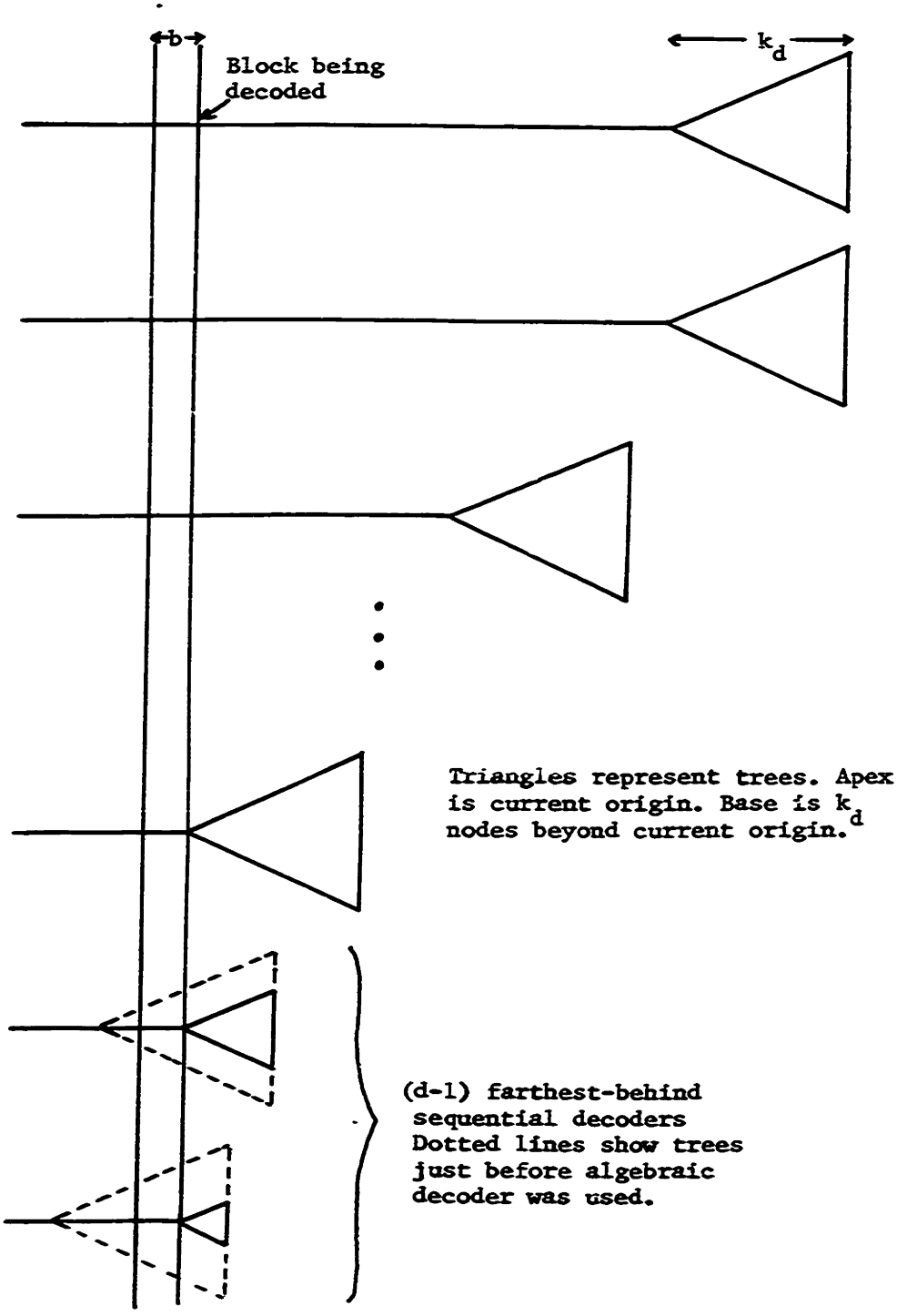

FIG. 3-2 TYPICAL DECODING SITUATION

of the base representing the deepest penetration. Each decoded (presumably correct) path is represented as a horizontal line leading to the origin. The trees are ordered from top to bottom according to their deepest penetration.

In practice, the algebraic decoder would presumably be used periodically, each sequential decoder being allowed to do a small fixed number of computations before the next use of the algebraic decoder. The most eligible algebraic codes are the trivial single parity check codes for small N, and the Reed-Solomon codes for N greater than say, 15 or 20. Both of these classes of codes have the largest possible minimum distance,

$$d = N(1-R) + 1$$

(= 2 for the single parity check codes). Reed-Solomon codes, which are briefly described in Appendix C, have found application in Forney's scheme of concatenated block codes[11]. If a Reed-Solomon code is to be used, the size of the code alphabet must be larger than $N + 1$; i.e. b must be large enough that $u^b > N + 1$.

It is interesting to point out that the algebraic encoding and decoding scheme is, in a sense, well-matched to its 'channel', which consists of the complex of convolutional

encoders, DMC's, and sequential decoders. This "channel" is a symmetric u^b-ary erasure channel (with arbitrarily small crossover probabilities). Each block presented to the algebraic decoder contains exactly d-1 erasures, the maximum number correctable. The erasures are corrected by a fixed number of operations that is proportional to d^2.

Even if the maximum number of correctable erasures per block is small, the use of the algebraic decoder can significantly reduce the frequency of occurrence of very large searches, as will be shown in the next section.

3.3 Upper Bound on the Distribution of Decoding Time for System I

In this section we will upper-bound the probability that the time spent by the sequential decoders in decoding the symbols of a block exceeds some value X. The unit of time will be defined as the maximum time interval required by a sequential decoder to do one computation. The decoding time of a typical algebraic decoder can be upper-bounded by a constant, and will be neglected.

Initially we shall assume that the encoding constraint length, k_e, is greater than $b \div k_d$. We also assume that the N convolutional codes are picked at random from the same ensemble as that considered in the previous chapter. All N codes and discrete memoryless channels are statistically independent and all possible information symbols are used with equal probability. Thus, our previously-derived random coding bounds will apply to the set of N sequential encoding and decoding systems.

We shall first derive an upper bound for a different hybrid scheme which we call system I, and which is easy to analyze, and then show that the result overbounds the distribution of decoding time for the scheme previously described, which we call system II. System I differs from system II in that, as sson as any block has been

decoded, all the sequential decoding algorithms return to the beginning of the next block and start afresh with the tightest possible threshold; no searching back into the b-symbols of previous blocks is allowed. System I also invokes the famous benevolent genie[20], who automatically places each sequential decoder on the correct path to start decoding the next block. System I clearly entails considerable repetition in decoding several adjacent blocks. For this reason, our bound will not be tight, especially for small values of X. We are motivated to consider system I first because of its greater homogeneity; the initial conditions are essentially the same for each successive block to be decoded, so that the bound we will derive will be valid for any decoded block.* On the other hand, an attempt to derive a bound for system II directly would involve statistics of the search pointer movement, which are mathematically intractable. The introduction of the genie in system I avoids the necessity of conditioning probabilities on the event that previous decoding decisions are correct.

---

* This does not imply that the times to decode successive blocks are statistically independent.

In both systems, hypothesized information digits $(b + k_d)$ nodes to the right of the extreme pointer are irrevocably decoded.

After decoding any block, all N sequential decoders start out in unison at the current origin, which is the beginning of the next block. A b-symbol of the block is decoded once the corresponding sequential decoder has progressed at least $k_d + b$ nodes beyond the current origin.

Let $\Gamma = k_d + b$.

Before a b-symbol is decoded, it is considered as an erasure by the block decoder. Our first objective will be to consider a single sequential decoder and find an upper bound for the probability that a b-symbol is still an erasure after X units of time or equivalently, the probability that more than X time units are required for the sequential decoder to decode a b-symbol. We deal with units of time instead of computations because the number of computations done up to a certain time will generally be different for each decoder. This is because the sequential decoders must be able to operate fast enough that they are often idle, waiting for new data to arrive from the channel.

Let μ be the number of time units elapsed between

successive received branches. Thus, μ is a measure of the sequential decoders' speed. Since the speed of any sequential decoder must be greater than the incoming data rate, we must take into account any idle time spent by a decoder which is waiting for new branches to arrive from the channel. To decode a b-symbol, any sequential decoder must usually reach a depth of k_d nodes beyond the last information symbol of the b-symbol. Therefore, while decoding a b-symbol of the first block, a sequential decoder may spend up to $\mu\Gamma$ time units waiting for new branches to arrive. After the first block has been decoded, no decoder needs to spend more than μb time units per block waiting for new branches to arrive.

Suppose we do not start counting time units till after the first Γ-b branches have arrived from the channel. Thus at time zero, the buffer already contains Γ-b branches, and the maximum idle time is μb for all blocks. Initially we consider the N decoders operating independently without assistance from the algebraic decoder. In system I, if the algebraic code can correct d-1 erasures, N-(d-1) sequential decoders must decode b-symbols unassisted, before the algebraic decoder can be used to decode the remaining d-1 b-symbols.

As mentioned in the previous chapter, the only computation variable that is amenable to random-code analysis is the total number of computations ever done in one or more adjacent incorrect subsets. Suppose C is the total number of computations ever done in the incorrect subsets of those Γ successive nodes along the correct path which follow the beginning of a b-symbol. Then a system I decoder will decode a b-symbol within $C + \mu b$ time units. This statement follows if we recall that the genie initially directs the decoder to the correct starting node and that searching in nodes previous to the first is not allowed; therefore after C computations, either all computations in the first Γ incorrect subsets are completed, in which case the algorithm has just reached a depth of Γ nodes, or else not all computations in the incorrect subsets are completed, in which case the decoder has used some of the C computations to attempt to proceed beyond Γ nodes. Therefore the distribution of C is an upper bound on the distribution of the number of computations necessary to decode the first b nodes which comprise the b-symbol.

Therefore, the time required to decode a b-symbol of a block by the j^{th} sequential decoder (j=1,2,...N) of system I is bounded by

$$T_j \leq \hat{T}_j = C_j + \mu b \qquad (3.3.1.)$$

where C_j = total number of computations done by the j^{th} decoder in Γ adjacent incorrect subsets, starting at the beginning of the symbol. Note that the sequential decoder may not actually do as many as C_j computations if it is assisted by the algebraic decoder, or if some of the searches are beyond the decoding constraint length k_d.

A block will be decoded by system I as soon as N-(d-1) sequential decoders, operating independently, have decoded their respective b-symbols; the remaining d-1 b-symbols may then be found by the algebraic decoder.

It is clear then, that the decoding time for a block is bounded by

$$T \leq \mu b + {}_dC \qquad (3.3.2.)$$

where ${}_dC = d^{th}$ largest of $(C_1, C_2, \ldots C_N)$. Then we have

$$\begin{aligned} pr(T \geq X) &\leq pr({}_dC \geq X-\mu b) \\ &\leq pr[\text{d or more of } (C_1, C_2, \ldots C_N) \geq X-\mu b] \end{aligned} \qquad (3.3.3.)$$

Because all the C_j are statistically independent and

identically distributed, we have

$$pr(T \geq x) \leq \sum_{i=d}^{N} \binom{N}{i} P_{x-\mu b}^{\,i} \left(1 - P_{x-\mu b}\right)^{N-i} \qquad (3.3.4.)$$

where $P_{x-\mu b} = pr(C \geq x - \mu b)$

The right side of (3.3.4.) is upper-bounded by use of the familiar Chernoff bound for the asymptotic distribution of sums of binomial random variables[6]:

Let $\delta = \frac{d}{N}$, then

$$pr(T \geq x) < \begin{cases} exp\left[-N\left[T_o(\delta) - H(\delta)\right]\right] & \text{if } P_{x-\mu b} < \delta \\ 1 & \text{if } P_{x-\mu b} \geq \delta \end{cases} \qquad (3.3.5.)$$

where $T_o(\delta) = -\delta \ln P_{x-\mu b} - (1-\delta)\left(1 - P_{x-\mu b}\right)$

and $H(\delta) = -\delta \ln \delta - (1-\delta) \ln (1-\delta)$

If $P_{X-\mu b} < \delta < 1/2$, this bound decreases exponentially with N. Thus, if $P_{X-\mu b}$ is held fixed, pr(T $\geq$ X) can be driven down exponentially (for large values of X) by

increasing the block length N. The bound in (3.3.5.) is known[6] to have the tightest possible exponential dependence on N.

In what follows we will assume that X is chosen large enough so that $P_{X-\mu b} < \delta$ (and $X > \mu b$). Inequality (3.3.5.) may be re-written as

$$pr(T \geq X) \leq exp[NH(\delta)]\ P_{x-\mu b}^{N\delta}\ (1-P_{x-\mu b})^{N(1-\delta)} \qquad (3.3.6.)$$

For large values of X, this is tightly upper-bounded by

$$pr(T \geq X) < exp[NH(\delta)]\ P_{x-\mu b}^{N\delta} = exp[NH(\delta)]\ P_{x-\mu b}^{d} \qquad (3.3.7.)$$

Inequality (3.3.7.) could have been obtained directly from

$$pr(T \geq X) \leq pr\ (C \geq X-\mu b \text{ for some set of } N\delta \text{ sequential decoders}) \leq \binom{N}{d} P_{x-\mu b}^{d}$$

which is valid for any $P_{X-\mu b} < 1$ (3.3.8.)

and the easily-proven result,

$$\binom{N}{d} < exp[N H(\delta)]$$

Now let us substitute the random coding bound for $P_{X-\mu b}$.

$$P_{x-\mu b} = pr(C \geq X-\mu b) < \frac{A_0 \Gamma^{\alpha'}}{(X-\mu b)^{\alpha}} \qquad (3.3.9.)$$

Where $A_0 \leq \overline{C^{\alpha}}$ is a constant, and the rate $r < r_{\alpha} = \frac{1}{\alpha} E_0(\alpha)$

and $\alpha' = \max(1,\alpha)$.

Then (3.3.7.) is re-written

$$pr(T \geq X) < \left[\frac{A_1 \Gamma^{\frac{\alpha'}{\alpha}}}{X-\mu b}\right]^{d\alpha} \qquad (3.3.10.)$$

where $A_1 = A_0^{\frac{1}{\alpha}} \exp\left[\frac{H(\delta)}{\alpha \delta}\right]$

Fig. (3-3) shows a sketch of this bound. The bound is useful only for

$$X \geq \mu b + A_1 \Gamma^{\frac{\alpha'}{\alpha}}$$

For smaller values of X, more useful bounds are

$$pr(T \geq X) < pr(\text{largest of } T_1, T_2, \ldots T_N \geq X)$$

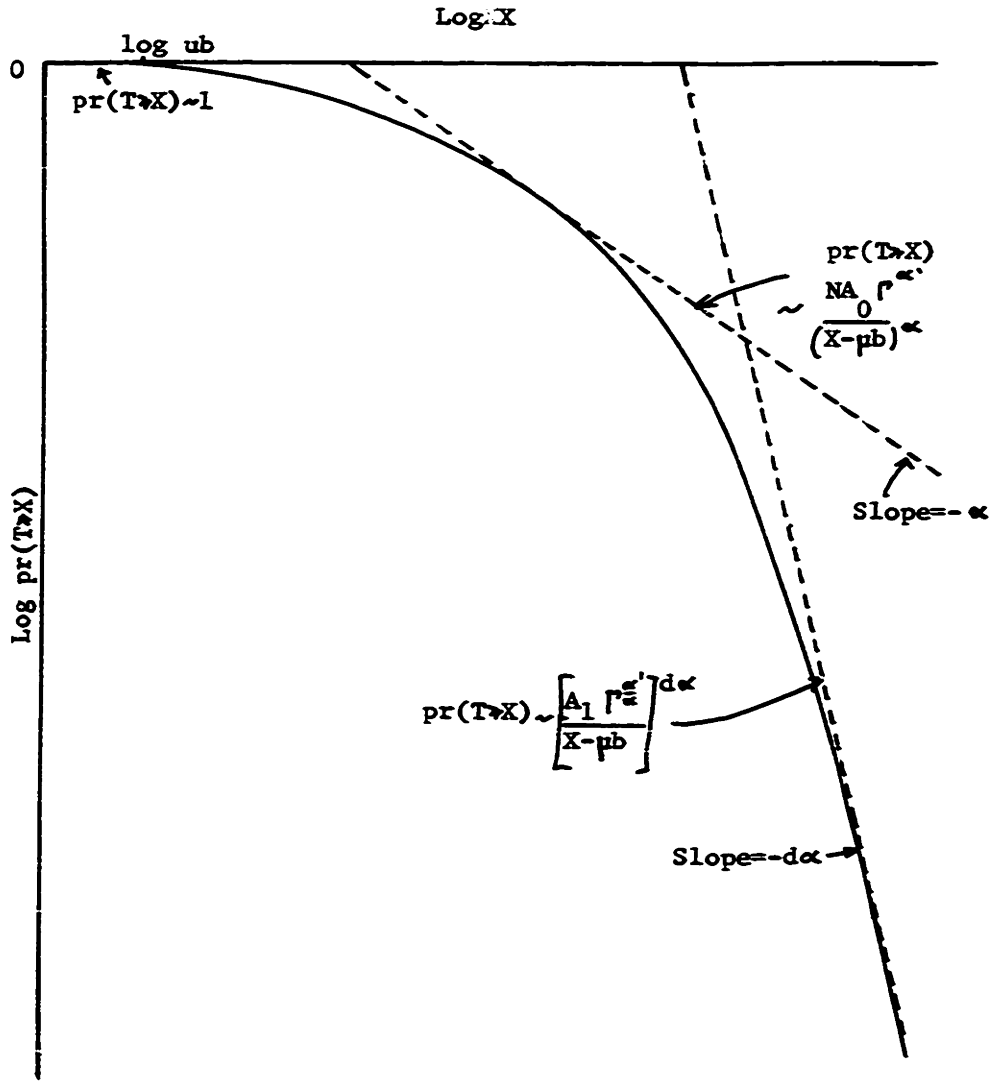

FIG. 3-3 SKETCH OF THE BOUNDS ON pr(T≥X) FOR SYSTEM I

$$= \quad 1 - (1 - P_{x-\mu b})^{N} \qquad (x > \mu b)$$

$$< \quad N \frac{A_0 \Gamma^{\alpha'}}{(x-\mu b)^{\alpha}}$$

Note that if $X > 2\mu b$, (3.3.10.) may be written as

$$pr(T \geq x) < \left[\frac{X}{2A_1 \Gamma^{\frac{\alpha'}{\alpha}}}\right]^{-d\alpha} \qquad (3.3.11.)$$

The result of this section is summarized by the following

Theorem: If T is the decoding time of system I for one block length of N and the minimum distance $d = \delta N$, then for very large X,

$$pr(T \geq x) \lesssim \left(\frac{X}{A}\right)^{-d\alpha}$$

where $$A = 2 \Gamma^{\frac{\alpha'}{\alpha}} A_0^{\frac{1}{\alpha}} \exp\left[\frac{H(\delta)}{\delta \alpha}\right]$$

is a constant, and α is the pareto exponent for "pure" sequential decoding.

The asymptotic distribution of decoding time per block for system I is thus bounded by a pareto distribution whose <u>effective pareto exponent</u> is the original pareto exponent α, multiplied by the minimum distance d, of the block code.

If the algebraic code is Reed-Solomon, (or if it is a simple code with N-1 information digits plus a parity digit),

$$d = 1 + (1-R)N.$$

The only restriction imposed by the use of Reed-Solomon codes is that u^b, the size of the code alphabet, be greater than the block length N.

Thus for Reed-Solomon codes we set $b = \lceil \log_u(N + 1) \rceil$ where $\lceil x \rceil$ means the smallest integer that is greater than x. With this restriction, (3,3.10.) is re-written as

$$pr(T > x) < \left[\frac{A_1 \lceil \log_u(N+1) + k_d \rceil^{\frac{\alpha'}{\alpha}}}{x - \mu \lceil \log_u(N+1) \rceil} \right]^{\alpha + \alpha(1-R)N} \qquad (3.3.12.)$$

The effective pareto exponent, which is meaningful for large values of x, is seen to increase linearly with the block length N, of the Reed-Solomon code.

3.4 Average Computation per Block for System I

Since the effective pareto exponent α_e, for the hybrid system I is $d\alpha$, we would surmise that the average number of computations per block is bounded by a finite constant for $\alpha_e > 1$, even if $0 < \alpha \leq 1$. This is true, as will now be shown.

Let C be the total number of computations done by a particular sequential decoder during the decoding of any block. Then the average, or expectation, of C may be written

$$\bar{C} = \sum_{x=1}^{\infty} x \, pr(C = x)$$

$$= \sum_{x=1}^{\infty} x \left[pr(C \geq x) - pr(C \geq x+1) \right]$$

$$= \sum_{x=1}^{\infty} pr(C \geq x) \qquad (3.4.1.)$$

Now C can only exceed X in system I if C', the corresponding quantity for the same sequential decoder unaided by the algebraic decoder, exceeds X. We choose to overbound pr(C≥X) by $pr(C' \geq X)$ for $1 \leq X \leq \mu b + A_1 \Gamma^{\frac{\alpha'}{\alpha'+1}}$. Also, we recognize that C is overbounded by T, the decoding time for the entire block, and hence we overbound $pr(C \geq X)$ by $pr(T \geq X)$ for $X \geq \mu b + A_1 \Gamma^{\frac{\alpha'}{\alpha'+1}} + 1$.

Thus,

$$\bar{C} \leq \sum_{x=1}^{\mu b + A_1 \Gamma^{\frac{\alpha'}{\alpha}}} pr(C' \geq x) + \sum_{x=\mu b + A_1 \Gamma^{\frac{\alpha'}{\alpha}}+1}^{\infty} pr(T \geq x) \qquad (3.4.2.)$$

But for $0 < \alpha < 1$,

$$\alpha' = 1 \quad \text{and} \quad pr(C' \geq x) \leq A_0 \Gamma x^{-\alpha}$$

where

$$A_0 \leq \overline{C'^{\alpha}}$$

Also, $pr(C' \geq X)$ is a probability. Hence

$$pr(C' \geq x) \leq 1 \quad \text{for} \quad x \leq (A_0 \Gamma)^{\frac{1}{\alpha}}$$

From (3.3.10.)

$$pr(T \geq x) \leq \left[\frac{A_1 \Gamma^{\frac{1}{\alpha}}}{x - \mu b}\right]^{d\alpha}$$

where

$$A_1 = A_0^{\frac{1}{\alpha}} \exp\left[\frac{H(\delta)}{\alpha \delta}\right]$$

Therefore (3.4.2.) may be re-written as

$$\bar{C} \leq \sum_{x=1}^{(A_0\Gamma)^{\frac{1}{\alpha}}} 1 + \sum_{x=(A_0\Gamma)^{\frac{1}{\alpha}}+1}^{\mu b + A_1 \Gamma^{\frac{1}{\alpha}}} A_0 \Gamma x^{-\alpha}$$

$$+ \sum_{x=\mu b + A_1 \Gamma^{\frac{1}{\alpha}}+1}^{\infty} \left[\frac{A_1 \Gamma^{\frac{1}{\alpha}}}{x - \mu b}\right]^{d\alpha} \qquad (3.4.3.)$$

We overbound the sums by integrals.

$$\bar{C} \leq (A_0\Gamma)^{\frac{1}{\alpha}} + \int_{(A_0\Gamma)^{\frac{1}{\alpha}}}^{\mu b + A_1 \Gamma^{\frac{1}{\alpha}}} A_0 \Gamma x^{-\alpha} dx + \int_{\mu b + A_1 \Gamma^{\frac{1}{\alpha}}}^{\infty} \left[\frac{A_1 \Gamma^{\frac{1}{\alpha}}}{x - \mu b}\right]^{d\alpha} dx$$

$$= \frac{A_0 \Gamma}{1-\alpha}\left[\mu b + (A_0\Gamma)^{\frac{1}{\alpha}} \exp\left[\frac{H(\delta)}{\delta \alpha}\right]\right]^{1-\alpha} - \frac{\alpha}{1-\alpha}(A_0\Gamma)^{\frac{1}{\alpha}}$$

$$+ \frac{(A_0\Gamma)^{\frac{1}{\alpha}}}{d\alpha - 1} \exp\left[\frac{H(\delta)}{\delta \alpha}\right] \qquad (3.4.4.)$$

A corresponding bound for $\alpha > 1$ turns out to be

$$\bar{C} \leq \frac{\alpha}{\alpha-1} A_o^{\frac{1}{\alpha}} \Gamma + \frac{A_o^{\frac{1}{\alpha}} \Gamma}{d\alpha - 1} \exp\left[\frac{H(\delta)}{\delta\alpha}\right] \qquad (3.4.5.)$$

For $\alpha = 1$, $\bar{C}$ has the bound

$$\bar{C} \leq A_o \Gamma \ln \frac{\mu b + A_o \Gamma \exp\left[\frac{H(\delta)}{\delta}\right]}{A_o \Gamma \exp\left[\frac{H(\delta)}{\delta}\right]} + \frac{A_o \Gamma}{d-1} \exp\left[\frac{H(\delta)}{\delta}\right] \qquad (3.4.6.)$$

The result of this section may be stated as a

<u>Theorem</u>: If $d\alpha > 1$, the average number of computations $\bar{C}$, done by any of the system I's sequential decoders in decoding a block is overbounded by

$$\bar{C} \leq \begin{cases} \frac{A_o \Gamma}{1-\alpha}\left[\mu b + (A_o\Gamma)^{\frac{1}{\alpha}} \exp\left[\frac{H(\delta)}{\delta\alpha}\right]\right]^{1-\alpha} - \frac{\alpha}{1-\alpha}(A_o\Gamma)^{\frac{1}{\alpha}} \\ \quad + \frac{(A_o\Gamma)^{\frac{1}{\alpha}}}{d\alpha-1} \exp\left[\frac{H(\delta)}{\delta\alpha}\right] & (0 < \alpha < 1) \\ \frac{\alpha}{\alpha-1} A_o^{\frac{1}{\alpha}} \Gamma + \frac{A_o^{\frac{1}{\alpha}} \Gamma}{d\alpha-1} \exp\left[\frac{H(\delta)}{\alpha\delta}\right] & (\alpha > 1) \\ A_o\Gamma \ln \frac{\mu b + A_o\Gamma \exp\left[\frac{H(\delta)}{\delta}\right]}{A_o\Gamma \exp\left[\frac{H(\delta)}{\delta}\right]} + \frac{A_o\Gamma}{d-1} \exp\left[\frac{H(\delta)}{\delta}\right] & (\alpha = 1) \end{cases}$$

All quantities in (3.4.4.) and (3.4.5.) are positive

finite constants. Therefore, if $\alpha_e = d\alpha > 1$, $\bar{C}$ has a finite upper bound which varies roughly as the $1/\alpha^{th}$ power of the decoding constraint length. The average total number of computations to decode the entire block is $N\bar{C}$. Note also that for very large values of μ, the right side of (3.4.4.) behaves as $\mu^{1-\alpha}$. Therefore for $0<\alpha<1$, corresponding to operation of the sequential decoders between the computational cutoff rate R_{comp} and channel capacity, the ratio $\bar{C}/\mu$ decreases to arbitrarily small values as μ increases. This means that sequential decoders which operate at sufficiently high speeds can keep up with the incoming data. Actually, the term μb in (3.4.4.) arises from our conservative requirements that each system I sequential decoder wait μb time units before starting to decode a symbol of a block. It is therefore reasonable to expect that $\bar{C}$ is almost independent of μ for system II. Note that the rate R may be as close to unity as desired, and if N is made sufficiently large, the overall information rate can, in principle, approach arbitrarily close to the channel capacity, without an infinite mean decoding time per block being incurred. In addition, this performance may be achieved with two stages of encoding and decoding, both of which have been shown easy to implement.

3.5 Upper Bound on the Buffer Overflow Probability for System I

An important component of the hybrid decoder is a buffer which stores the B most recent branches received by each sequential decoder. In this section, we overbound $P_L(B)$, the probability that this buffer overflows before L blocks (or NLb information symbols) are decoded by system I. The bound will turn out to be non-trivial only for

$$L < \frac{\mu(B-\Gamma)}{A_1 \Gamma^{\alpha'}}$$

(μ, Γ, A_1, α, and α' have been specified in the previous sections.)

We assume that at time zero, there are already $\frac{\Gamma}{b} - 1$ blocks stored in the buffer. A new branch arrives for each decoder every μ time units. The distribution of T_i, the time to decode the i^{th} block, has been overbounded in section (5.3). We assume that d is large enough so that $d\alpha > 1$, and hence T_i has a finite mean.

For the hybrid scheme, it is useful to define the waiting line as the number of branches received by each sequential decoder since the arrival of the last branches of the most recently decoded block In other words, the waiting line is defined as the longest of the N individual

waiting lines. The buffer will overflow during the decoding of the first L blocks only if the maximum value attained by the waiting line during this time exceeds B received branches. Suppose W_i (measured in received branches) is the maximum attained by the waiting line during the decoding of the i^{th} block. Then

$$P_L(B) = pr\left[\max\left\{W_1, W_2, \cdots W_L\right\} \geq B\right] \qquad (3.5.1.)$$

Now if there are $\frac{\Gamma}{b} - 1$ blocks already in storage at time zero,

$$W_1 \leq \frac{1}{\mu} T_1 + b\left(\frac{\Gamma}{b} - 1\right)$$

$$= \frac{1}{\mu} T_1 + \Gamma - b \qquad (3.5.2.)$$

since up to $\frac{1}{\mu} T_1$ new branches arrive while the first block is being decoded. Just after the first block has been decoded, each sequential decoder has decoded b symbols and the waiting line is overbounded by

$$W_1 - b \leq \frac{1}{\mu} T_1 + \Gamma - 2b$$

Up to $\frac{1}{\mu} T_2$ new branches arrive while the second block is being decoded, and therefore

$$W_2 \leq \frac{1}{\mu}(T_1+T_2) + \Gamma - 2b \tag{3.5.3.}$$

In general,

$$W_i \leq \frac{1}{\mu} \sum_{j=1}^{i} T_j + \Gamma - ib \tag{3.5.4.}$$

The decoding time, T_j can be overbounded by (3.3.2.):

$$T_j \leq \mu b + {}_d C^{(j)} \tag{3.5.5.}$$

where ${}_d C^{(j)}$ = d^{th} largest of $(C_1^{(j)}, C_2^{(j)}, \ldots C_N^{(j)})$; and $(C_1^{(j)}, \ldots C_N^{(j)})$ are the number of computations done during the decoding of the j^{th} block by the first through the N^{th} decoders. Therefore,

$$\frac{1}{\mu} \sum_{j=1}^{i} T_j + \Gamma - ib \leq \frac{1}{\mu} \sum_{j=1}^{i} {}_d C^{(j)} + \Gamma \tag{3.5.6.}$$

Therefore, since ${}_d C^{(j)}$ and Γ are positive,

$$\max(W_1, W_2, \ldots W_L) \leq \frac{1}{\mu} \sum_{j=1}^{L} {}_d C^{(j)} + \Gamma \tag{3.5.7.}$$

and

$$P_L(B) \leq pr\left[\sum_{j=1}^{L} {}_d C^{(j)} \geq \rho(B-\Gamma)\right] \qquad (3.5.8.)$$

Now we must overbound the distribution

$$pr\left[\sum_{j=1}^{L} {}_d C^{(j)} \geq x\right]$$

This may be done with the use of Chebysheff's (see (2.6.44.)) inequality.

$$pr\left[\sum_{j=1}^{L} {}_d C^{(j)} \geq x\right] \leq \overline{\left[\sum_{j=1}^{L} {}_d C^{(j)}\right]^s}\; x^{-s} \qquad (3.5.9.)$$

Lemma: If $d\alpha - s = \epsilon > 0$,

$$\overline{\left[\sum_{j=1}^{L} {}_d C^{(j)}\right]^s} \leq d\alpha\left(1+\frac{1}{\epsilon}\right)\left[A_1 \Gamma^{\frac{\alpha'}{\alpha}}\right]^{d\alpha} \qquad (3.5.10.)$$

Proof: We first invoke Minkowski's inequality[28] for $s > 1$.

$$\overline{\left[\sum_{j=1}^{L} {}_dC^{(j)}\right]^s} \leq \left[\sum_{j=1}^{L} \overline{\left({}_dC^{(j)}\right)^s}^{\frac{1}{s}}\right]^s \qquad (3.5.11.)$$

But

$$\overline{\left({}_dC^{(j)}\right)^s} = \sum_{z=1}^{\infty} z^s \, pr\left({}_dC^{(j)} = z\right)$$

$$= \sum_{z=1}^{\infty} z^s \left[1-\left(1-\tfrac{1}{z}\right)^s\right] pr\left({}_dC^{(j)} \geq z\right)$$

Because $-(1-\frac{s}{z})$ overbounds $-(1-\frac{1}{z})^s$, we may write

$$\overline{\left({}_dC^{(j)}\right)^s} \leq s \sum_{z=1}^{\infty} z^{s-1} \, pr\left({}_dC^{(j)} \geq z\right) \qquad (3.5.12.)$$

Note that for system I, $pr({}_dC^{(j)} \geq z)$ is independent of j. Consequently we drop the j and, following the argument leading to (3.3.3.), we write

$$pr\left({}_dC^{(j)} \geq z\right) \leq pr\left[d \text{ or more of } (c_1, c_2, \cdots c_N) \geq z\right] \qquad (3.5.13.)$$

As in (3.3.8.) this is further bounded by

$$pr\left({}_dC \geq z\right) \leq \binom{N}{d}\left[pr(C \geq z)\right]^d$$

$$\leq \left[\frac{A_1 \Gamma^{\frac{\alpha'}{\alpha}}}{z}\right]^{d\alpha} \qquad (3.5.14.)$$

where the notation is the same as that of section (3.3).

We use this to overbound the right hand side of (3.5.12.).

$$\overline{\left({}_dC\right)^s} \leq s\left[A_1 \Gamma^{\frac{\alpha'}{\alpha}}\right]^{d\alpha} \sum_{z=1}^{\infty} z^{s-1-d\alpha} \qquad (3.5.15.)$$

Now for the sum in (3.5.15.) to converge, we must assume

$$s \leq d\alpha - \epsilon$$

for any $\epsilon > 0$.

Overbounding the sum in (3.5.15.) by an integral and setting $s = d\alpha - \epsilon$, we obtain

$$\overline{\left({}_dC\right)^s} \leq d\alpha\left[A_1 \Gamma^{\frac{\alpha'}{\alpha}}\right]^{d\alpha}\left[1 + \int_1^{\infty} z^{-(1+\epsilon)}\, dz\right]$$

$$= d\alpha\left(1 + \frac{1}{\epsilon}\right)\left[A_1 \Gamma^{\frac{\alpha'}{\alpha}}\right]^{d\alpha} \qquad (3.5.16.)$$

Thus if $s = d\alpha - \epsilon$ and ϵ is fixed at an arbitrarily small

positive value, $\overline{\left(d^{C^{(j)}}\right)^s}$ is a constant, independent of j. Then, using (3.5.11.), we obtain

$$\overline{\left[\sum_{j=1}^{L} d^{C^{(j)}}\right]^{d\alpha-\epsilon}} \leq L^{d\alpha-\epsilon}\, d\alpha\left(1+\frac{1}{\epsilon}\right)\left[A_1 \Gamma^{\frac{\alpha'}{\alpha}}\right]^{d\alpha}$$

Q.E.D.

We may now use the lemma to overbound the right hand side of (3.5.8.). With $X = \mu(B-\Gamma)$ in (3.5.9.) we obtain

$$P_L(B) \leq d\alpha\left(1+\frac{1}{\epsilon}\right)\left[\frac{L A_1 \Gamma^{\frac{\alpha'}{\alpha}}}{\mu(B-\Gamma)}\right]^{d\alpha-\epsilon}\left[A_1 \Gamma^{\frac{\alpha'}{\alpha}}\right]^{\epsilon} \qquad (3.5.17.)$$

and if $B > 2\Gamma$ (as is usually the case),

$$P_L(B) \leq d\alpha\left(1+\frac{1}{\epsilon}\right)\left[\frac{\mu B}{2L A_1 \Gamma^{\frac{\alpha'}{\alpha}}}\right]^{-(d\alpha-\epsilon)}\left[A_1 \Gamma^{\frac{\alpha'}{\alpha}}\right]^{\epsilon} \qquad (3.5.18.)$$

which is non-trivial for

$$B > \Gamma + \frac{L A_1 \Gamma^{\frac{\alpha'}{\alpha}}}{\mu}$$

This condition may be fulfilled for moderate values of L (relative to μB). Note that ϵ may be made as small as we like, so that the exponent in (3.5.17.) is asymptotically equal to that of our bound on $\Pr(T \geq X)$.

The following theorem summarizes the result of this section.

Theorem: If $d\alpha - \epsilon > 1$ in system I, the probability of buffer overflow before L blocks are decoded is overbounded by

$$P_L(B) \leq d\alpha\left(1+\frac{1}{\epsilon}\right)\left(\frac{\mu B}{AL}\right)^{-(d\alpha-\epsilon)} A^{\epsilon}$$

where

$$A = 2A_0^{\frac{1}{\alpha}} \Gamma^{\frac{\alpha'}{\alpha}} \exp\left[\frac{H(\delta)}{\alpha\delta}\right]$$

is a constant.

In particular, the probability that the buffer overflows during the decoding of the first block (L = 1) is bounded by

$$P_1(B) \leq d\alpha\left(1+\frac{1}{\epsilon}\right)\left(\frac{\mu B}{A}\right)^{-(d\alpha-\epsilon)} A^{\epsilon} \qquad (3.5.19.)$$

It was assumed in our model that at least one new block enters the buffer during the time taken to decode one block. This assumption facilitated the derivation of an upper bound for $P_L(B)$ for system I, but the bound is

non-trivial only for relatively small values of L. At this point, we obtain a tighter but non-rigorous bound, using heuristic arguments similar to those of Savage[26].

Assume the speed parameter μ, is large enough so that waiting line peaks are nearly disjoint. Then it can be argued that each large waiting line peak (which may be large enough to cause buffer overflow) is due to an abnormal span of channel transitions, which is statistically independent from such spans which cause other waiting line peaks. The probability of buffer overflow during the decoding of L successive blocks then, can reasonably be expected to be proportional to L; i.e.

$$P_L(B) \leq Lp_1(B) \quad \text{(approximately)}$$

$$\lesssim \quad L \left(\frac{\mu B}{A}\right)^{-d\alpha} \qquad (3.5.20.)$$

for large μB.

Experimentally[31], it appears that $\mu/C > 5$ or 10 is sufficient to ensure that successive waiting line peaks are nearly disjoint.

3.6 Bounds for System II

These results were derived for system I. Let us now relate them to the real system, which we have called system II.

The progress of system I and system II is identical up to the first time that a symbol of the first block is decoded. The corresponding sequential decoder is then allowed to proceed unhindered in system II, while in system I it is moved back to the beginning of the second block and held there until the entire first block has been decoded. Since the sequential decoders are not allowed to change any previously decoded digits in either system, it is clear that the decoding time for the first block is exactly the same in both system I and system II. Furthermore, if system I decodes the first block (or any other) correctly, so does system II (if no errors are made with the genie, none are made without him!)

Suppose that a block has just been correctly decoded. We compare the progress of systems I and II and given the same sequence of channel inputs and outputs. Then all sequential decoders of system I are ready to start afresh with the tightest possible thresholds at the correct starting nodes for the next block. In system II, assume all the sequential decoders are on paths stemming from the correct starting

nodes for the next block. In system II, assume all the sequential decoders are on paths stemming from the correct starting nodes for the next block. We now show that the time for system I to decode the next block will be equal or greater than that for system II.

The next block is decoded in either system, as soon as N-d+1 sequential decoders reach Γ nodes beyond the current origin. We want to show that the first N-d+1 system II sequential decoders will reach the Γ^{th} node before the first N-d+1 system I sequential decoders. This will be proven by showing that any unassisted system I decoder cannot reach the Γ^{th} node before the corresponding system II decoder. Then since the first N-d+1 system I decoders to reach the Γ^{th} node are unassisted, they cannot reach it before the corresponding system II decoders, which in turn take equal or greater time than the quickest N-d+1 system II decoders.

Clearly, if any system II decoder is already beyond the Γ^{th} node, the corresponding system I decoder has lost the race to the Γ^{th} node. Suppose a system II decoder is now between the current origin 0, and the Γ^{th} node. By hypothesis, it is in the correct subset of 0 (we consider the occurrence of errors later). Thus, at some previous

time t, it must have been in the present state of the corresponding system I decoder - at 0 for the first time with the tightest possible threshold.

Since time t, the system II decoder may have visited nodes previous to 0 (which the system I decoder is not allowed to do) but we know it eventually reached its present node and state in the correct subset of 0. From now on it is subject to the same restriction (not to visit nodes previous to 0) as the system I decoder. But for every step backward from 0 that system II decoder took between time t and the present, it must also have taken a step forward to 0; otherwise it could never have reached its present position.

Two properties of the Fano algorithm are that: (1) node 0 cannot be visited in the forward direction twice with the same threshold, and (2) the threshold cannot be lowered until all accessible nodes have been examined. These properties imply that each visit of the system II decoder to node 0 in the forward direction is made with the next lower threshold than on the previous visit.

But any attempt by the system I decoder to visit nodes previous to 0 is thwarted; instead the threshold is lowered one notch, and it attempts to proceed in the forward direction.

Consequently, searching of nodes prior to 0 by the system II decoder is entirely equivalent to the lowering of the threshold by system I decoder when it backs up to node 0.

Therefore the unassisted system I decoder, starting now, will search nodes on paths stemming from 0 in the same order as the system II decoder, starting at time t. In particular, the system I decoder will eventually reach the present node and state of the system II decoder, after which it will duplicate the computations which the system II decoder is about to start now. Thus in the absence of decoding errors, we see that system I cannot decode a block before system II. Therefore, $pr(T \geq X)$, $\overline{C}$, and $P_L(B)$ for system II are bounded by (3.3.11.), (3.4.4.) and (3.5.8.) respectively.

Up to now we have assumed no errors are made by system I or system II. This assumption does not affect our bounds for system I because of the presence of the benevolent genie. However, a detectable error occurring in system II would generally cause subsequent errors and greatly increased decoding times for the later blocks. This condition could only be ended by 'resynchronization' or the use of a feedback channel[6,21]. Such built in proceedures have been found necessary for any communication system employing

convolutional codes. Happily, arbitrarily low detectable and undetectable error probabilities can be bought relatively cheaply (at least relative to the cost of a low buffer overflow probability) by increasing k_d and k_e respectively.

Coming back to the decoding time distribution, we take account of the effect of errors occurring in system II by conservatively assuming that an error always causes $T \geq X$ and adding $L_p(e)$ to the computation time distribution obtained for system I, where L is the number of blocks since the last block knownto be correct (for instance, the last resynchronization interval), and p(e) is the error probability for one block. We can take account of a finite encoding constraint length k_e, by assuming the convolution encoders are randomly time-varying and by conservatively assuming that $T \geq X$ if the conditions necessary for an undetectable error occur. Since there are Nb information symbols in a block, we bound p(e) by the sum of the error probabilities for each information symbol[20]. From the results quoted in section (2.5) with U=R

$$p(e) \leq Nb \left[h_1 \exp\left[-k_e \vee R_{comp}(R)\right] + h_2 \exp\left[-k_d \vee E_1(R)\right] \right] \qquad (3.6.1.)$$

where h_1 and h_2 are constants, and $R_{comp}(R)$ and $E_1(R)$ are positive for rates up to channel capacity. Therefore, combining our bound for system I with the error probability bound we have, for system II,

pr(T $\geq$ X or error before L^{th} block decoded)

$$\lesssim \left(\frac{X}{A}\right)^{-d\alpha} + L\,p(e) \qquad (Large\ \ X) \tag{3.6.2.}$$

where

$$A = 2\,A_0^{\frac{1}{\alpha}}\,\Gamma^{\frac{\alpha'}{\alpha}}\exp\left[\frac{H(\delta)}{\alpha\delta}\right] \qquad (a\ constant)$$

and p(e) is bounded by (3.6.1.).

For the buffer overflow probability $p_L(B)$, in system II, we assume conservatively that the occurrence of a detected or undetected error in any of the first L blocks is sufficient to cause buffer overflow. Thus, by a union bound, for system II,

pr(buffer overflow or error before L blocks decoded)

$$\lesssim d\alpha\left(1+\frac{1}{\epsilon}\right)\left(\frac{\mu B}{AL}\right)^{-(d\alpha-\epsilon)} A^{\epsilon} \tag{3.6.3.}$$

The bound on pr(T $\geq$ X) for system II is likely not tight with respect to the parameters $\Gamma = b + k_d$ and μ for

the following reasons:

(1) It was derived under the assumption that the system I decoders always return to the beginning of the next block after going k_d nodes beyond the end of the last. Thus, while the probability $pr(T \geq X)$ should increase with k_d, it should not vary as $k_d^{d\alpha}$.

(2) The bound was derived with the assumption that during the decoding of any block, at least μb time units elapse while waiting for new branches to arrive from the channel, irrespective of the waiting line before decoding. Thus for $\Gamma < X < \mu b$, $pr(T \geq X)$ should actually be less than 1.

We were forced into using these crude upper bounds in our simple model by the apparent complexity of an analysis based on the dynamical interaction of the various sequential decoders. However, the end result is useful in providing an asymptotic bound.

3.7 Generalizations and Extensions

In this section, we consider several possible variations of the hybrid scheme described earlier.

(a) Decoding Erased Synchronization Blocks

Most communications sytems employing sequential decoding allow the sequential decoder periodically to 'resynchronize itself'; i.e. to start decoding from a new tree origin without requiring a knowledge of previous decoding decisions[6]. This is done by periodically (every M input information symbols) erasing the encoder's memory of all previous input information symbols, thereby defining a new tree origin. Thus, the stream of channel input symbols is effectively divided into blocks, which we will call 'resynchronization blocks'. Each block corresponds to M input information symbols; the channel symbols which comprise it are independent of those comprising any other block. To avoid a large probability of error for the last few information symbols in a resynchronization block, the last k_e information symbols for each block are a sequence which, be pre-arrangement, is known to the receiver.

In practice, there is often a maximum number of computations which the sequential decoder is allowed to perform

to decode a resynchronization block. Blocks requiring more computations are treated as erasures. Suppose the information symbols of each resynchronization block are considered as a letter from a high-order alphabet. Then N such letters can be grouped as a Reed-Solomon code word. If the Reed-Solomon code has rate R and the erasure probability per resynchronization block is less than 1-R, the probability of an erasure being passed on to the user of the information decreases exponentially with N. The distribution of computation for a resynchronization block is given by (2.6.44.) with $\Lambda = M$. Thus if the tolerable number of computations per resynchronization block is large enough, appropriate values of N and R can be found which yield an arbitrarily small probability of uncorrected erasures at any overall information rate up to channel capacity.

(b) Variations of the Algebraic Decoder

If a Reed-Solomon code is used, N(1-R) erased b-symbols per block of N may be decoded. Since the decoder is always presented with exactly this number of erasures in each block, the Reed-Solomon code, whose rate is R, is, in a sense, correcting erasures which occur with probability (1-R). However, R is the capacity (normalized with respect

to alphabet size) of an erasure channel with erasure probability (1-R). In this sense then, the use of a Reed-Solomon code and minimum distance erasure correction is optimum (in the limit of zero error probability).

At least two approaches are available to reduce the probability of decoding error to any arbitrarily small value. The first approach was described previously: increase the decoding constraint length k_d, and correct d-1 erasures with the algebraic decoder. The effective pareto exponent is then $d\alpha$. Asymptotically, the bound on $pr(T \geq X)$ behaves as

$$pr(T>x) \lesssim \left(\frac{X}{A}\right)^{-d\alpha} + L\,p(e) \qquad (3.6.2.)$$

where $$A = 2(k_d+b)^{\frac{s'}{\alpha}} A_0^{\frac{1}{\alpha}} \exp\left[\frac{H(s)}{s\alpha}\right]$$

The bound increases algebraically with k_d, but decreases exponentially with d. Thus, if we are only interested in achieving the best asymptotic behavior, this is the best approach to take.

Note that the bound applies to system I, which is required to repeat many computations unnecessarily to facilitate the analysis. The dependence of the computation

distribution on k_d for the actual system is best determined by simulation. In the simulation which will be described later, the empirical distribution of computation appeared relatively insensitive to changes in k_d which were large enough to cause substantial changes in detectable error probability.

A second possible approach to the reduction of the error probability is to keep k_d fixed, but to allow the algebraic decoder to correct up to w errors and e erasures, where $2w + e \leq d-1$.
The required number of decoding operations is then roughly proportional to d^3. The effective pareto exponent is proportional to the number of correctable erasures, and in this case it is $(d-2w)\alpha$. If achieving the largest possible effective pareto exponent is of paramount importance (and it usually is), then this approach should be avoided, at least for small or moderate values of N. However, if N is large enough so that d is greater than about 10 or 20, the ability to correct one or two errors per block would not greatly diminish the effective pareto exponent, and in fact, might be necessary to make the error probability equal or less than the probability of buffer overflow.

(c) Pooling the Farthest-Behind (d + J) Decoders*

Suppose that after system II has been in operation for a certain time, N-(d+J) sequential decoders have decoded at least n information symbols, while the remaining (d+J) 'farthest-behind' decoders have not. For convenience, we assume (without significant loss of generality), that the alphabet of the Reed-Solomon code is identical to the input alphabet (of u symbols) of the convolutional encoders. The (d+J) undecoded information symbols in each block up to the n^{th} are constrained by d-1 parity relations. Therefore in each block up to the n^{th}, only

$$(d+J) - (d-1) = J+1$$

of these (d+J) information symbols can be specified independently. Similarly, of the (d+J) trees through which the (d+J) farthest-behind decoders are searching, only (J+1) can have independent hypothesized paths of length n branches or less. Then, we can consider merging the trees of the (d+J) farthest-behind decoders to form a composite tree with u^{J+1} branches diverging from each node, and v(d+J) channel input symbols labelling each branch. The rate of this composite tree code is

---

* Suggested by Prof. J.M.Wozencraft

$$r_J = \frac{1}{v(d+J)} \ln\left(u^{J+1}\right) = \frac{J+1}{J+d} r \qquad \text{nats per channel use.}$$

where r is the rate of the individual tree codes. In other words, the (J+1) independently specified information symbols in a block affect the transmission of (J+d)v channel input symbols. Note however, that <u>each</u> of the (J+1) independently specified information symbols in a block affects the transmission of only

$$(J+d)v - Jv = dv$$

channel input symbols. Thus, for $J > 0$, the minimum distance of the composite tree is presumably worse than that of an 'average' randomly-picked tree with the same alphabet size and rate.

In any case, the rate r_J is much smaller than r for small values of J.

$$r_0 = \frac{1}{d} r$$

Thus, we can contemplate 'pooling' the (d+J) farthest-behind sequential decoders to form a sequential decoder operating under essentially the same conditions, but with a code rate $r_J < r$. The pareto exponent $\alpha(r_J)$, corresponding to rate r_J may be less than the original pareto exponent $\alpha(r)$. This is certainly true for $J = 0$, and

may be true for $J > 0$, in spite of the degraded minimum distance of the composite code.

If this concept is to be used in a hybrid scheme, strategies must be specified for bringing the 'pooled' decoder into play for an appropriate value of J, and for dismissing it to allow the individual decoders to proceed on their own once a particularly 'noisy' span of data has been decoded. Note that a decoder formed in this way by pooling the farthest-behind (d+J) decoders can only penetrate out to the latest block for which N-(d+J) information symbols have already been decoded by the remaining sequential decoders.

This pooled decoding concept seems difficult to analyze. However, by a heuristic argument, it is possible to get some feeling for the maximum improvement that can be expected. This argument does not take into account the degraded minimum distance properties of the composite code, and so it should be regarded as a lower bound to performance, at least for $J > 0$.

Statistics for the computations done by the d+J farthest-behind decoders must be conditioned by the fact that they are the farthest-behind. We make the assumption that the ordering of a number of independent sequential

decoders (with the same rate r), in terms of the number of digits they have decoded up to the present time, depends only on their respective channel disturbances, and is independent of the rate r of the decoders. This seems a reasonable approximation, since severe channel disturbances almost always cause high computation, no matter what the code structure or rate. Thus, the computations done by a sequential decoder formed by pooling the (d+J) farthest-behind decoders would presumably be a random variable with about the same probability distribution as the largest of $\binom{N}{d+J}$ independent pareto-distributed random variables with a pareto exponent of $\alpha(r_J)$. Neglecting combinatorial coefficients then, the distribution for the pooled decoder would have a pareto exponent of

$$\alpha_J = \alpha\left(\frac{J+1}{J+d}\, r\right). \quad \text{(This decreases with J.)}$$

This is likely an overestimate for $J \geq 1$, because of the degraded minimum distance of the composite code.

Before this pooled decoder may be activated to work on a particular block however, (N-d+J) symbols of the block must already have been decoded by the individual sequential decoders. The distribution of the number of computations required has an effective pareto exponent of

$$\alpha'_J = (d+J+1)\,\alpha(r).\quad \text{(This increases with J.)}$$

Roughly speaking then, the resulting distribution of computation might reasonably be expected to have an effective pareto exponent which is the smaller of α_J and α'_J. The exponents α_J and α'_J are plotted as functions of J in the following figure.

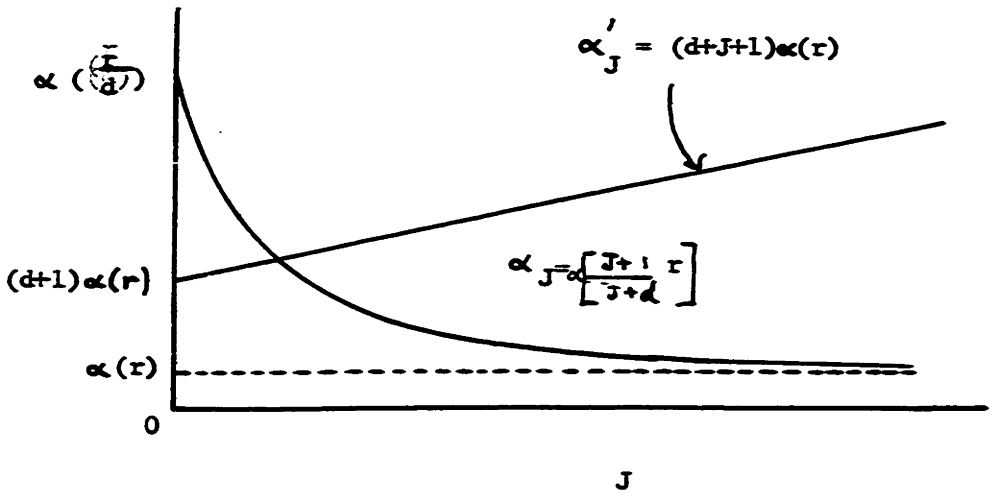

FIG. 3-4 α_J AND α'_J VERSUS J

This graph indicates that a seemingly "optimum" J is that value for which $\alpha_J = \alpha'_J$.

For any channel, we could, in principle, solve the equation $\alpha_J = \alpha'_J$ to find an approximately "optimum" J, and thereby obtain an estimate of the degree of improvement due to pooling the farthest-behind decoders. As a

convenient example, we consider a very noisy channel, for which it can be shown[33] that

$$\alpha(r) = \frac{2R_{comp}}{r} - 1$$

If this expression is substituted in α_J and α'_J and the equation $\alpha_J = \alpha'_J$ is solved for J, the resulting effective pareto exponent for this value of J turns out to be

$$\alpha_e = \frac{d+1}{2}\,\alpha(r)\left[\left(1 + 4\,\frac{d-\alpha(r)-1}{(d+1)^2\,\alpha(r)}\right)^{\frac{1}{2}} + 1\right]$$

Substituting $(d+1)\,\alpha(r) = \nu$ and overbounding $d-\alpha(r)-1$ by $d+1$, we get

$$\alpha_e < \frac{\nu}{2}\left[\left(1 + \frac{4}{\nu}\right)^{\frac{1}{2}} + 1\right] < \nu + 1$$

Therefore for the very noisy channel, the greatest pareto exponent achievable through the use of this pooling technique is apparently less than

$$\nu + 1 = d\,\alpha(r) + \alpha(r) + 1$$

representing an increase of less than $\alpha(r) + 1$ above the value for the simpler hybrid scheme.

This analysis has been heuristic, but its application

to the very noisy channel is felt to be indicative of the actual degree of improvement. In particular, it would seem to indicate that the pooling scheme would be the most useful for small values of J in hybrid schemes where d and the Reed-Solomon code block length are small.

3.8 Discussion of the Hybrid Scheme

It has been shown that the probability distribution of the decoding time per block is overbounded by a pareto distribution whose effective (asymptotic) pareto exponent is the product of the algebraic code's minimum distance d and the original pareto exponent for a single unaided sequential decoder, α. This result, which was derived for system I, holds for system II (the real system) in the absence of errors. If $d\alpha > 1$, the mean computation is finite if no previous errors were made by the sequential decoders. Moreover, values of N, R, and d may be found which guarantee a finite mean for any overall rate rR, less than channel capacity. As channel capacity is approached, however, the original pareto exponent α approaches zero, and the bound on the mean computation increases rapidly (exponentially with $\frac{1}{\alpha}$).

A fairly realistic measure of the cost of the hybrid decoding scheme is the total storage available for the received branches. If each of the N sequential decoders has a buffer of size B, the total storage is BN. An important measure of performance of a scheme employing sequential decoding is the probability of buffer overflow expressed as a function of the total storage. It was shown

that in the absence of errors, the probability of buffer overflow for the hybrid scheme during the decoding of L blocks is rather crudely overbounded by

$$P_L(B) < d_\alpha \left(1 + \frac{1}{\epsilon}\right) \left[\frac{\mu B}{AL}\right]^{-(d_\alpha - \epsilon)} A^{\epsilon} \qquad (3.8.1.)$$

where $A = 2 A_0^{\frac{1}{\alpha}} (k_1 + b)^{\frac{\alpha'}{\alpha}} \exp\left[\frac{H(\delta)}{\delta_\alpha}\right]$

If the algebraic code is Reed-Solomon, the minimum distance d, is proportional to the block length N for fixed rate R, and the parameter b must vary as ℓn N. Suppose L and μ are fixed, and B is made proportional to $(\ell n\ N)^{\frac{\alpha'(1+\epsilon)}{\alpha}}$ with the constant of proportionality large enough so that $\mu B > AL$ for all N. Then if r and R are fixed,

$$d_\alpha = N \delta_\alpha$$

is proportional to N. Then the bound (3.8.1.) behaves as

$$N(\delta_\alpha k_1)(k_2)^{-N\delta_\alpha}$$

where $\delta_\alpha k_1$ and k_2 are constants. This decreases almost exponentially with N, which in turn varies as

$$\frac{\text{total storage}}{(\ell n\ N)^{\frac{\alpha'(1+\epsilon)}{\alpha}}} .$$

Thus, $P_L(B)$ can be made to decrease almost exponentially with total storage for any fixed rate less than channel capacity.

Note that the possibility of decoding errors by the sequential decoders does not present a serious problem, since increasing the decoding constraint length exponentially decreases the error probability, but at most algebraically increases the average computation per block. The attainment of an exponentially low error probability by sequential encoding and decoding components frees the algebraic decoder to correct only erasures, thereby minimizing its complexity.

It was noted earlier that our bounds, which were derived for system I, are likely not tight when applied to the actual system, especially in their behavior with respect to the parameters μ, k_d, and L. The dependence of the distribution of computation on μ and k_d, as indicated by the simulation (see chapter 5), does in fact seem weaker than the dependence indicated by the bounds on $pr(T \geq X)$ and C.

CHAPTER 4

PERFORMANCE OF THE HYBRID SCHEME - SOME EXAMPLES

The hybrid scheme was shown in the previous chapter to have an asymptotically pareto probability distribution of the decoding time per block, with an effective pareto exponent of $d\alpha$. Performance curves showing the relationships of the system parameters to the effective pareto exponent are given in this chapter for two interesting channels - the additive white gaussian noise channel and the M - ary erasure channel.

4.1 The Additive White Gaussian Noise Channel

During each use of an additive white gaussian noise channel, a transmitted waveform with energy E is received along with independent gaussian noise whose power spectral density is $N_0/2$ over an infinite bandwidth. This channel is a fairly realistic model for deep space telemetry channels. The signal-to-noise ratio per channel use is defined as $10 \log_{10} E/N_0$ db. The signal-to-noise ratio per bit is an important system parameter, and is defined as

$$\left.\frac{E_b}{N_o}\right|_{db} = 10 \log_{10} \frac{E}{N_o R}$$

where E_b is the signal energy expended per bit of transmitted information, and R is the rate in bits per channel use. (Note that rates in this section are in bits per channel use.)

Shannon has shown that the capacity of the infinite-bandwidth additive white gaussian noise channel is $E/N_0 \log_2 e$ bits per channel use. Stated differently, arbitrarily reliable communication

(through coding) is possible for an additive white gaussian noise channel if and only if the signal-to-noise ratio per bit exceeds

$$-10 \log_{10} (\log_e 2) = -1.6 \text{ db}$$

The magic figure of -1.6db in connection with the additive white gaussian noise channel is sometimes referred to as the Shannon limit.

Practical considerations require that the number of possible transmitted waveforms per channel use be relatively small. It can be shown[20] that in the limit of small signal-to-noise ratios per channel use, binary antipodal (i.e. binary simplex) signal waveforms are optimum. In our example, we assume a signalling scheme in which successive binary digits emerging from the convolutional encoder are transmitted as binary antipodal waveforms with energy E. After each channel use, the received waveform is crosscorrelated with the positive binary waveform by sampling the output of a matched filter. This sample is quantized into one of eight possible levels (three-bit quantization). The cascade of channel waveform generator, additive white gaussian noise channel, matched filter, sampler, and quantizer is equivalent to a discrete memoryless channel with two input letters and eight output letters. This signalling scheme in conjunction with sequential decoding has been studied previously[33,32]. A somewhat ad hoc quantization was chosen, and is sketched in fig. 4-1.

$$T = \sqrt{N_0/2}$$

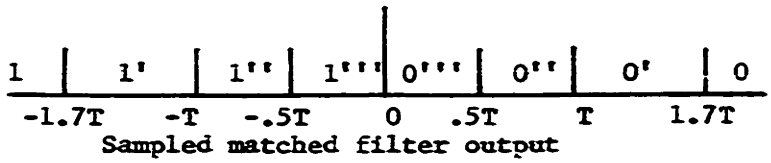

FIG. 4-1 QUANTIZATION SCHEME

For code rates less than 1/6 bits per channel use, the minimum required signal-to-noise ratio per bit is no more than 3/4 db above the Shannon limit. Thus, binary antipodal signalling and eight-level output quantization is simple, but relatively efficient, for code rates less than about 1/6.

For a given signal-to-noise ratio per channel use, the transition probabilities for the equivalent discrete memoryless channel may be calculated. Then, the pareto exponent $\alpha(r)$, corresponding to a particular rate r may be calculated from the relation

$$r = \frac{1}{\ln 2} \frac{E_0(\alpha(r))}{\alpha(r)} \qquad (4.1.1.)$$

where the rate r is in bits per channel use. Fig. (4-2) shows the pareto exponent as a function of rate for a number of signal-to-noise ratios. The data for these curves was calculated with an IBM 7094 computer program supplied by Prof. I.M. Jacobs.

With the aid of the curves of fig. (4-2), we can illustrate the relationship between the effective pareto exponent α_e, of the hybrid decoding scheme and the system parameters: signal-to-noise ratio, rates r and R, and the block length N. If the algebraic code is Reed-Solomon, the effective pareto exponent is given by

$$\alpha_e = \left[N(1-R) + 1 \right] \alpha(r) \qquad (4.1.2.)$$

It may be important to achieve a given effective pareto exponent with the highest possible overall rate rR. In principle, one could fix α_e and N, and use the Lagrange multiplier technique to optimize r and R

in (4.1.2.). However, it turns out to be much simpler to use (4.1.2.) and the curves of fig. (4-2) to plot overall rate rR as a function of Reed-Solomon code rate R for fixed block length, signal-to-noise ratio per channel use and effective pareto exponent.

Representative curves are shown in figs. (4-3), (4-4), and (4-5). Diophantine constraints are ignored in these figures. Each figure is for a different effective pareto exponent, and contains curves for several block lengths. The signal-to-noise ratio per channel use is, in each case, -7.5 db. Note that these curves all have maxima which are very broad and flat. Evidently then, it is not necessary to use extreme care in choosing the algebraic code rate R to maximize rR for given values of N and E/N_0, and α_e, when the hybrid scheme is used in conjunction with the additive white gaussian noise channel. Note also that the optimum value of R increases with block length and decreases with the effective pareto exponent. Fig. (4-6) shows similar curves for various values of E/N_0, with N and α_e fixed at 100 and 1 respectively. The maxima appear to be insensitive to E/N_0.

Although the effective pareto exponent is a useful measure of computational performance, it provides little information as to the average computation per decoded information symbol. The average computation rises sharply as the original pareto exponent $\alpha(r)$ decreases, as indicated by inequalities (3.4.4.), (3.4.5.), and (3.4.6.). Thus, to minimize the average computation, it is desirable to have the rate r as low as possible. To achieve an acceptable average computation, it may then be necessary to operate at a lower overall rate than that shown possible by curves like those of figs. (4-3), (4-3), and (4-5).

The performance of a coding and modulation scheme for the additive

white gaussian noise channel is best shown by curves of required $E_b/E_{b_{min}}$ versus overall rate, where $E_{b_{min}}$ is the minimum required signal energy for a given value of N_0 guaranteed by the Shannon limit. Such curves are shown in figs. (4-7), (4-8), and (4-9) for effective pareto exponents of 1.0, 2.0, and 10.0, respectively. The values of R used in calculating these curves were the optimum values for $10 \log_{10} E/N_0 = -7.5$ db, taken from figs. (4-3), (4-4), and (4-5). A point on the curves was obtained by fixing α_e, N, R, E/N_0, and r, then finding $\alpha(r)$ from fig. (4-2), and finally using the relation

$$\left.\frac{E_b}{E_{b_{min}}}\right|_{db} = 10 \log_{10} \frac{E}{N_0 r R} + 1.6$$

The capacity of the equivalent discrete memoryless channel is less than that of the infinite-bandwidth white gaussian noise channel. The upward trend of all the curves reflects the fact that binary antipodal signalling is less efficient at higher rates and signal-to-noise ratios per channel use.

These curves show that for fixed overall rate and effective pareto exponent, substantial decreases in $E_b/E_{b_{min}}$ can be obtained by increasing the block length N. For example, with a rate of 0.1 bits per channel use and an effective pareto exponent of 2.0, the hybrid scheme with N = 100 achieves a gain of almost 3.5 db over a single conventional sequential decoder. Also, with N = 100, the hybrid scheme approaches the Shannon limit to within 1.6 db, while achieving an effective pareto exponent of 2.0. As mentioned before, however, operation at a rate that is relatively close to channel capacity may mean a very large (but finite) average number of computations per decoded bit.

These curves illustrate the fact that one can improve the asymptotic behavior of the distribution of computation by increasing N, without requiring an increase in signal-to-noise ratio per bit. Even relatively moderate block lengths like 10 or 30 can yield substantial improvement over conventional sequential decoding systems, especially when the required effective pareto exponent is high.

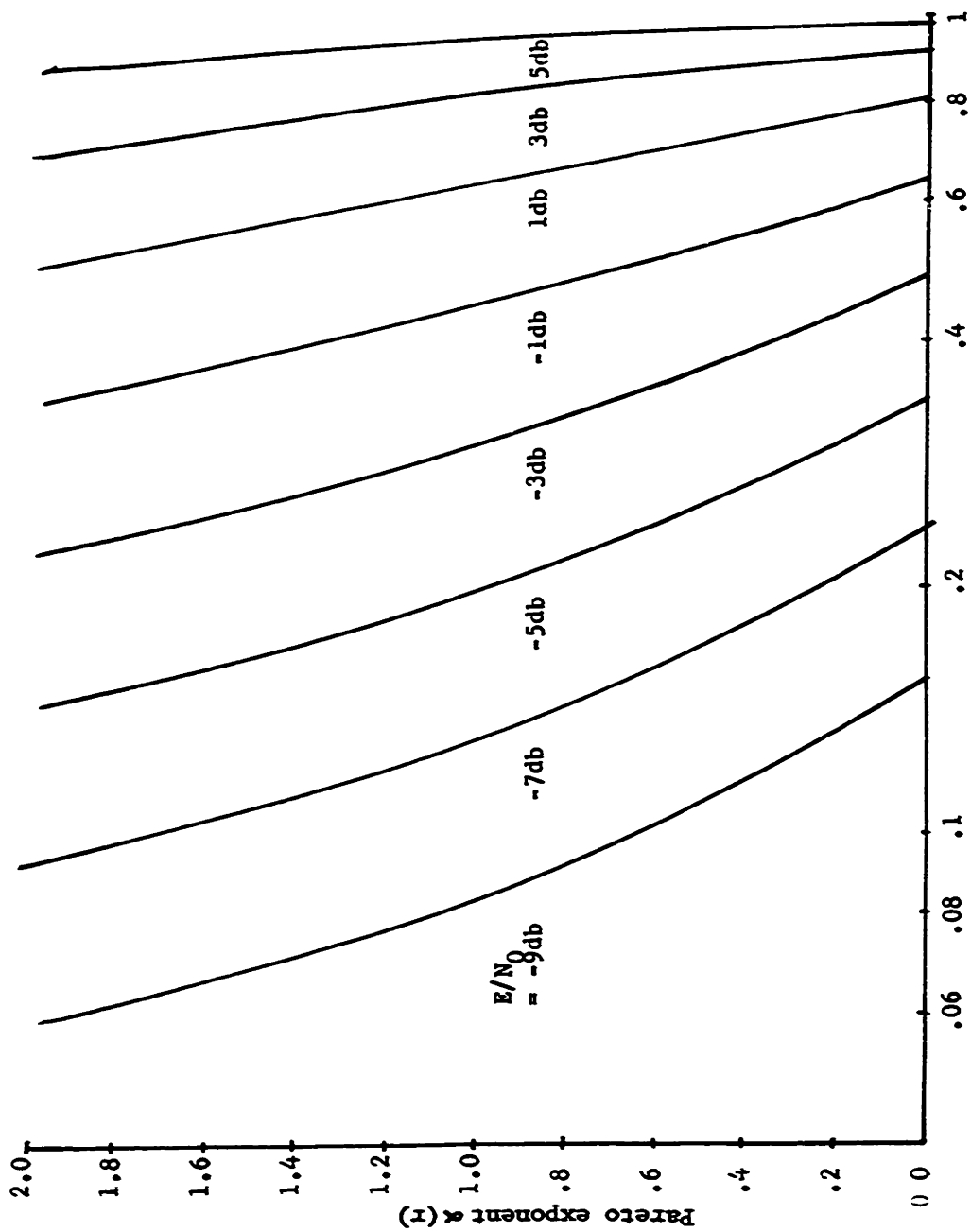

FIG. 4-2 $\alpha(r)$ FOR THE QUANTIZED GAUSSIAN CHANNEL WITH BINARY ANTIPODAL SIGNALLING

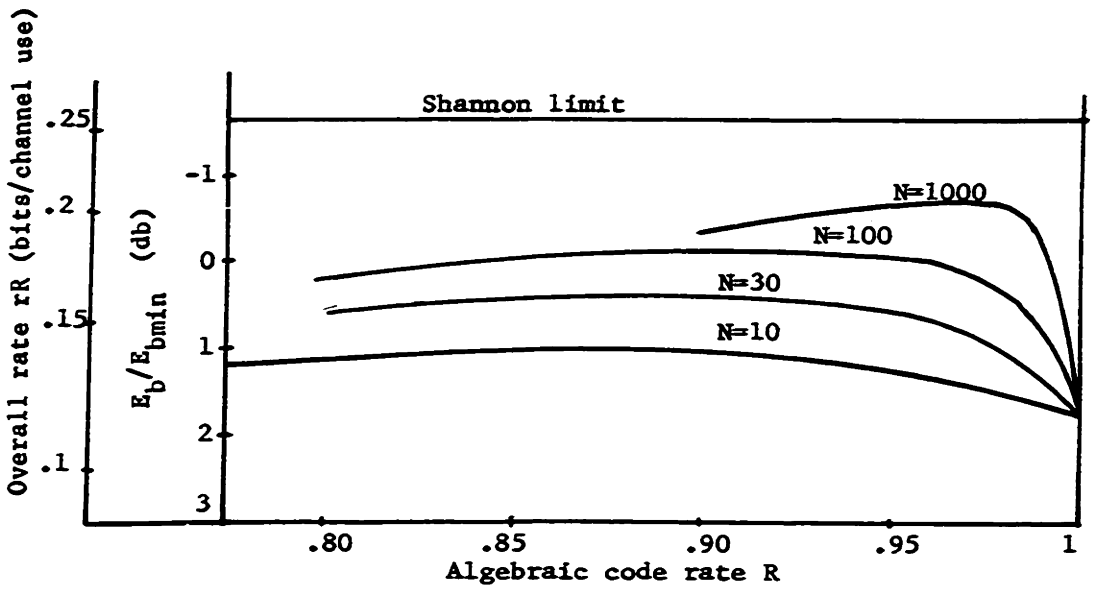

FIG. 4-3 rR VERSUS R TO YIELD $\alpha_e = 1.0$
FOR $E/N_0 = -7.5$ db

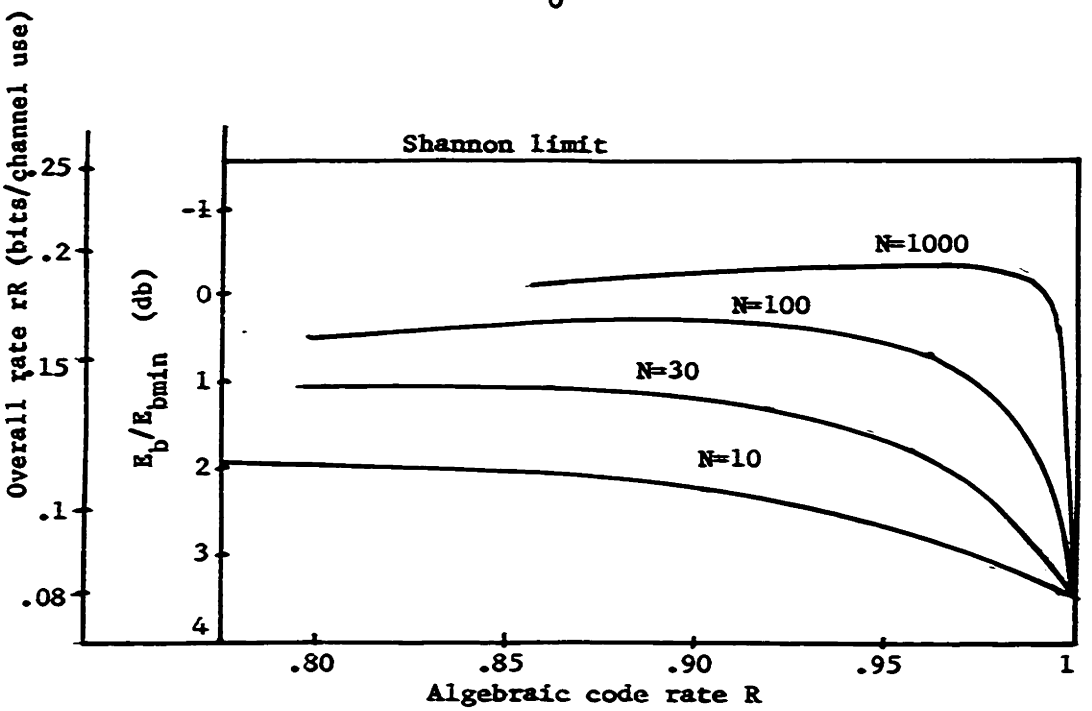

FIG. 4-4 rR VERSUS R TO YIELD $\alpha_e = 2.0$
FOR $E/N_0 = -7.5$ db

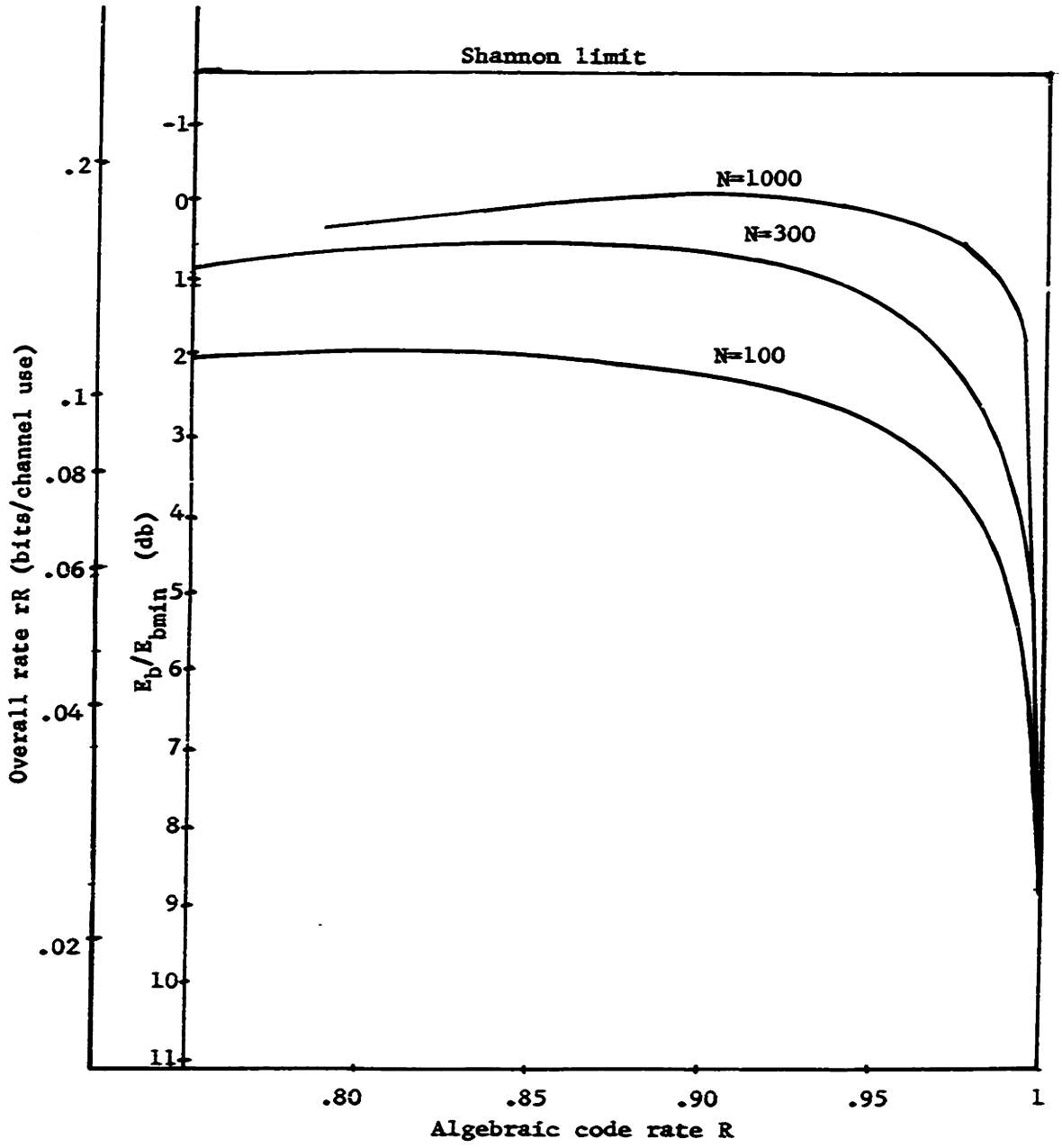

FIG. 4-5 rR VERSUS R TO YIELD $\alpha_e = 10.0$ FOR $E/N_0 = -7.5$ db

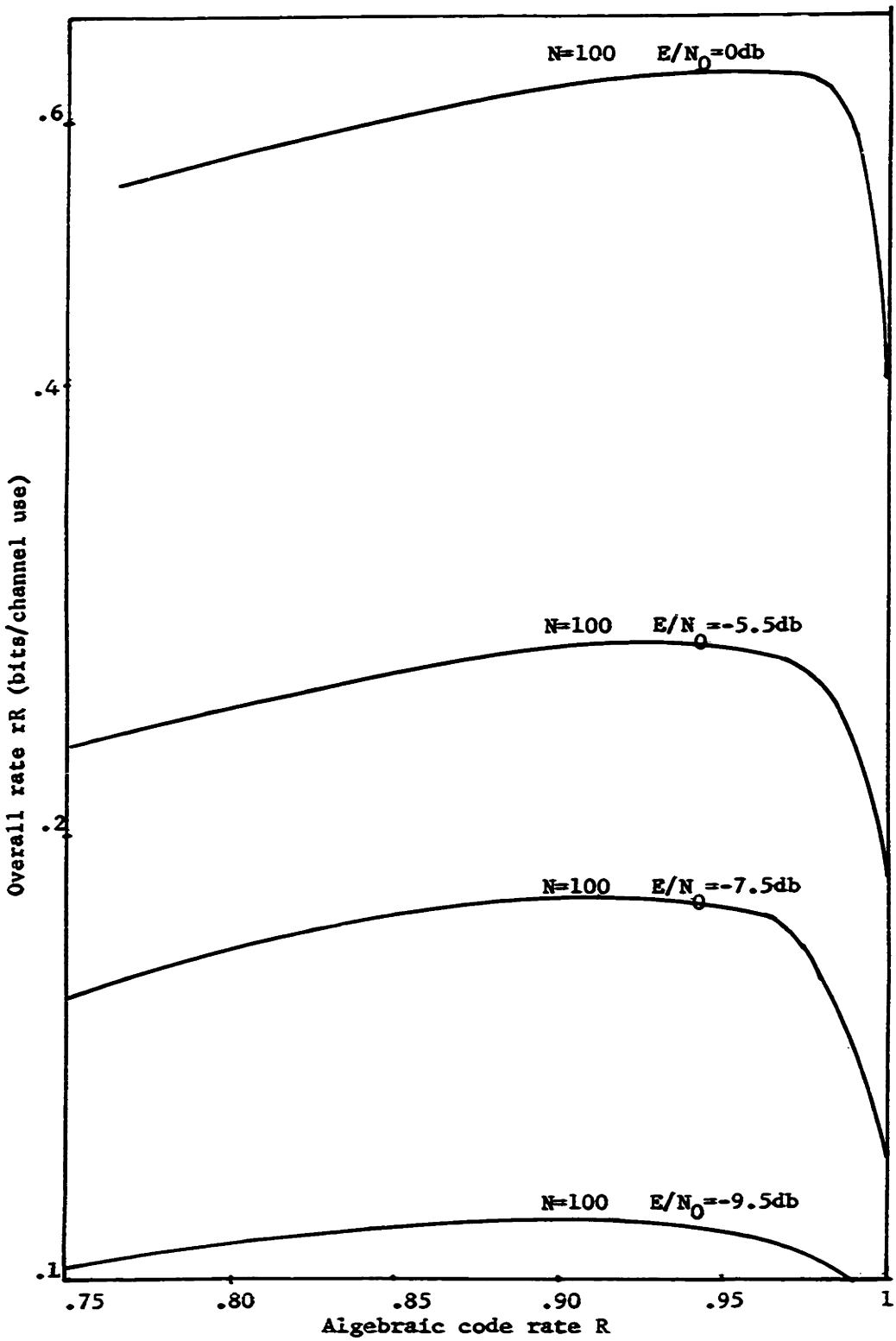

FIG. 4-6 rR VERSUS R TO YIELD $\alpha_e = 1.0$ FOR VARIOUS SIGNAL-TO-NOISE RATIOS

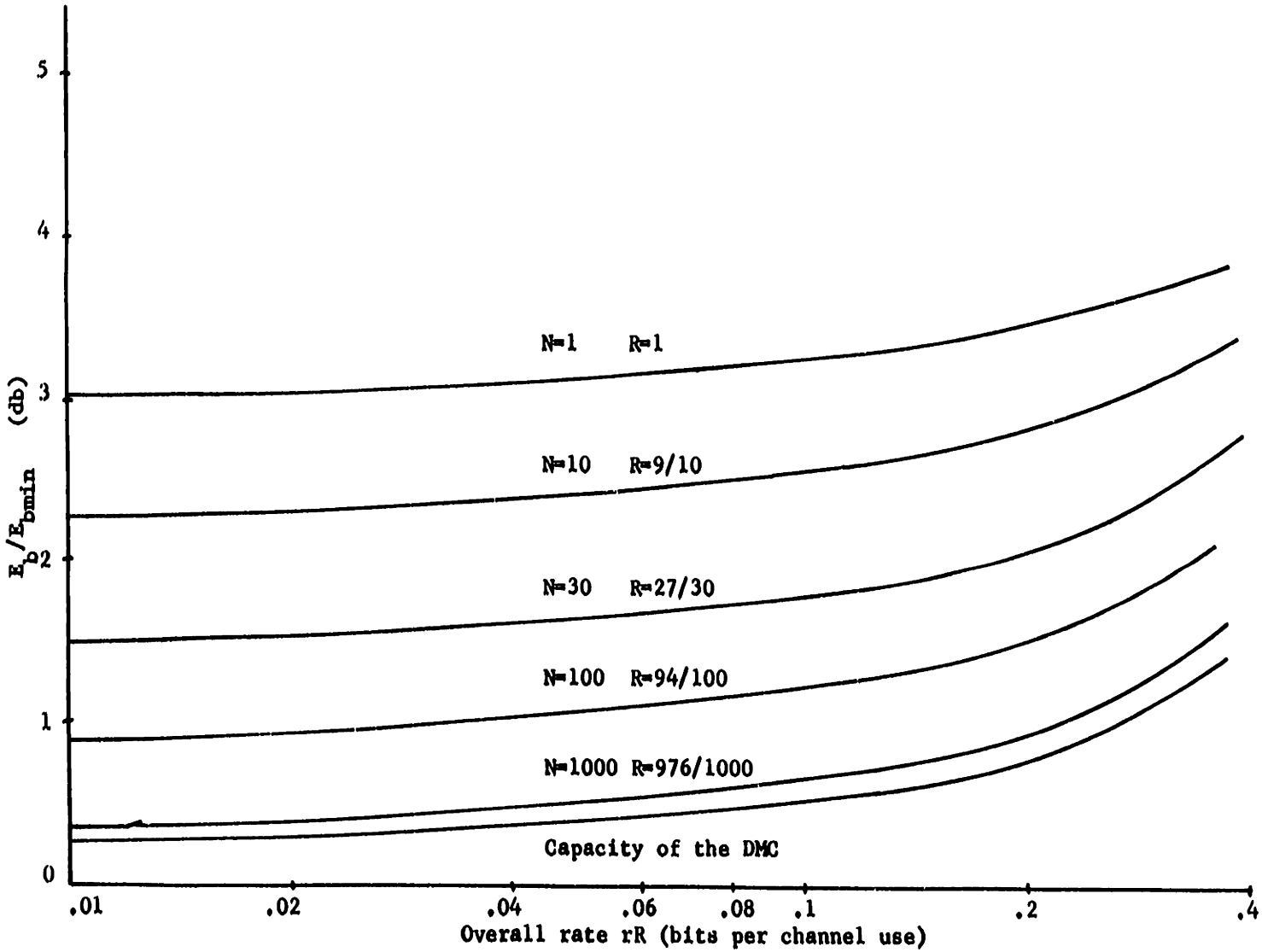

FIG. 4-7 E_b/E_{bmin} VERSUS rR TO YIELD $\alpha_e = 1.0$

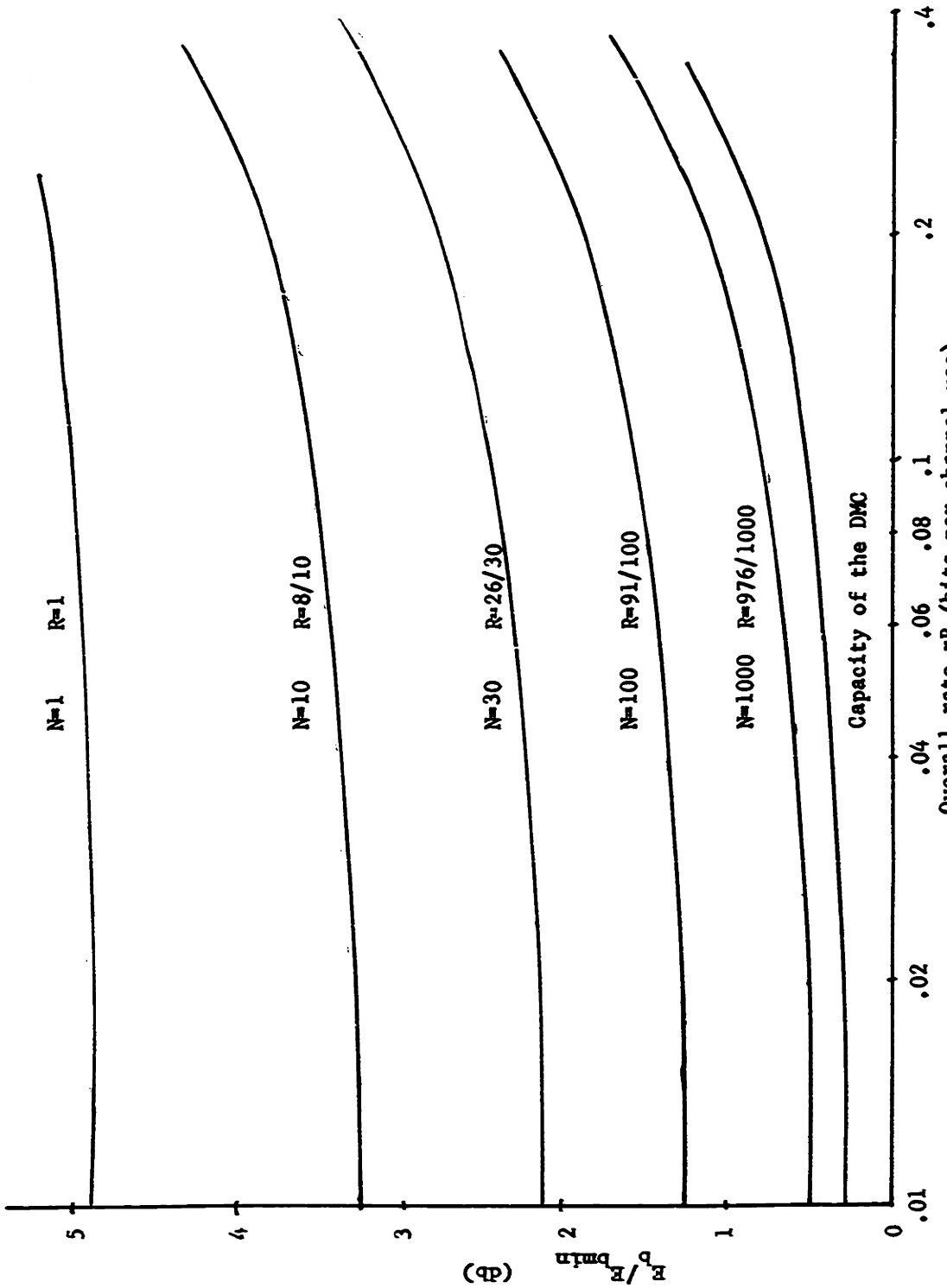

FIG. 4-8 E_b/E_{bmin} VERSUS rR TO YIELD $\alpha_e = 2.0$

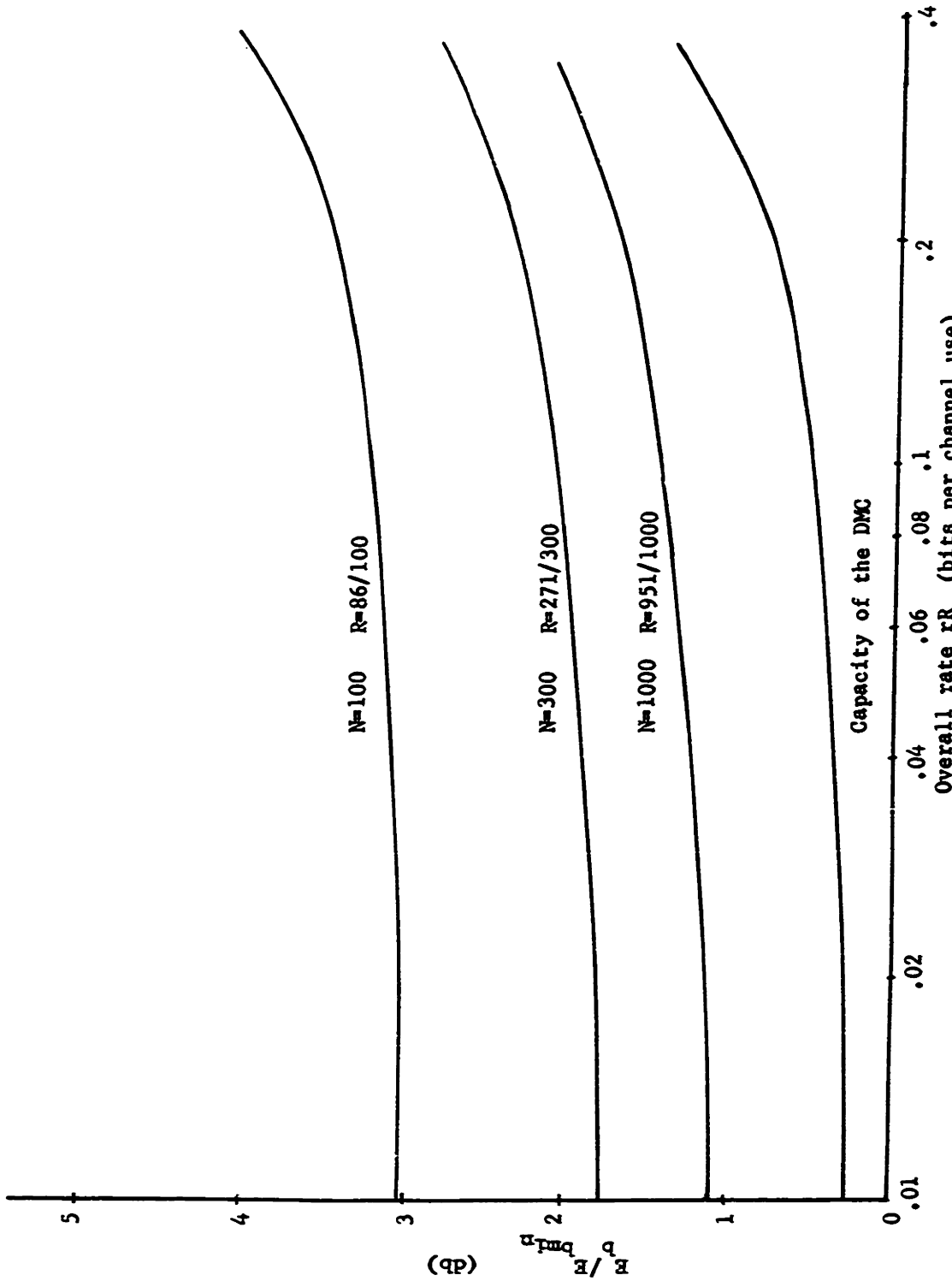

FIG. 4-9 E_b/E_{bmin} VERSUS rR TO YIELD $\alpha_e = 10.0$

4.2 M-ary Erasure Channel

For most discrete memoryless channels that are models for real communication channels, $R_{comp} \geq 1/2\ C$, where C is channel capacity. In this section, we consider an M-ary erasure channel, with M = 1024, for which $R_{comp} < 1/2\ C$. The M-ary erasure channel, illustrated in fig. (4-10), is not a good model of any known practical communication channel. However, it may be considered a crude model of diversity transmission of orthogonal signal waveforms over a fading dispersive channel, for which, at least in certain cases, $R_{comp} << 1/2\ C$. [20,35]

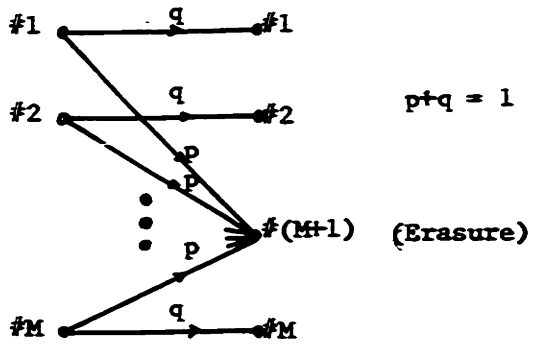

FIG. 4-10 M-ARY ERASURE CHANNEL

The chief merit of the M-ary erasure channel is its mathematical simplicity. The performance data to be presented for the hybrid scheme used in conjunction with a 1024-ary erasure channel may well be typical of DMC's derived from real channels for which R_{comp} is much less than capacity. Actually, for the M-ary erasure channel, a Reed-Solomon coding scheme or a modification of Epstein's algebraic scheme[36] for the binary erasure channel may be more suitable than a scheme employing sequential decoding.

Referring to fig. (4-10), we pick $p = q = 1/2$. Then the transition probabilities are

$$q_{ij} = \begin{cases} \frac{1}{2} & (i = j;\ j = 1,2,\ldots M) \\ 0 & (i \neq j;\ i = 1,2,\ldots M,\ j = 1,2,\ldots M) \\ \frac{1}{2} & (i = 1,2,\ldots M;\ j = M+1) \end{cases}$$

By symmetry, the input probability distribution that maximizes the average mutual information and $E_0(s)$ is

$$p_i = 1/M \quad ; i = 1,2,\ldots M$$

Then, it is easy to show that channel capacity C is given by

$$C = 1/2 \ \ell n\, M \text{ nats per channel use,} \tag{4.2.1.}$$

and that $E_0(s)$ is given by

$$E_0(s) = \ell n\, 2 - \ell n\, (1 + M^{-s}), \tag{4.2.2.}$$

so that

$$E_0(1) = R_{comp} = \ell n\, 2 - \ell n\, (1 + M^{-1})$$
$$<< 1/2\ C \quad \text{for large M.}$$

The pareto exponent α (r), is given by the relation

$$r = \frac{E_0(\alpha(r))}{\alpha(r)} \tag{4.2.3.}$$

and is plotted in fig. (4-11) for the case $M = 1024$, $p = 1/2$. Note that R_{comp} is about 1/5 C in this case. Note also that for $r > R_{comp}$,

$\alpha(r)$ decreases rather slowly with r, approaching channel capacity with almost zero slope.

Figures (4-12), (4-13), and (4-14) correspond to figures (4-3), (4-4), and (4-5) of the previous section, and show the overall rate rR versus R for fixed N and α_e. Again, the maxima are broad and flat. For sufficiently large values of N, operation up to channel capacity is possible with any given effective pareto exponent. However, operation at rates which are substantial fractions of channel capacity is at the expense of relatively tiny values of $\alpha(r)$ and correspondingly large values of average computation. Some representative values of $\alpha(r)$ are marked on the curves of fig. (4-12). These curves suggest that the improvement afforded by increasing the block length of the hybrid scheme is dramatic for channels whose R_{comp} is much less than capacity.

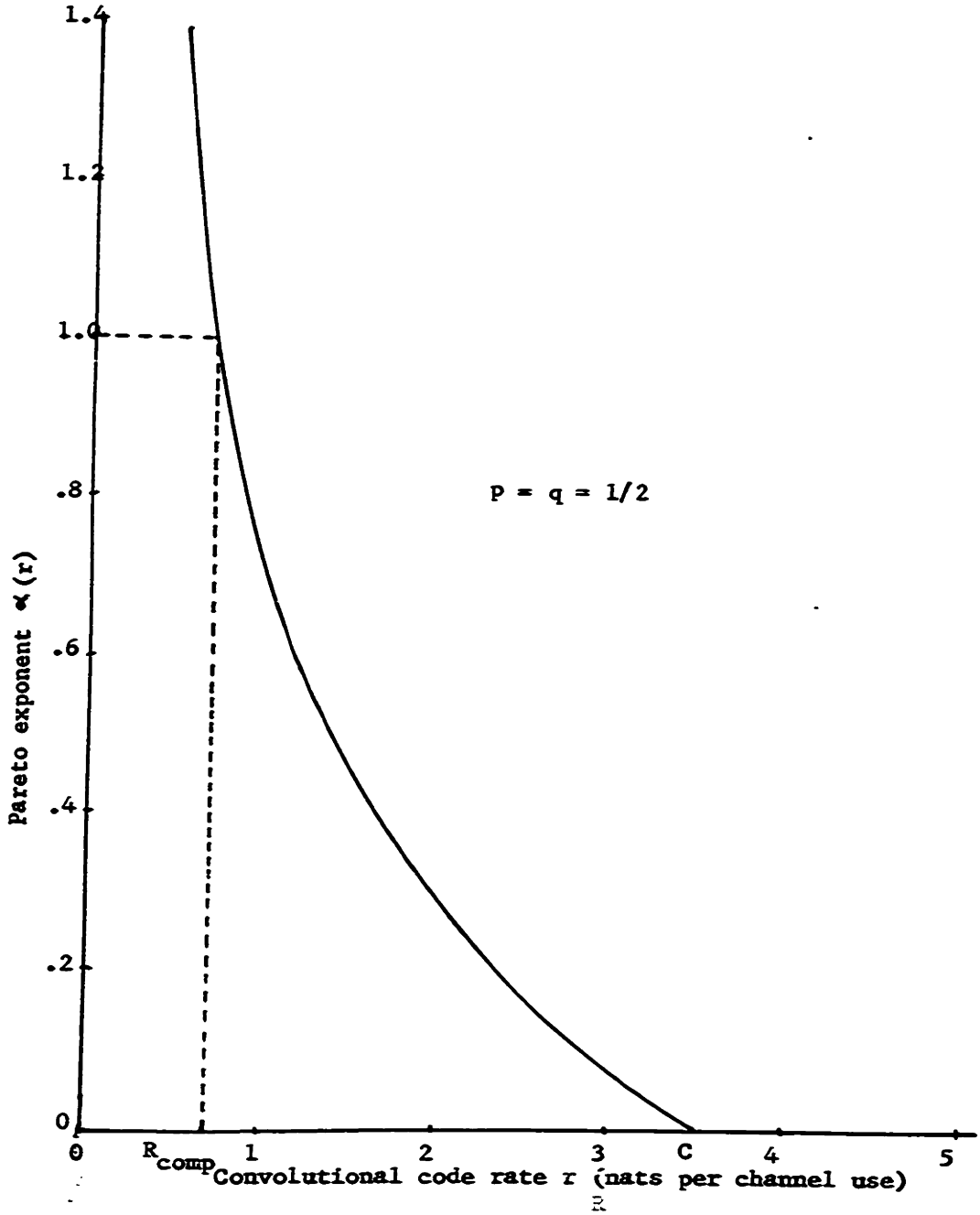

FIG. 4-11 PARETO EXPONENT ∝(r) FOR 1024-ARY ERASURE CHANNEL

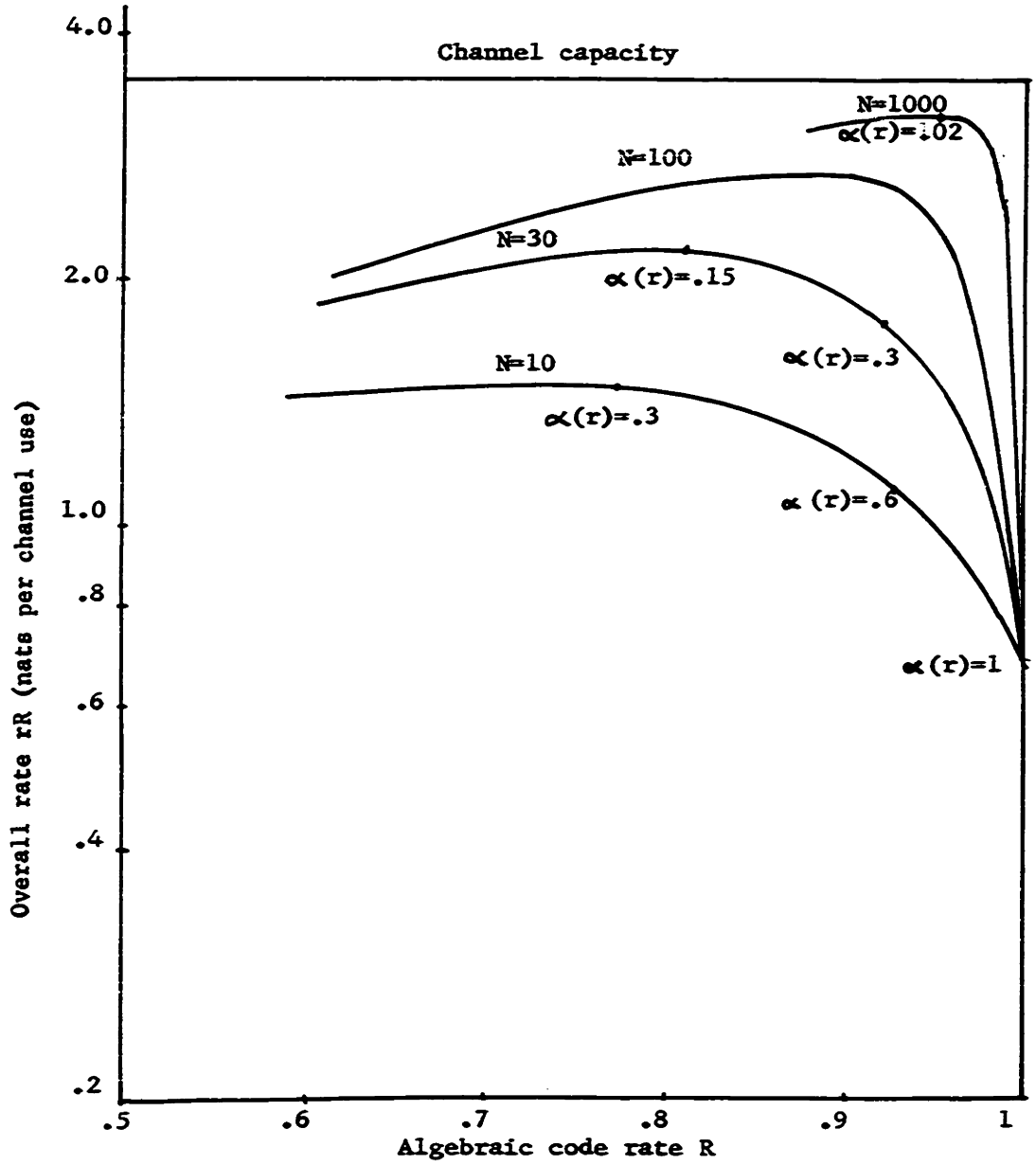

FIG. 4-12 rR VERSUS R TO YIELD α_e = 1.0 FOR THE 1024-ARY ERASURE CHANNEL WITH p=q=1/2

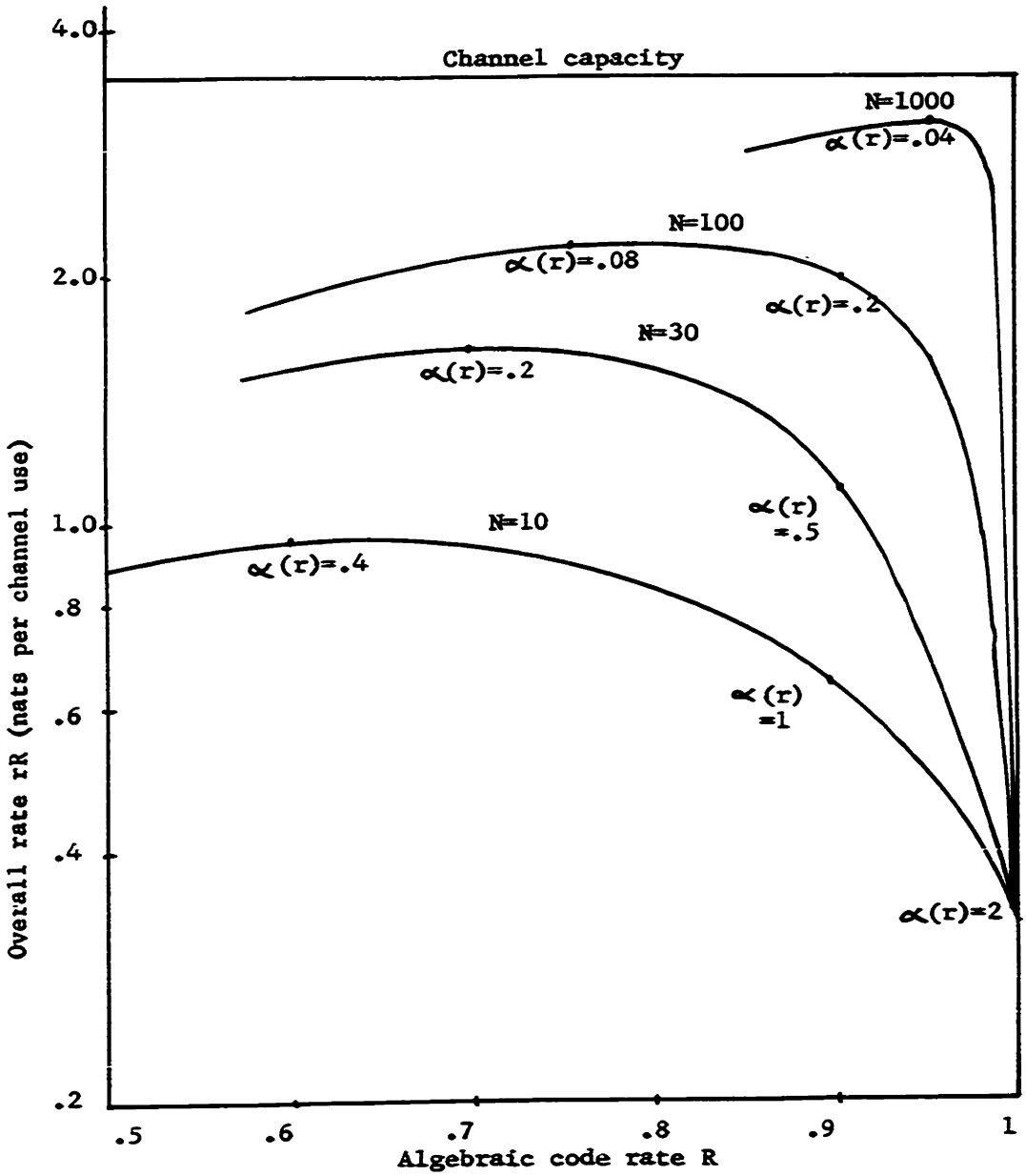

FIG. 4-13 rR VERSUS R TO YIELD α_e =2.0 FOR THE 1024-ARY ERASURE CHANNEL WITH p=q=1/2

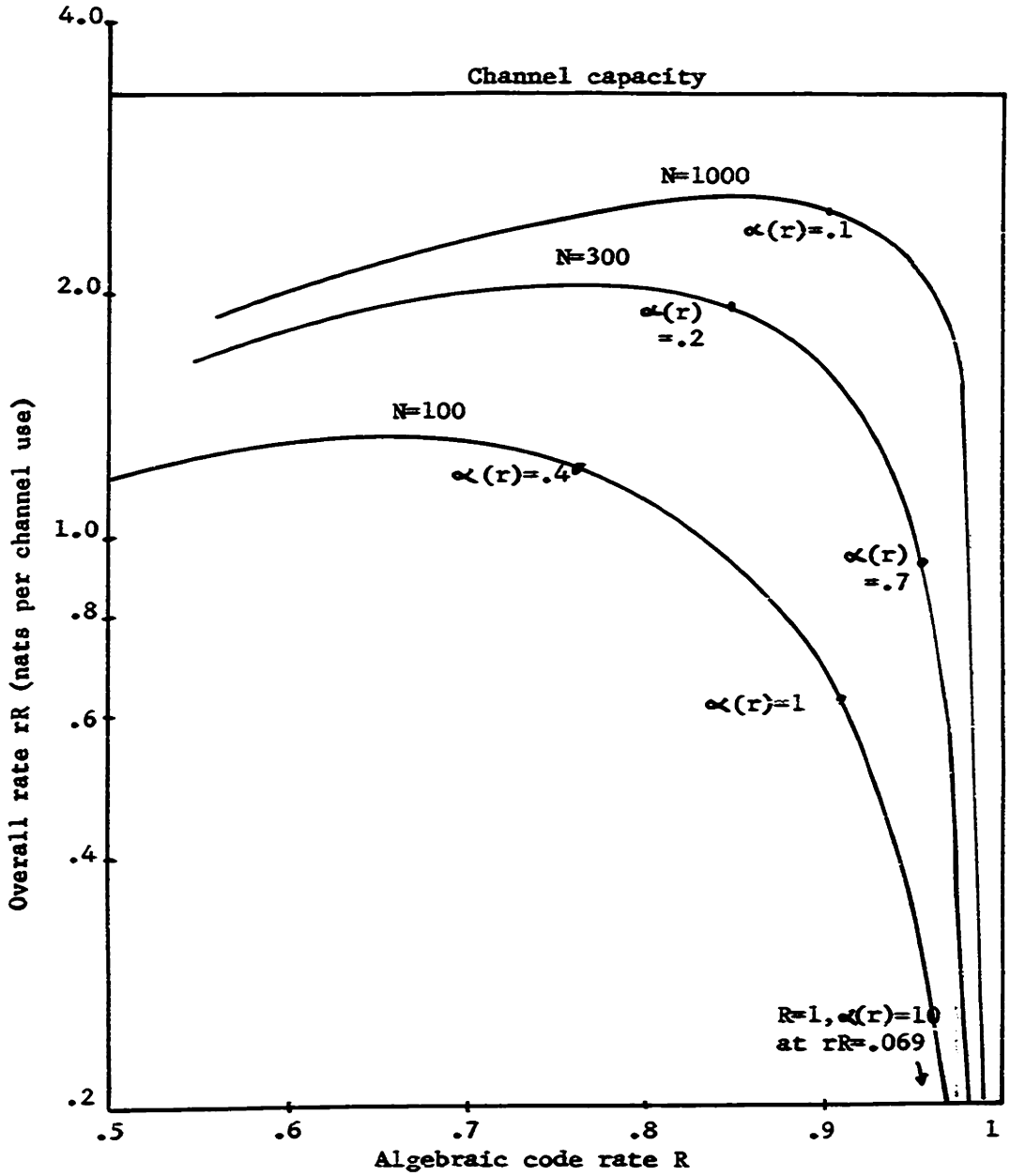

FIG. 4-14 rR VERSUS R TO YIELD α_e = 10.0 FOR THE 1024-ARY ERASURE CHANNEL WITH p=q=1/2

CHAPTER 5

SIMULATION OF THE HYBRID SCHEME

Although the upper bounds derived in Chapter 3 are interesting from a theoretical standpoint, they are likely weak, especially in regard to the coefficients and to the system parameters k_d and μ. A more precise evaluation of system behavior can be obtained by simulation.

In this chapter, we describe a computer simulation of a hybrid scheme that is basically similar to system II, described in Chapter 3. The simulated DMC was the additive white gaussian noise channel with binary antipodal signal input and eight-level-quantized output. The main purposes of the simulation were to observe the effects of the system parameters on the statistics of computation and to observe the dynamics of the decoding procedure.

5.1 Description of the Simulation

The simulated channel and modulation scheme were described in section (4.1). An earlier simulation of conventional sequential decoding with periodic resynchronization was described in reference 32. The simulation was performed on a PDP-6 computer associated with Project MAC at M.I.T.. This machine has a 2μsec., 16,000-word memory. The simulation of sequential decoding was based on a convenient and flexible simulation facility for sequential decoding developed by Niessen[37].

Since the simulated two-input, eight-output DMC was symmetric from the input, it was convenient and permissible to assume that

only zeros were transmitted. Each channel output was generated with a 12-bit random number generator[38] whose output was quantized into eight levels in accordance with the desired channel transition probabilities.

Basically, the simulation involved the time-sharing of the Fano algorithm (illustrated in fig. (2-5)) by N = 10 separate sequential decoders. The number N was limited by the number of available storage words in the computer memory. (About 300 36-bit words were required to store the current data for each decoder.)

The Fano algorithm was allocated to each decoder in turn; i.e., the algorithm operated on the current data for each decoder in turn. Each decoder was allowed to perform up to 500 computations during its turn. (One computation was defined as one use of box A in fig. (2-5).) Current data for each decoder included path metric, current threshold, flags, the latest 256 path metric increments on the currently-hypothesized path, the latest 256 channel outputs, the latest 256 hypothesized branches, the first incorrectly-hypothesized branch on the current path, and the maximum current penetration into the tree.

The algebraic code word was assumed to consist of nine information bits and one parity bit which was their mod-2 sum. Thus, the minimum distance was two, and one erasure per block could be corrected. It was not necessary to simulate the action of the algebraic decoder, since correctable erasure patterns were easy to recognize. Following each cycle through all ten sequential decoders, and before the beginning of the next cycle, the second-smallest maximum penetration into the tree (designated MINMAX) was determined, and the current

origin (common to all the sequential decoders) was set to a depth of (MINMAX - k_d) nodes. Then, any digits which were not yet decoded by the "farthest-behind" decoder (the one with the smallest maximum penetration into the tree), and which were further back than the newly-determined current origin, were verified or corrected if necessary, as if their correct values had been determined by the algebraic decoder. Any digits further back than the current origin, which were incorrectly hypothesized by other decoders, would be recognized by the program as being capable of causing a catastrophic error. The frequency of such errors decreases exponentially with k_d. If no such error was made, the next cycle through all the sequential decoders would be initiated, after the farthest-behind decoder had, if necessary, been reset to start along the correct path from the current origin.

The simulation was not performed in real time. Channel outputs were not generated periodically but only when needed. Waiting line effects arising from periodically-arriving channel outputs were simulated by using by using a counter, which stored the current total number of received branches. The counter was incremented by one every μ computations, where μ was a measure of the simulated decoder's speed. The current waiting line for any decoder was the total number of received branches divided by ten minus the current depth of penetration for that decoder. Whenever the waiting line of any sequential decoder dropped to zero, the counter was incremented enough to advance the "time" to the arrival of the next branch or to the end of the present decoder's turn, whichever came first.

In other words, during each turn, a decoder was effectively allotted a time equivalent to 500 computations, including "dead time", during which it may have waited for new channel outputs to arrive.

The ten convolutional encoders were assumed to be identical. The code was of rate 1/7 bits per channel use; i.e. six check bits and one information bit constituted a branch. The encoding constraint length was 72, long enough to make the probability of undetectable error minute. The first 24 bits of each convolutional code parity net were constructed in an ad hoc fashion from previously-optimized "front-ends"[32]. The remaining 48 bits were selected at random.

Three different signal-to-noise ratios per channel use were simulated: -6.5 db, -7.0 db, and -7.5 db. With r = 1/7 bits per channel use and R = 9/10, the signal-to-noise ratios per information bit were respectively 2.38 db, 1.88 db, and 1.38 db. The respective pareto exponents $\alpha(r)$, are determined from fig. (4-2) as 1.06, 0.83, and 0.63. Thus, for -7 db, and -7.5 db, the rate 1/7 is significantly above R_{comp}.

The sequential decoders used the normal Fano path metric with the bias U set equal to the rate of the convolutional code. No attempt was made to optimize the threshold spacing, since previous work[30] has indicated that the statistics of computation are not very sensitive to this parameter. A simplified flow diagram of the program is shown in fig.(5-1). The notation is as follows:

N : number of sequential decoders

MINMAX : d^{th} smallest of MAX(1), MAX(2),...MAX(N)

MAX (I) : 1,2,...N: maximum current penetration of I^{th} decoder

k_d : decoding constraint length

NO : current origin

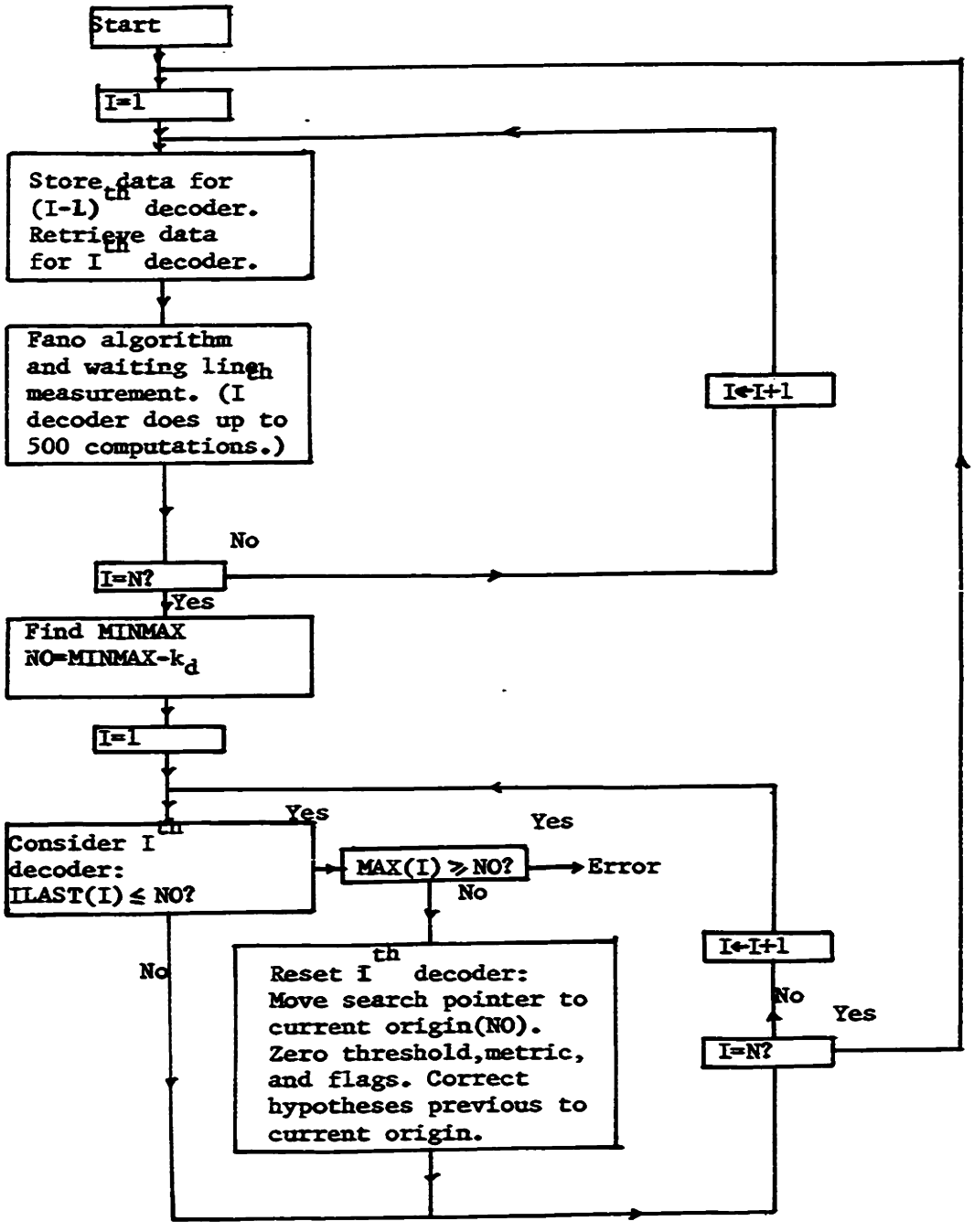

FIG. 5-1 WORD FLOW CHART OF THE SIMULATION PROGRAM

5.2 Results of the Simulation

Several runs were made with various values of E/N_0, k_d, and μ. For each run, the following data was obtained: number of errors, average number of computations per decoded bit, number of times the "farthest-behind" decoder had to be reset, maximum waiting line, maximum search depth, histogram of the search depth, and histogram of the number of computations per search. A search consists of all computations done between first reaching some depth MAX and first reaching depth MAX + 1. The search depth is the maximum number of nodes which a sequential decoder backs up during a search. In addition, waiting lines, forward progress through the trees, occurrences of resets, etc. could be continuously monitored during a run. The display system built into Niessen's sequential decoding facility was not used, because for proper operation it would have required too much memory.

During these runs, no undetectable errors were observed. One detectable error was observed for $10 \log_{10} E/N_0 = -7.5$ db and $k_d = 175$. Evidently, it would have been avoided if the decoding constraint length k_d were larger.

(a) Average Computation

Table (5.1) shows the measured average number of computations per decoded bit for various values of the parameters E/N_0, μ, and k_d. From this table, note the following:

(1) The average computation rises sharply as E/N_0 is lowered, but is less than the sample average for pure sequential decoding.

(2) The sample mean is more variable at -7.5 db than at -6.5 db, for a given sample size.

TABLE 5.1 AVERAGE COMPUTATION FOR r = 1/7, R = 9/10

$10 \log_{10} E/N_0$ (db)	μ	k_d	MEASURED AVERAGE COMPUTATION PER BIT	TOTAL NUMBER OF BITS DECODED
-6.5	40	100	5.4	1,100,000
-6.5	20	100	5.5	134,000
-6.5	1***	100	5.5	190,000
-6.5	20	255	5.7	281,000
-6.5	500	100	5.2	55,000*
-7.0	50	150	11.7	162,000
-7.0	50	**	22	154,000
-7.5	200	175	58	270,000
-7.5	100	175	47	143,000
-7.5	200	255	45	157,000
-7.5	100	**	78	74,300

* : No resets were observed during this run.

**: Simulation of pure sequential decoding with no help from any algebraic decoder; i.e. no resets.

***: μ was less than the average computation. The waiting line for each decoder was continually increasing.

(3) The average seems relatively insensitive to the parameters μ and k_d. For 10 $\log_{10} E/N_0$ = -6.5 db and μ = 500 the simulated machine speed was so much faster than the average computational load that the maximum observed waiting line was roughly equal to the maximum observed search depth. At any time, the earliest known incorrect digit never was further back than the current origin; consequently no resets were observed.

(b) Dynamics of the Hybrid Decoder

The observed number of resets for the various runs ranged between one and ten per 10,000 decoded bits. Typically, a sequential decoder was reset once or twice in succession, indicating that it had been held up by a small "cluster" of relatively improbable channel outputs. However, upwards of five successive resets of a single decoder were occasionally observed. Once the current origin was moved past the cluster of noisy channel outputs, the recently reset sequential decoder typically moved forward very quickly through the tree to rejoin the other further-ahead decoders.

Fig. (5-2) dramatically illustrates this behavior. It shows the waiting line of one of the sequential decoder building up as new branches are received. (The remaining decoders all had waiting lines which were close to zero during this period.) When the remaining decoders had moved far enough ahead of this lagging decoder, it was reset with the aid of the algebraic decoder;almost immediately its waiting line dropped to zero.

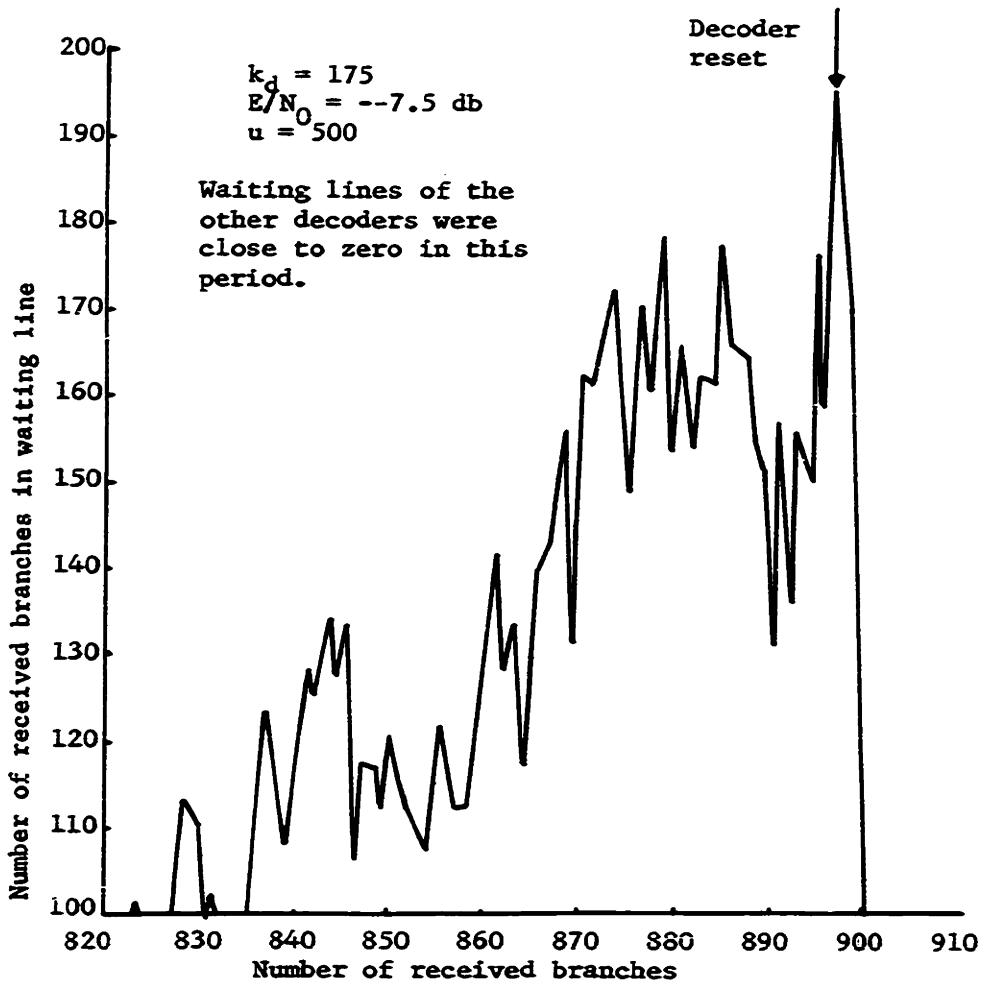

FIG. 5-2 WAITING LINE BUILD-UP OF ONE OF THE DECODERS

Fig. (5-3) shows the waiting lines of all ten sequential decoders over a span of 600 received branches. Arrows indicate resets. Up to the 550th received branch, the waiting lines of decoders #3, #7, and #8 were rising. Then, decoder #8 was able to move past its troublesome span of noisy channel outputs, and its waiting line dropped to zero. The current origin then happened to move far enough ahead that decoder #3 could be reset, and its waiting line dropped to zero. The only lagging decoder was then #7, which was soon reset, allowing its waiting line to drop to zero. Needless to say, situations in which two or more different decoders were reset in quick succession were rare.

It is interesting to compare the waiting line behavior of the simulated hybrid system to that of pure sequential decoding at rates above R_{comp}. The waiting lines for pure sequential decoding would be expected to exhibit a steadily increasing trend. When pure sequential decoding (R = 1) was simulated at -7.0 db, for example, sequential decoder #5 had only penetrated about 3000 branches into its tree by the time all of the other decoders had penetrated at least 20,000 branches into their respective trees.

(c) Histograms of Computations per Search and of Search Depth

Fig. (5-4) shows cumulative histograms of the number of computations per search obtained for the three different signal-to-noise ratios. On logarithmic paper, a pareto distribution should appear as a straight line with a slope equal to the negative of the pareto exponent. Note that the dotted curves for pure sequential decoding

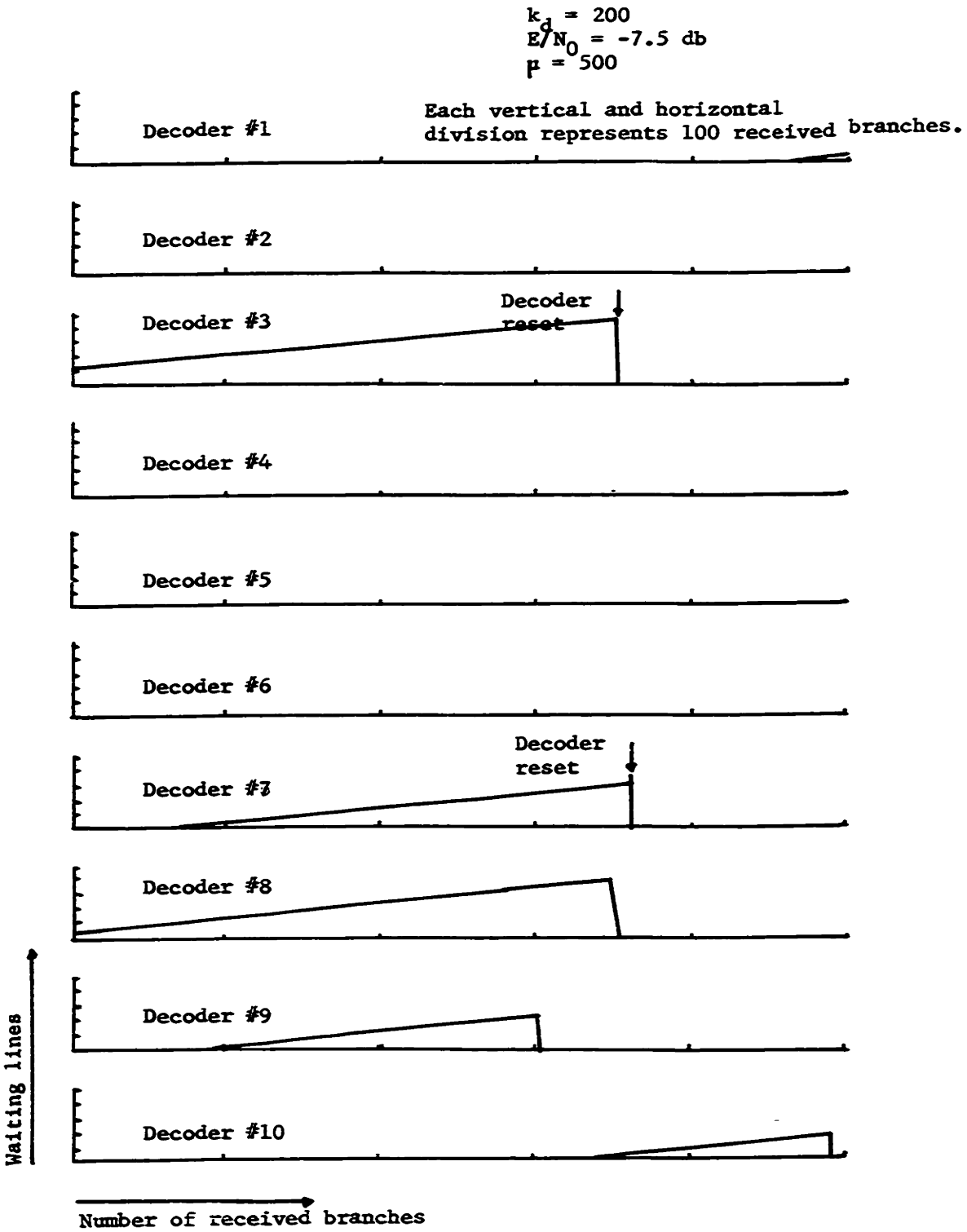

FIG. 5-3 WAITING LINES OF THE TEN DECODERS

and the upper portion of the solid curves for the hybrid scheme have slopes which agree fairly well with the pareto exponents $\alpha(1/7)$ for the three signal-to-noise ratios. Since the minimum distance of the single parity check block code was two, the effective pareto exponent for the hybrid scheme would be expected to be twice as large as $\alpha(1/7)$. It was difficult to obtain an accurate estimate of the tail of the distribution within a reasonable (less than say, five hours) amount of computer time. However, the measured slopes of the tail of the histograms agree with the predicted values $2\alpha(1/7)$, to within about 10%. Fig. (5-5), showing the histograms for three separate runs at -7.5 db, illustrates the uncertainty in the tail behavior due to the relatively small sample sizes used.

Histograms were obtained for various values of the parameters μ and k_d. The resulting variations in the tails of the observed distributions were of the same order of magnitude as the variations between different runs with the same parameters. Thus, there is little evidence to suggest that the distributions are very sensitive to μ and k_d. Figs. (5-6) and (5-7) show histograms for different values of k_d and μ respectively.

A hybrid scheme employing a rate 8/10 algebraic code with minimum distance d = 3 was also simulated. A Reed-Solomon code on a 16-letter alphabet with 8 information symbols and two check symbols per code word could be used for such a code. Then, a b-symbol would consist of four binary information symbols. In the simulation, the requirement that erased bits be algebraically decoded in groups of four was ignored for convenience. It was expected that

the effect on the observed histogram would be negligible. Fig. (5-8) shows the histogram that was obtained. As expected, the slope on the tail roughly corresponded to an effective pareto exponent of $3\alpha(1/7)$.

The second type of cumulative histogram that was obtained from each run was that of the search depth. As shown in fig. (5-9), the distribution appears to be of the exponential type in the observed range of search depths. The observed distribution for the hybrid scheme appeared little different from that for pure sequential decoding in this range. Thus, such curves may provide a basis for an appropriate choice of the decoding constraint length k_d.

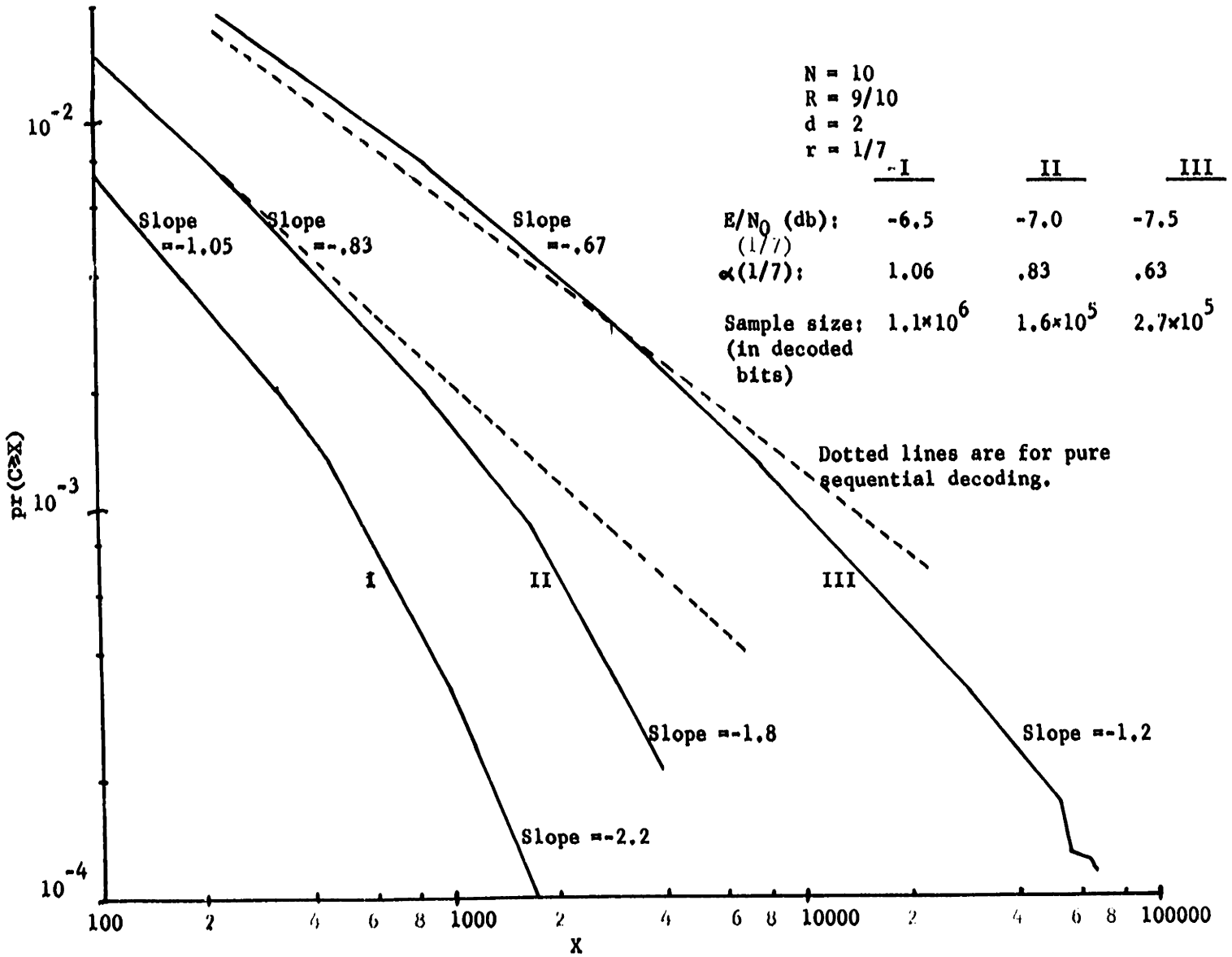

FIG. 5-4 EMPIRICAL DISTRIBUTION OF COMPUTATIONS PER SEARCH, C

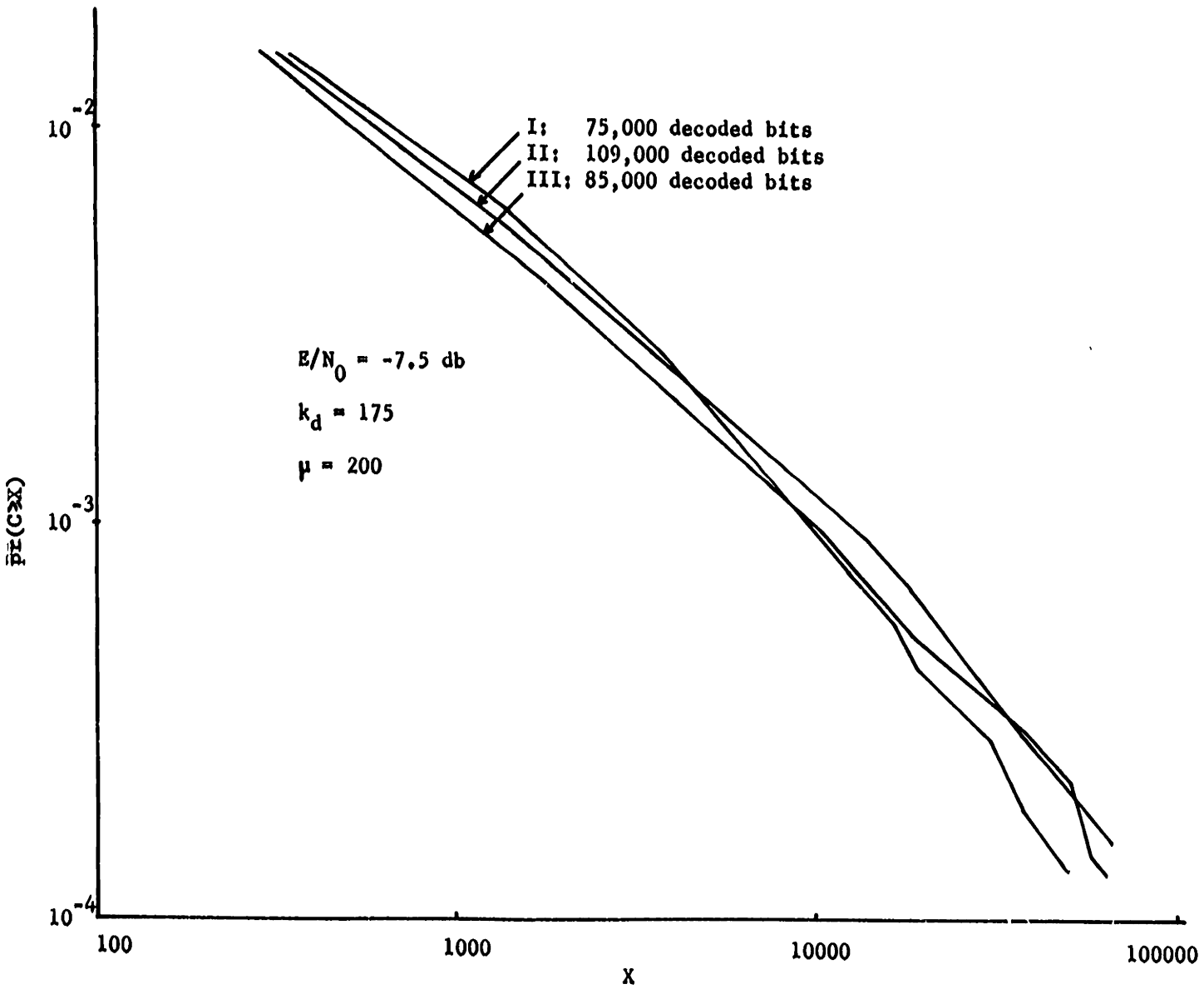

FIG. 5-5 COMPUTATION DISTRIBUTIONS OBTAINED FOR THREE RUNS WITH IDENTICAL PARAMETERS

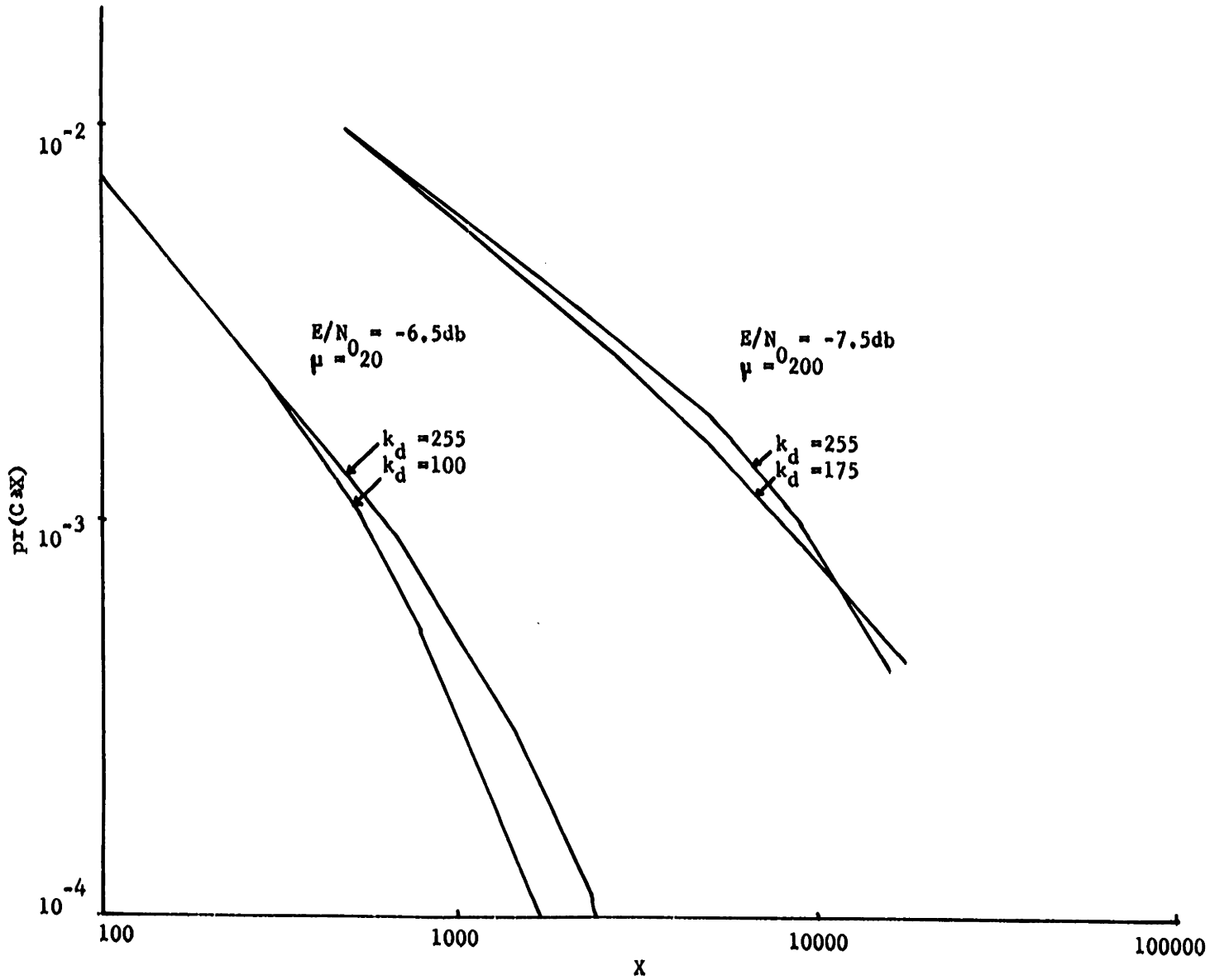

FIG. 5-6 EFFECT OF VARYING k_d

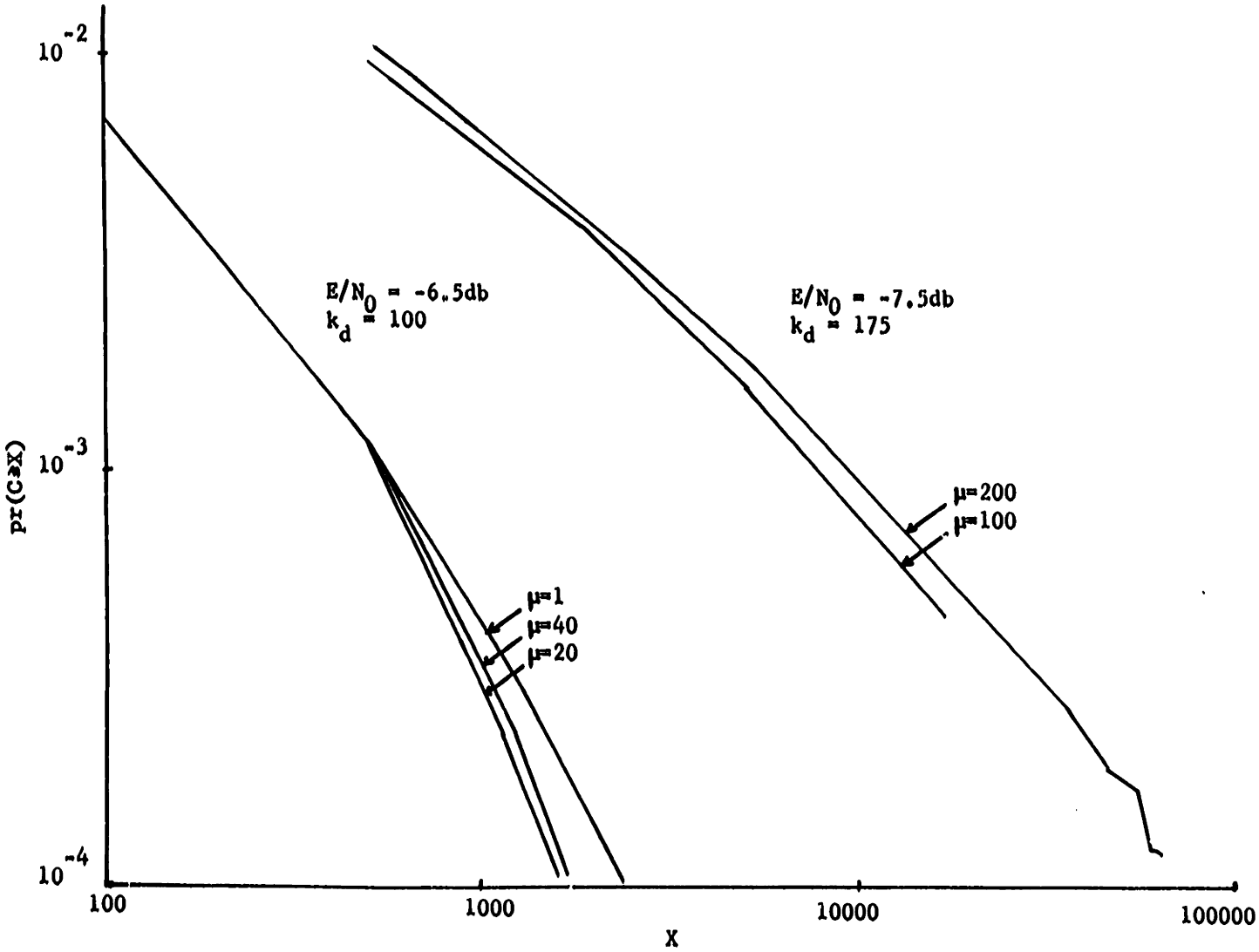

FIG. 5-7 EFFECT OF VARYING μ

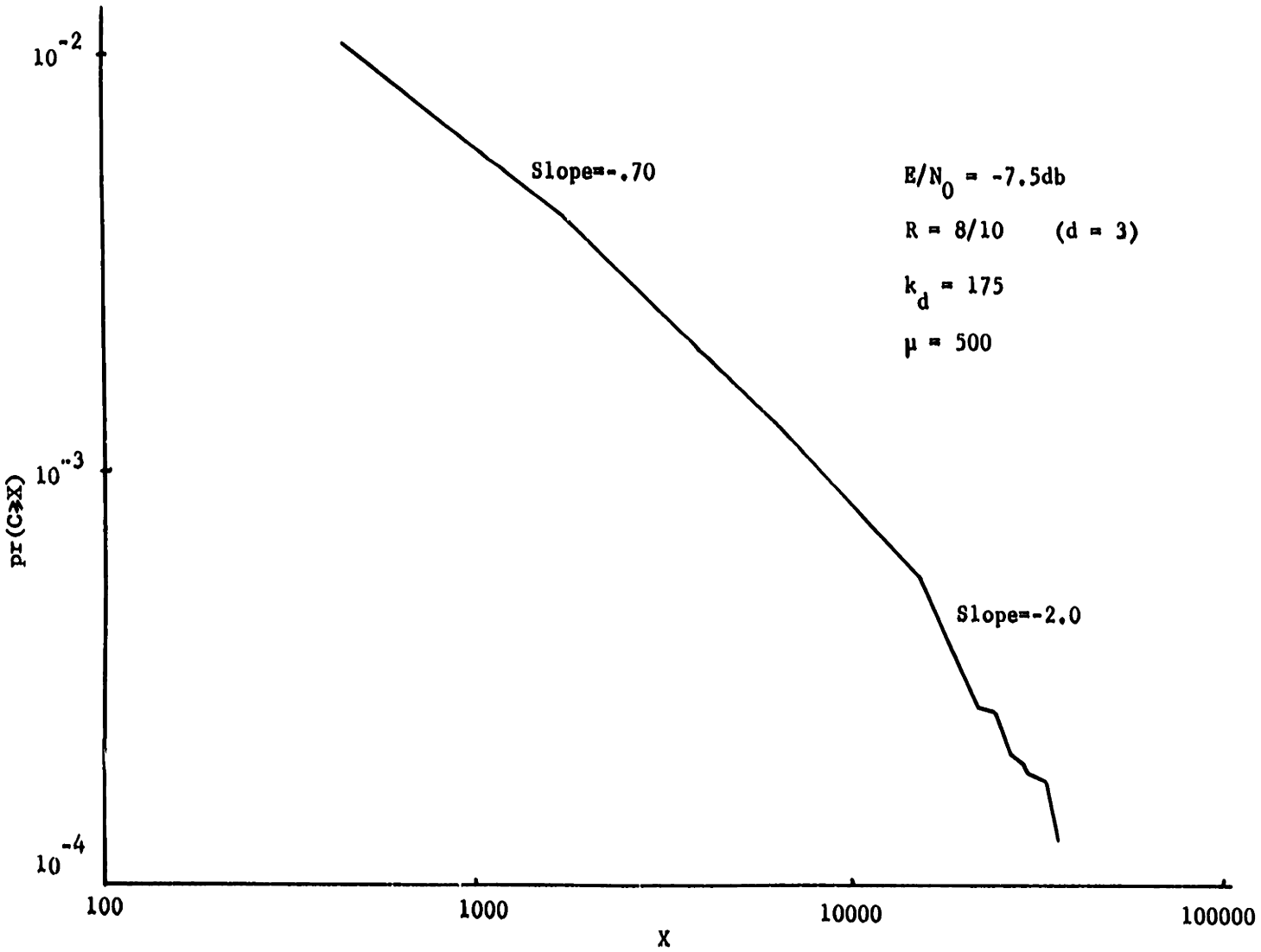

FIG. 5-8 DISTRIBUTION OF COMPUTATION WITH d = 3

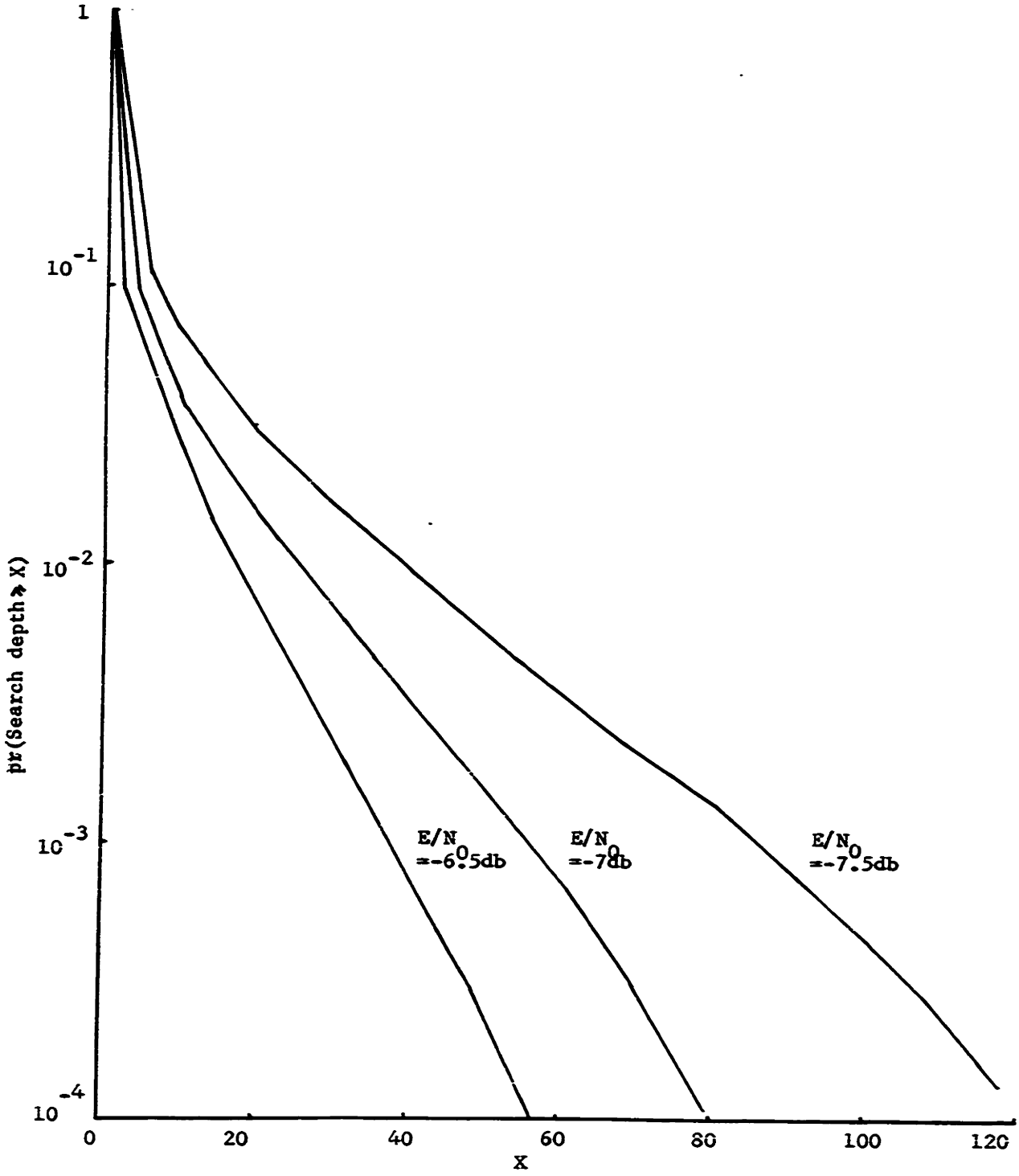

FIG. 5-9 EMPIRICAL DISTRIBUTION OF SEARCH DEPTH

5.3 Summary

The simulation confirmed that the use of the hybrid scheme, even with an algebraic code block length as short as ten, can operate above the "computational cutoff" rate R_{comp}, with an effective pareto exponent greater than unity. The effect of the algebraic decoder was to be continually "nipping at the heels" of the most-lagging sequential decoder, thus causing it to advance faster through its tree. Serious waiting line build-ups tended to be curtailed, and searches requiring large number of computations were rarer.

The results also tended to confirm our previous conjecture that the computation statistics are not nearly as sensitive to the parameters μ and k_d as the upper bounds of Chapter 3 would indicate.

CHAPTER 6

IMPLEMENTATION AND DESIGN OF THE HYBRID SCHEME

In this chapter, we outline some approaches to the instrumentation and parameter specification for the hybrid sequential and algebraic scheme. It should be emphasized that the hybrid scheme is not simple and inexpensive relative to other known coding schemes. Nevertheless, such complex digital systems for extremely reliable communication are becoming increasingly feasible, as integrated digital circuits continue to become cheaper and more reliable.

6.1 Implementation of the Encoder

The encoder includes N separate convolutional encoders. A resonable supposition, substantiated by the results of the simulation, is that the use of N identical time-invariant convolutional encoders causes no appreciable degredation of performance. The N identical convolutional encoders are then implemented conveniently as one convolutional encoder which contains a shift register whose output taps are spaced at intervals of N stages, rather than of one stage. In this way, the i^{th} branch produced by the encoder ($i = 1,2,...N$) and branches which follow it by multiples of N, are essentially generated by the i^{th} convolutional encoder and are independent of branches generated by the other (N-1) convolutional encoders. Note that the length of the shift register required by this configuration is N times the encoding constraint length of each individual convolutional encoder, but that the number of output taps from the shift

register is independent of N. This is an important consideration, since the cost of shift registers generally depends rather strongly on the number of input/output connections.

Fig. 6-1 illustrates this encoder configuration, including the algebraic encoder. In the diagram, each shift register stage is assumed to store the equivalent of one convolutional encoder input information symbol ($\log_2 u$ bits). All dimensions are in terms of numbers of stages. The algebraic encoder, whose implementation is straightforward[8], is assumed to generate a succession of b-symbols, each equivalent to b u-ary information symbols. For each input information symbol, the convolutional encoder generates v channel input symbols, by means of the v parity nets and waveform generator.

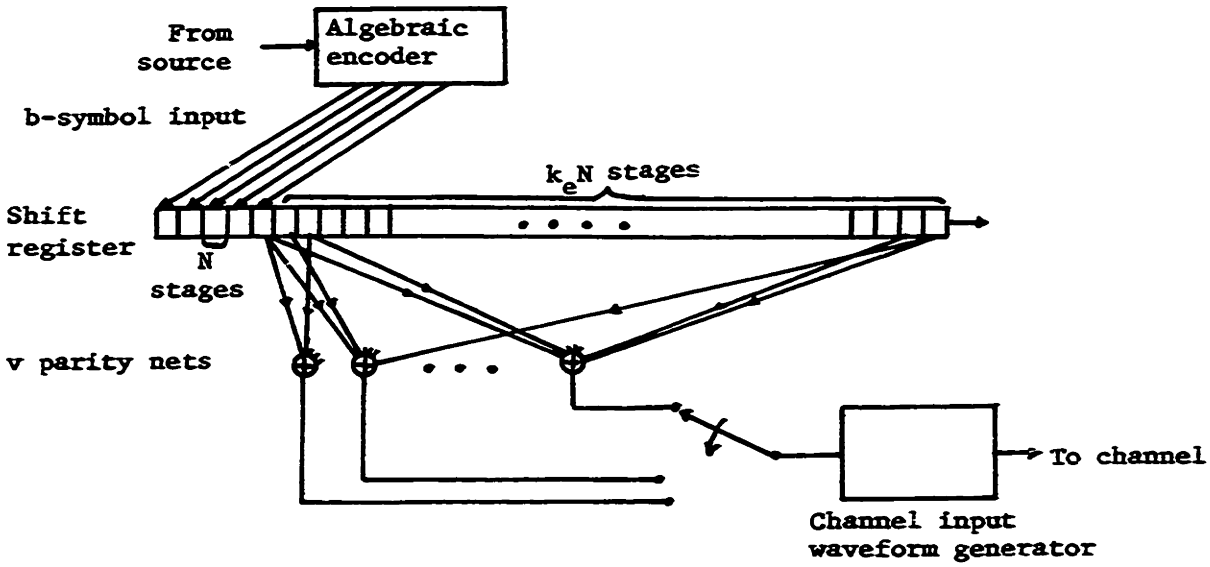

FIG. 6-1 ENCODER CONFIGURATION

6.2 Implementation of the Hybrid Decoder

Two possible decoder configurations come to mind. The first involves essentially the time-shared use of one sequential decoder. A single logic unit, incorporating the Fano algorithm, is allocated to one sequential decoder after another in a round-robin fashion. Separate, fast-access storage must be provided for each decoder's current path metric, threshold, flags, position in tree, etc. This same approach was employed in the simulation.

The major cost is due to the fast-access buffer storage which must be provided for the received branches and hypotheses for each sequential decoder. As in the encoder configuration, this may be realized as an NB-stage shift register which is accessible only at every N^{th} stage. (Each stage stores a received branch and a hypothesized information symbolsand B stages are allocated to each sequential decoder.) Then the total storage is proportional to N, while the number of input/output connections is essentially independent of N.

At any instant, all the latest B received branches and hypotheses for one sequential decoder are accessible to the Fano algorithm and to the algebraic decoder's assistance. The algorithm switches to a new decoder with the arrival of a new branch from the channel. The algebraic decoder is activated after each "cycle" through all N sequential decoders. Any sequential decoder's hypothesis which disagrees with the algebraic decoder's decisions is corrected when the sequential decoder next becomes accessible. The buffer storage arrangement is similar to that shown in fig. (2-6) of Chapter 2, the only difference being that the B input/output taps are at intervals of N, rather than one, stages.

As a further feature, idle time, during which a decoder is waiting for new branches to arrive from the channel, could be cut to a minimum by switching to the next decoder as soon as the waiting line of any decoder drops to zero. In this way, the available computation facilities are shared only among those decoders with non-zero waiting lines. If this feature is used, the first N stages of the BN-stage received branch register must be accessible from the channel to allow the BN-stage register to be shifted several times between received branches, while maintaining the proper order among the received branches.

This configuration of the N sequential decoders seems preferrable when the circuitry is fast enough, relative to the incoming data rate, that the number of computations that can be done between two successive received branches exceeds the average computation per branch by a factor of more than five or ten. If this is not possible, because of limitations on the speed of the logic or memory unit*, a second approach is preferrable.

The second approach to the implementation of the decoder is conceptually simple, but its cost would be at least N times the cost of a single sequential decoder. It incorporates N separate sequential decoders, each with its own logic unit and buffer storage of size B. Note that if the data rate over the channel is fixed, the rate of arriving branches into each sequential decoder is proportional to 1/N. Therefore, if N can be made arbitrarily

*This will typically be the case when the convolutional code rate r is close to channel capacity.

large, there is no longer a restriction imposed by the speed of the circuitry. (Although the cost increases at least linearly with N.)

The implementation of a suitable algebraic decoder should be straightforward. Explicit algorithms and decoder design have been considered by Peterson[8], Gorenstein and Zierler[39], Forney[11], and Berlekamp[40]. The required storage, apart from the N b-symbols per block stored in the sequential decoder's storage buffers, is proportional to d, the minimum distance. If only erasures are to be corrected, the number of computations per block is proportional to d^2. If some errors are to be corrected, it may be proportional to d^3.

6.3 A Design Philosophy

The selection of system parameters which maximize performance and minimize cost is an important problem for the designer of a communication system employing coding. The choice of a modulation/demodulation system (which determines an equivalent DMC) and the specification of the parameters of a sequential decoding scheme are interrelated problems which have been discussed by others.[31,33,41] In introducing the hybrid decoding scheme, we further complicate the design problem by adding the parameter of algebraic code block length N and rate R. In this section, we suggest an approach to the selection of reasonable parameters for the hybrid scheme for a given DMC. The design philosophy described here is oversimplified, it should be considered as merely a guideline to more elaborate design procedures based on extensive simulation.

We suppose that the parameters to be specified are μ, r, $\alpha(r)$, R, N, B, k_e, and k_d, all of which have been defined in Chapters 2 and 3. Let us consider them in the above order.

(a) The Machine Speed Parameter μ

This parameter, which is the number of computations that can be done between received branches, is determined by the ratio of the maximum operating speed of the available components to the number of channel uses per second. (We assume the decoder configuration is the first of the two described in the previous section.)

(b) The Convolutional Code Rate r, and Pareto Exponent $\alpha(r)$

The machine speed parameter μ should exceed the average number

of computations per branch by at least a factor of five or ten. The average computation strongly depends on $\alpha(r)$ and therefore on r. Through simulation, it should be possible to specify an appropriate value of r, and therefore of $\alpha(r)$, to yield a reasonable average computation relative to the machine speed.

(c) The Algebraic Code Rate R

We assume that the algebraic code is Reed-Solomon, and that only erasures are to be corrected. The curves of Chapter 4 suggest that for a fixed effective pareto exponent, the overall rate does not vary much over a fairly broad range of values of R. It is therefore suggested that an appropriate value of R (and perhaps of r) be selected by trial-and-error, to yield the best tradeoff between system cost and overall information rate.

(d) The Block Length N, and Buffer Size B

The simulation showed that if the machine speed parameter μ exceeds the average computational load by a factor of five or more, successive waiting line peaks tend to be disjoint and appear to occur statistically independently. In this case, $p_1(B)$, the probability of buffer overflow while decoding the first block is a convenient and realistic measure of performance. Suppose that values of $\alpha(r)$ and R have been tentatively selected. Then our problem is equivalent to the minimization of $p_1(B)$ for a fixed total amount of storage. The total storage will be taken as $S_t = BN$.

Note that $p_1(B)$ may be overbounded directly, using inequality (3.5.4.), as

$$p_1(B) \leq pr(W_1 \geq B)$$

$$\leq pr(T_1 \geq \mu(B - \Gamma + b))$$

$$= pr(T_1 \geq \mu(B - k_d)) \qquad (6.3.1.)$$

Then using the principal result of section 3.4, (inequality (3.3.11.)), we get

$$p_1(B) \leq \left[\frac{\mu(B-k_d)}{A}\right]^{-N\delta\alpha} \qquad (6.3.2.)$$

where $$A = 2(k_d+b)^{\alpha'} A_0^{\frac{1}{\alpha}} \exp\left[\frac{H(\delta)}{\alpha\delta}\right]$$

$$\alpha = \alpha(r)$$

$$\alpha' = \max(1, \alpha)$$

$$\delta = (1-R) + \frac{1}{N} \simeq 1-R \quad \text{for large } N.$$

$$A_0 = \text{a constant}$$

The results of the simulation suggested that the statistics of computation (and therefore, presumably, the probability of buffer overflow) are relatively insensitive to the decoding constraint length k_d. Let us therefore take the liberty of neglecting k_d and b, re-writing (6.3.2.) as

$$p_1(B) \simeq \left(\frac{\mu B}{A}\right)^{-N\delta\alpha} \qquad (6.3.3.)$$

$$\text{where} \qquad A \approx \quad A_0^{\frac{1}{\alpha}} \; exp\left[\frac{H(\delta)}{\alpha\delta}\right],$$

in the hope that the resulting inaccuracy, if any, is not serious. By using the Lagrange multiplier technique, we may now minimize the right hand side of (6.3.3.) with respect to B and N, keeping BN fixed. The constraint may be written as

$$\lambda\,(BN - S_+) = 0$$

for any λ. We differentiate $p_1(B) + \lambda(BN - S_+)$ with respect to B and N, and set the respective partial derivatives to zero. The resulting solutions for B and N are readily shown to yield a minimum.

$$\frac{\partial}{\partial B}\left[\, p_1(B) + \lambda\,(BN - S_+)\right]$$

$$= -N\delta\alpha\left(\frac{\mu B}{A}\right)^{-N\delta\alpha}\frac{1}{B} + \lambda N = 0 \qquad (6.3.4.)$$

and

$$\frac{\partial}{\partial N}\left[\, p_1(B) + \lambda\,(BN - S_+)\right]$$

$$= -\delta\alpha\left(\frac{\mu B}{A}\right)^{-N\delta\alpha} ln\left(\frac{\mu B}{A}\right) + \lambda B = 0 \qquad (6.3.5.)$$

Solving (6.3.4.) and (6.3.5.) we find that λ and N are eliminated,

and

$$\ln\left(\frac{\mu B}{A}\right) = 1 \qquad (6.3.6.)$$

or

$$B = \frac{Ae}{\mu} = \frac{A_o}{\mu}^{\frac{1}{\alpha}} \exp\left[1 + \frac{H(\delta)}{\alpha\delta}\right]$$

Thus, the minimizing value of B in this simplified analysis is independent of N or S_t, assuming it is less than S_t. (A resultant value of B greater than S_t might indicate that a single pure sequential decoder would make better use of the available storage S_t.) Substituting (6.3.6.) into (6.3.3.) we get

$$P_1(B)\Big|_{opt} \approx e^{-N\delta\alpha}$$

$$= \exp\left[-\frac{\delta\alpha}{B_o} S_t\right]$$

where $\delta \simeq 1-R$

and B_o is the solution of (6.3.6.). Thus, $P_1(B)\big|_{opt}$ decreases exponentially in the total storage S_t.

Equation (6.3.6.) may be written as

$$\left(\frac{A}{\mu B}\right)^{-\alpha} = e^{-\alpha}$$

or

$$\frac{A_o}{(\mu B)^{\alpha}} = \exp\left[-\alpha - \frac{H(\delta)}{\delta}\right] \qquad (6.3.7.)$$

The constant A_0 is the coefficient in the pareto distribution of computation. It can be seen from the experimental curves of fig. 5-4 that it is on the order of unity. This is also confirmed by previous estimates of its value[31]. If α is less than one, and $\delta = 0.1$,

$$\exp\left[-\alpha - \frac{H(\delta)}{\delta}\right] \simeq 10^{-2}$$

The expression on the left-hand side of (6.3.7.) is proportional to the probability that a buffer of size B in a single, unaided sequential decoder overflows while decoding a branch. Thus, this very rough reasoning seems to indicate that optimal allocation of total storage is achieved by fixing B to yield a single-decoder buffer overflow probability on the order of 10^{-2} or 10^{-3}, and then setting $N = S_t/B$. The buffer size B should, of course, be made at least as large as the decoding constraint length k_d. As stated before, the ultimate justification for this procedure would require more extensive simulation.

(e) The Constraint Lengths k_e and k_d

The constraint lengths k_e and k_d determine the probability of error. The simulation indicated that variation of k_e and k_d within a reasonable range should not greatly affect the statistics of computation. Thus their values may be chosen independently of the other system parameters to yield a given tolerable error probability, with the aid of the upper bounds quoted in Chapter 2 or experimental search depth histograms such as those of fig. 5-9.

CHAPTER 7

CONCLUSIONS AND SUGGESTIONS FOR FUTURE RESEARCH

7.1 Applicability of the Hybrid Scheme and Summary of its Properties

The variability of computation and the buffer overflow probability for conventional sequential decoding in conjunction with a given discrete memoryless channel are much more sensitive to the information rate than to the system cost and complexity. The implication is that if highly reliable communication is desired, a reasonable trade-off between system performance and complexity may only be possible at an information rate which is considerably below the channel capacity.

We have proposed a hybrid decoding scheme, in which conventional sequential decoding is supplemented by algebraic decoding. The new hybrid scheme involves N essentially parallel sequential encoding/decoding systems; the most dramatic improvement over conventional sequential decoding is obtained when N is relatively large, say on the order of 50 or 100. Thus, the hybrid scheme may be one or two orders of magnitude more complex than present-day sequential decoding systems. However, the improvement in computational performance due to a given system complexity increment can be considerably greater for the hybrid scheme than for conventional sequential decoding. Thus, the hybrid scheme may find application where extremely reliable communication is required at information rates which are between R_{comp} and channel capacity. It is also well to remember that systems meeting such requirements are becoming increasingly feasible due to the rapid progress of digital component technology.

The decoding time per block was shown in Chapter 3 to be a random variable whose asymptotic distribution is upper-bounded by a pareto distribution with an effective pareto exponent of $d\alpha$, where α is the pareto exponent for pure sequential decoding and d is the algebraic code's minimum distance. This result was substantiated by simulation. Using this result, it was shown that for a sufficiently large block length N, of the algebraic code, the average computation is bounded by a constant at any information rate less than capacity. The average computation was shown analytically, and by simulation, to increase rapidly as the information rate approached channel capacity. It was also shown that if a fixed amount of buffer storage is allotted to each decoder, the probability of buffer overflow decreases essentially exponentially with the <u>total</u> buffer storage at the decoder. It was suggested that best use would be made of a fixed total amount of storage NB, by making N, the number of sequential decoders, large and B, the amount of storage allotted to each, relatively small.

7.2 Remarks on Multi-Stage Encoding/Decoding Schemes

Ziv[9], Pinsker[10], and Forney[11] have recently proposed interesting coding schemes for use with discrete memoryless channels. The common feature of these schemes is that they incorporate two or more separate stages of encoding and decoding, including an "inner" stage to correct most errors introduced by the channel, and an "outer" stage which corrects the few errors made by the inner stage. In all three schemes the inner stage is an arbitrary block coding and decoding scheme. In Ziv's scheme, there is an intermediate stage in which errors made by the inner stage are detected, and then treated as erasures. After a scrambling/de-scrambling procedure, these erasures may be corrected by the outer stage, also a block encoding/decoding scheme. Forney's scheme is somehwat similar except that there is no intermediate stage: a large-alphabet Reed-Solomon code outer stage corrects errors, and perhaps erasures, made by the inner stage. Pinsker's scheme utilizes sequential encoding/decoding for the outer stage. The principle is that if the inner stage has sufficiently low error probability, the rate R_{comp}, seen by the sequential decoder, is little different from channel capacity. (This is, in a sense, the inverse of our hybrid scheme.)

These schemes can operate at any rate up to channel capacity, with error probabilities which decrease exponentially with encoding complexity (exponentially with the square root of encoding complexity in Ziv's scheme), while the outer decoder's complexity increases no faster than algebraically with encoder complexity. The minimum required complexity of the inner decoder has not been fully established; it is certainly low for rates which are well below

channel capacity, but presumably increases rapidly as the rate approaches capacity. These schemes can achieve low error probability with reasonable decoding complexity by tolerating a considerably greater encoding complexity (by perhaps one or two orders of magnitude) than that which the coding theorems show to be sufficient.

The hybrid sequential and algebraic decoding scheme falls into the category of such multi-stage schemes. It differs from the three previously mentioned schemes in that the inner stage employs sequential encoding and decoding, which is known to yield exponentially small error probability for rates up to channel capacity. Furthermore, there is feedback from the outer to the inner decoding stage: the decreased variability of computation of the inner (sequential) decoder, for all rates up to channel capacity, derives from the verification and correction of its decisions by the outer (algebraic) decoder. The latter has the relatively easy job of correcting exactly d-1 erasures in each block.

7.3 Suggestions for Future Research

(a) Lower Bounds

It is not known whether the interaction of the sequential decoders with the algebraic decoder, described in Chapter 3, is optimal in any sense. Presumably, there are alternative strategies, (for example, the pooling scheme described in section 3.7). It would be interesting, but probably difficult, to obtain general lower bounds on the computation distribution for a class of hybrid sequential and block decoding schemes, such as the bounds obtained for pure sequential decoding by Jacobs and Berlekamp[16]. Attempts to derive such general lower bounds have so far been fruitless. A further benefit of such lower bounds would be estimates of the tightness of the upper bounds obtained in Chapter 3.

(b) Dynamic Behavior of the Hybrid Scheme

Tighter upper bounds for the computational performance of the hybrid scheme could presumably be obtained by an analysis of its dynamic behavior. The dynamic behavior is closely related to the waiting line behavior of sequential decoding. This seems a difficult subject for research because of the statistical dependencies between the decoding of successive branches or groups of branches. The only result obtained was the lower bound on the waiting line distribution for infinite buffer size, derived in Appendix A. A different approach to upper-bounding the computational statistics of the hybrid scheme, which avoids consideration of the waiting line behavior would be valuable.

(c) Generalizations to Other Classes of Channels

A modified hybrid scheme may be applicable to channels other than the DMC, such as channels with randomly-varying phase angle or fading, dispersive channels. Studies of such applications will no doubt require extensive simulation.

APPENDIX A

ASYMPTOTIC WAITING LINE DISTRIBUTION FOR A SEQUENTIAL DECODER WITH INFINITE BUFFER SIZE

Suppose the incoming stream of symbols arriving from the channel is divided into blocks, each consisting of N_0 adjacent channel output symbols. (N_0 will be specified later.) We want to lower-bound $pr(w > X)$, which is the cumulative probability distribution for the waiting line existing just after all computations have been completed on some particular block, which we will call block #1. We label the i^{th} block previous to block #1 as block #(i+1), (i = 1,2,...). Assume further that decoding has been proceeding long enough so that at least $n = X/N_0$ blocks of length N_0 have arrived already. Then we lower-bound $pr(w \geq X)$ by the following argument:

> If the number of new channel output symbols which arrived during the decoding of block #1 exceeded N_0 by more than X, then w will surely exceed X. Similarly, if the number of channel output symbols which arrived during the decoding of block #(i+1) exceeded $(i+1)N_0$ by more than X, then the waiting line w channot possibly decay to less than X by the time block #1 has

has been decoded. If C_{i+1} is the number of computations to decode block #(i+1), then the number of new channel symbols which arrive while it is being decoded is at least $\frac{1}{\mu} C_{i+1}$. Thus, we lower-bound pr(w $\geq$ X) by the probability of a union of events:

$$pr(w \geq x) \geq pr\left[\frac{C_1}{\mu} \geq X+N_0 \text{ or } \frac{C_2}{\mu} \geq X+2N_0 \text{ or } \cdots \right.$$
$$\left. \cdots \text{or } \frac{C_n}{\mu} \geq X+nN_0 \right] \qquad \text{(A.1.)}$$

We lower bound the computations associated with any block before all its symbols are finally decoded by postulating a genie who informs the sequential decoder whether it is correct whenever it reaches the end of a block. If incorrect, it is told to keep searching in the block. If correct, it is told to proceed further without ever returning. In the absence of errors (which we have assumed to occur with probability zero), the computations and waiting line of the genie-aided decoder underbound those of the unassisted decoder, since many fruitless computations are eliminated by the genie.

Now let us choose N_0 so that

$$N_0[R_s - E_0'(s)] - o(\sqrt{N_0})$$
$$\geq \ln[p(X+nN_0)] \geq (N_0-1)[R_s - E_0'(s)] - o(\sqrt{N_0-1}) \qquad \text{(A.2.)}$$

Note that $\ell n\ \mu(X + nN_0) = \ell n\ 2\mu X$ if $nN_0 = X$. (A.2.) does not conflict with our previous choice of $nN_0 = X$ with $n \geq 1$, provided X is picked large enough. For future reference, we note that

$$N_0 = o(\ell n\ \mu X)$$

(A.2.) means that block #(i) of N_0 channel symbols ($i = i,2,\ldots n$) contains a sequence of N_i channel symbols with the associated distribution of computation lower-bounded by

$$pr(C \geq \mu(X+iN_0)) > [\mu(X+iN_0)]^{-(s+\epsilon)} \exp\left[-o\left(\frac{1}{\epsilon}\right)\right] \qquad \text{(A.3.)}$$

Therefore for future reference we note that

$$pr(C_i \geq \mu(X+iN_0)) \geq [\mu(X+iN_0)]^{-(s+\epsilon)} \exp\left[-o\left(\frac{1}{\epsilon}\right)\right] \qquad \text{(A.4.)}$$

Note also that because the channel is memoryless and because the genie prevents searching into a block before a previous block is decode, the number of computations for successive blocks are statistically independent.

By a theorem in Feller[19] (Bonferroni Inequality, p.100), we have

$$pr(\omega \geq x) \geq \sum_{i=1}^{n} pr[C_i \geq \mu(x+iN_0)] - \sum_{i=2}^{N} \sum_{j=1}^{i-1} pr[C_i \geq \mu(x+iN_0)]\, pr[C_j \geq \mu(x+jN_0)] \quad (A.5.)$$

(A.5.) may be re-written as

$$pr(\omega \geq x) \geq \sum_{i=1}^{n} pr[C_i \geq \mu(x+iN_0)] \times \left[1 - \sum_{j=1}^{i-1} pr[C_j \geq \mu(x+jN_0)]\right] \quad (A.6.)$$

It is given that the average computation per block is less than μN_0 (otherwise a stationary waiting line distribution would not exist). Thus, from Chebysheff's inequality (section 2.6)

$$pr[C_i \geq \mu(x+jN_0)] \leq \frac{\mu N_0}{\mu(x+jN_0)} = \frac{N_0}{x+jN_0} \quad (A.7.)$$

Therefore

$$\sum_{j=1}^{i-1} pr[C_j \geq \mu(x+jN_0)] \leq \sum_{j=1}^{i-1} \frac{N_0}{x+iN_0} \leq \int_0^i \frac{N_0\, dy}{x+yN_0} = \ln \frac{x+iN_0}{x} \quad (A.8.)$$

But

$$i N_o \leq n N_o = X$$

Therefore we have

$$\ln \frac{X+iN_o}{X} \leq \ln \frac{X+nN_o}{X} = \ln 2$$

and

$$1 - \sum_{j=1}^{i-1} pr[C_j \geq \mu(X+jN_o)] > 1 - \ln 2 = .306$$

Thus, we have for (A.6.):

$$pr(\omega \geq X) > .306 \sum_{i=1}^{n} pr[C_i \geq \mu(X+iN_o)]$$

and using (A.4.) we get

$$pr(\omega \geq X) > .306 \sum_{i=1}^{n} [\mu(X+iN_o)]^{-(s+\epsilon)} \exp\left[-o\left(\frac{1}{\epsilon}\right)\right] \qquad \text{(A.9.)}$$

$$> .306 \exp\left[-o\left(\frac{1}{\epsilon}\right)\right] \int_{1}^{n} [\mu(X+yN_o)]^{-(s+\epsilon)} dy \qquad \text{(A.10.)}$$

$$= \frac{.306\ \mu^{-(s+\epsilon)}}{(s+\epsilon-1)\,N_0} \exp\left[-O\left(\frac{1}{\epsilon}\right)\right]\left[(X+N_0)^{-(s-1+\epsilon)} - (X+nN_0)^{-(s-1+\epsilon)}\right] \quad \text{(A.11.)}$$

But since $nN_0 = X$,

$$(X+N_0)^{-(s-1+\epsilon)} - (X+nN_0)^{-(s-1+\epsilon)}$$

$$= (X+N_0)^{-(s-1+\epsilon)} - (2X)^{-(s-1+\epsilon)}$$

$$\gg X^{-(s-1+\epsilon)}\left[\left(1-2^{-(s-1+\epsilon)}\right) - O\left(\frac{\ln \mu X}{X}\right)\right] \quad \text{(A.12.)}$$

The latter inequality arises since we have taken $N_0 = O(\ell n\ X)$.

Therefore, our final result is

$$pr(\omega \gg X) > \mu^{-(s+\epsilon)}\ X^{-(s-1+\epsilon)}\ f(\epsilon, X)$$

where $$f(\epsilon,X) = \frac{.306}{(s-1+\epsilon)} \exp\left[-O\left(\frac{1}{\epsilon}\right)\right] O\left(\frac{1}{\ln \mu X}\right)\left[1-2^{-(s-1+\epsilon)} - O\left(\frac{\ln \mu X}{X}\right)\right] \quad \text{(A.13.)}$$

Thus,

$$f(\epsilon,X) = \exp\left[-O\left(\frac{1}{\epsilon}\right)\right]\left[O\left(\frac{1}{\ln \mu X}\right)\right]$$

For large X and small ϵ, (A.13.) behaves as

$$pr(\omega \gg X) \gtrsim \mu^{-s}\ X^{-(s-1)}$$

Appendix B

Lemmas Used in Bounding $\overline{C^s}$ $(0 < s \leq 1)$

Lemma 1

For a sequence of non-negative random variables $\{Z_i\}$,

$$\overline{\left(\sum_{i=1}^{N} Z_i\right)^s} \leq \sum_{i=1}^{N} \overline{Z_i^s} \qquad \text{for} \quad 0 \leq s \leq 1$$

Proof: By definition,

$$\overline{\left(\sum_{i=1}^{N} Z_i\right)^s} \triangleq \sum_{Z_1} \cdots \sum_{Z_N} p(Z_1, Z_2, \cdots Z_N) \left(\sum_{i=1}^{N} Z_i\right)^s \qquad \text{(B.1.)}$$

We now invoke a well-known inequality[28]. For a sequence of positive numbers $\{x_i\}$,

$$\left(\sum_{i=1}^{N} x_i\right)^s \leq \sum_{i=1}^{N} x_i^s \qquad \text{if} \quad 0 \leq s \leq 1 \qquad \text{(B.2.)}$$

Applying (B.2.) to the inner sum of (B.1.) we get

$$\overline{\left(\sum_{i=1}^{N} Z_i\right)^s} \leq \sum_{i=1}^{N} \sum_{Z_i} Z_i^s \left(\prod_{\substack{j=1 \\ j \neq i}}^{N} \sum_{Z_j} p(Z_1, Z_2, \cdots Z_N)\right)$$

$$= \sum_{i=1}^{N} \sum_{Z_i} Z_i^s \; p(Z_i)$$

$$= \sum_{i=1}^{N} \overline{z_i^s} \qquad \text{Q.E.D.}$$

Lemma 2

$$F(h,\ell) \le \begin{cases} M_\ell^{\,s} \exp\left[-(\ell-h)\vee E_b(\sigma,s) - h\vee E_a(\sigma,s)\right] & (\ell \ge h) \\ M_\ell^{\,s} \exp\left[-(h-\ell)\vee E_c(\sigma,s) - \ell\vee E_a(\sigma,s)\right] & (\ell \le h) \end{cases}$$

Proof: $F(h,\ell)$ is defined by (2.6.19.). We consider separately the cases $\ell \ge h$ and $\ell \le h$.

Let

$$F(h,\ell) = \begin{cases} F_1(h,\ell) & (\ell \ge h) \\ F_2(h,\ell) & (\ell \le h) \end{cases} \qquad \text{(B.3.)}$$

Consider $F_1(h,\ell)$ first. It may be written as

$$F_1(h,\ell) = \sum_{\bar{Y}_{\ell v}} f(\bar{Y}_{hv})^{\sigma s} f(\bar{Y}_{\ell v})^{-\sigma s} \sum_{\bar{X}_{\ell v}} p(\bar{X}_{\ell v})\, p(\bar{Y}_{\ell v} \mid \bar{X}_{\ell v})\, p(\bar{Y}_{hv} \mid \bar{X}_{hv})^{-\sigma s}$$

$$\times \overline{\left[\sum_{n=1}^{M_\ell} p(\bar{Y}_{\ell v} \mid \bar{X}'_{\ell v}(n))^{\sigma} \right]^{s}} \qquad \text{(B.4.)}$$

where we define the conditional expectation

$$\overline{\left[\sum_{n=1}^{M_\ell} p(\bar{y}_{\ell v} \mid \bar{x}_{\ell v}'(n))^{\sigma} \right]^{s}}$$

$$= \sum_{\bar{x}_{\ell v}'(1)} \cdots \sum_{\bar{x}_{\ell v}'(M_\ell)} p\left[\bar{x}_{\ell v}'(1) \cdots \bar{x}_{\ell v}'(M_\ell) \mid \bar{x}_{\ell v} \right] \times \left[\sum_{n=1}^{M_\ell} p(\bar{y}_{\ell v} \mid \bar{x}_{\ell v}'(n))^{\sigma} \right]^{s} \qquad \text{(B.5.)}$$

Now for $0 < s \leq 1$,

$$\left[\sum_{n=1}^{M_\ell} p(\bar{y}_{\ell v} \mid \bar{x}_{\ell v}'(n))^{\sigma} \right]^{s}$$

is a convex upward function of the positive random variable

$$\sum_{n=1}^{M_\ell} p(\bar{y}_{\ell v} \mid \bar{x}_{\ell v}'(n))^{\sigma}$$

Also, since

$$\sum_{\bar{x}_{\ell v}'(1)} \cdots \sum_{\bar{x}_{\ell v}'(M_\ell)} p\left[\bar{x}_{\ell v}'(1) \cdots \bar{x}_{\ell v}'(M_\ell) \mid \bar{x}_{\ell v} \right] = 1$$

we may invoke the well-known[29] inequality for weighted means: for Z a positive random variable and $0 \leq s \leq 1$, $\overline{Z^s} \leq \bar{Z}^s$. Applying this to (B.5.),

we get:

$$\overline{\left[\sum_{n=1}^{M_\ell} p(\bar{Y}_{\ell v} \mid \bar{x}_{\ell v}'(n))^{\sigma}\right]^{s}}$$

$$\leq \left[\sum_{n=1}^{M_\ell}\left[\sum_{\bar{x}_{\ell v}'(1)} \cdots \sum_{\bar{x}_{\ell v}'(M_\ell)} p(\bar{Y}_{\ell v} \mid \bar{x}_{\ell v}'(n))^{\sigma}\, p\left[\bar{x}_{\ell v}'(1) \cdots \bar{x}_{\ell v}'(M_\ell) \mid \bar{x}_{\ell v}\right]\right]\right]^{s}$$

$$= \left[\sum_{n=1}^{M_\ell} \sum_{\bar{x}_{\ell v}'(n)} p(\bar{Y}_{\ell v} \mid \bar{x}_{\ell v}'(n))^{\sigma}\, p(\bar{x}_{\ell v}'(n) \mid \bar{x}_{\ell v})\right]^{s} \qquad \text{(B.6.)}$$

Now since there is pairwise independence between the correct path and any incorrect path in the incorrect subset,

$$p(\bar{x}_{\ell v}'(n) \mid \bar{x}_{\ell v}) = p(\bar{x}_{\ell v}'(n))$$

Furthermore, since the distribution of symbols for all paths in the same, the inner sum in (B.6.) is independent of n. We use these facts to re-write (B.6.) as

$$\overline{\left[\sum_{n=1}^{M_\ell} p(\bar{Y}_{\ell v} \mid \bar{x}_{\ell v}'(n))^{\sigma}\right]^{s}}$$

$$\leq \left[M_\ell \sum_{\bar{x}_{\ell v}'} p(\bar{Y}_{\ell v} \mid \bar{x}_{\ell v})^{\sigma}\right]^{s} \qquad \text{(B.7.)}$$

where we have dropped the variable n.

The sequence of steps leading from (B.5.) to (B.7.) would have been easier if we had assumed statistical independence among <u>all</u> paths of the tree instead of just pairwise statistical independence. Our purpose was to show that the bound applies to the ensemble of randomly-picked infinite constraint length convolutional codes.[6]

Before substituting (B.7.) back into (B.4.), we note that if $\ell \geq h$, then $\bar{y}_{nv}$ and $\bar{x}_{nv}$ are the first hv components of $\bar{y}_{\ell v}$ and $\bar{x}_{\ell v}$ respectively. Furthermore, successive components of any vector are statistically independent, since the channel is memoryless. Thus, for example,

$$p(\bar{y}_{\ell v}) = p(\bar{y}_{hv})\, p(\bar{y}_{\ell v-hv})$$

where $\bar{y}_{\ell v\text{-}hu}$ is the vector representing the last ℓv-hu symbols of $\bar{y}_{\ell v}$. Now we substitute (B.7.) into (B.4.)

$$F_i(h,\ell) \leq \sum_{\bar{y}_{hv}} \sum_{\bar{y}_{\ell v-hv}} f(\bar{y}_{\ell v-hv})^{-\sigma s} \times \left[\sum_{\bar{x}_{\ell v-hv}} p(\bar{x}_{\ell v-hv})\, p(\bar{y}_{\ell v-hv} / \bar{x}_{\ell v-hv}) \sum_{\bar{x}_{hv}} p(\bar{x}_{hv})\, p(\bar{y}_{hv} / \bar{x}_{hv})^{1-\sigma s} \right]$$

$$\times \left[M_\ell \sum_{\bar{x}'_{\ell v-hv}} p(\bar{x}'_{\ell v-hv})\, p(\bar{y}_{\ell v-hv} \mid \bar{x}'_{\ell v-hv})^{\sigma} \sum_{\bar{x}'_{hv}} p(\bar{x}'_{hv})\, p(\bar{y}_{hv} \mid \bar{x}'_{hv})^{\sigma} \right]^{s} \tag{B-8}$$

Using the fact that

$$\sum_{\bar{x}_{\ell v-hv}} p(\bar{x}_{\ell v-hv})\, p(\bar{y}_{\ell v-hv} \mid \bar{x}_{\ell v-hv}) = f(\bar{y}_{\ell v-hv}),$$

We can re-arrange the sums in (B.8.) as follows:

$$F_i(h,\ell) \le M_\ell^{s} \sum_{\bar{y}_{\ell v-hv}} f(\bar{y}_{\ell v-hv})^{1-\sigma s} \left[\sum_{\bar{x}'_{\ell v-hv}} p(\bar{x}'_{\ell v-hv})\, p(\bar{y}_{\ell v-hv} \mid \bar{x}'_{\ell v-hv})^{\sigma} \right]^{s}$$

$$\times \sum_{\bar{y}_{hv}} \left[\sum_{\bar{x}_{hv}} p(\bar{x}_{hv})\, p(\bar{y}_{hv} \mid \bar{x}_{hv})^{1-\sigma s} \right]$$

$$\times \left[\sum_{\bar{x}'_{hv}} p(\bar{x}'_{hv})\, p(\bar{y}_{hv} \mid \bar{x}'_{hv})^{\sigma} \right]^{s} \tag{B.9.}$$

Using the probability measures expressed in (2.6.11.) through (2.6.13.), we can write a sum of the form

$$\sum_{\bar{x}_N} p(\bar{x}_N)\, p(\bar{y}_N \mid \bar{x}_N)^{\sigma}$$

as

$$\sum_{\bar{x}_N} \prod_{k=1}^{N} p(x_k)\, p(y_k|x_k)^{\sigma} \qquad \text{(B.10.)}$$

where the $\{x_k\}$ and $\{y_k\}$ are the individual symbols from the channel input and output alphabets respectively, of the sequences $\bar{x}_N$ and $\bar{y}_N$. The $\{x_k\}$ may be identified with the integers from 1 to P, and $\{y_k\}$ with the integers 1 to Q.

Thus, (B.10.) may be re-written as

$$\sum_{\bar{x}_N} p(\bar{x}_N)\, p(\bar{y}_N|\bar{x}_N)^{\sigma} = \prod_{k=1}^{N} \sum_{i=1}^{P} p(x_k=i)\, p(y_k=j|x_k=i)^{\sigma}$$

$$j = 1,2,\ldots Q \qquad \text{(B.11.)}$$

Because all symbols are chosen independentaly from the same distribution, this may be re-written as

$$\sum_{\bar{x}_N} p(\bar{x}_N)\, p(\bar{y}_N|\bar{x}_N)^{\sigma} = \left[\sum_{i=1}^{P} p_i\, q_{ij}^{\sigma}\right]^{N}$$

where the j's correspond to the sequence of symbols in $\bar{y}_N$.

Using this technique on all sums in (B.9.) we get

$$F_1(h,\ell) \le M_\ell^{\,s}\left[\sum_{j=1}^{Q} f_j^{\,1-\sigma s}\left(\sum_{i=1}^{P} p_i q_{ij}^{\,\sigma}\right)^{s}\right]^{(\ell-h)v}$$

$$\times\left[\sum_{j=1}^{Q}\left(\sum_{i=1}^{P} p_i q_{ij}^{\,1-\sigma s}\right)\left(\sum_{i=1}^{P} p_i q_{ij}^{\,\sigma}\right)^{s}\right]^{hv} \qquad \text{(B.12.)}$$

$$= M_\ell^{\,s}\exp\left[-(\ell-h)v\,E_b(\sigma,s) - hv\,E_a(\sigma,s)\right] \qquad (\ell > h) \qquad \text{(B.13.)}$$

where $E_a(\sigma,s)$ and $E_b(\sigma,s)$ were defined by (2.6.21.) and by (2.6.22.).

For $\ell \le h$,

$$F_2(h,\ell) = \sum_{\bar{Y}_{hv}} f(\bar{Y}_{hv})^{\sigma s} f(\bar{Y}_{\ell v})^{-\sigma s} \sum_{\bar{X}_{hv}} p(\bar{X}_{hv})\, p(\bar{Y}_{hv} \mid \bar{X}_{hv})^{1-\sigma s}$$

$$\times\left[\sum_{n=1}^{M_\ell} p(\bar{Y}_{\ell v} \mid \bar{X}'_{\ell v}(n))^{\sigma}\right]^{s} \qquad \text{(B.14.)}$$

The remainder of the derivation for $F_2(h,\ell)$ closely parallels that for $F_1(h,\ell)$. The final result is

$$F_2(h,\ell) \le M_\ell^{\,s} \exp\left[-(h-\ell)\vee E_c(\sigma,s) - \ell \vee E_a(\sigma,s)\right] \qquad (\ell \le h) \tag{B.15.}$$

Q.E.D.

Lemma 3

For $0 < s \le 1$,

$$E_b(s) \ge \frac{s}{1+s} E_0(s)$$

Proof:

$$E_b(s) = E_b\left(\frac{1}{1+s}, s\right)$$

where $E_b(\sigma,s)$ is defined by (2.6.22.)

Thus $E_b(s)$ is defined by

$$\exp\left[-E_b(s)\right] = \sum_{j=1}^{Q} f_j^{\frac{1}{1+s}} \left(\sum_{i=1}^{P} p_i q_{ij}^{\frac{1}{1+s}} \right)^s \tag{B.16.}$$

We now invoke Holder's inequality[28], which states that for two sequences of positive numbers $\{a_j\}$ and $\{b_j\}$ and some λ for which $0 \le \lambda \le 1$,

$$\sum_j a_j b_j \le \left(\sum_j a_j^{\frac{1}{\lambda}} \right)^{\lambda} \left(\sum_j b_j^{\frac{1}{1-\lambda}} \right)^{1-\lambda} \tag{B.17.}$$

We identify a_j with $f_j^{\frac{1}{1+s}}$, b_j with

$$\left(\sum_{i=1}^{P} p_i q_{ij}^{\frac{1}{1+s}} \right)^s$$

and λ with $\frac{1}{1+s}$.

Then the right hand side of (B.16.) is overbounded by

$$\exp\left[-E_b(s)\right] \leq \left[\sum_{j=1}^{Q} f_j^{\frac{1}{1+s}(1+s)} \right]^{\frac{1}{1+s}} \left[\sum_{j=1}^{Q} \left(\sum_{i=1}^{P} p_i q_{ij}^{\frac{1}{1+s}} \right)^s {}^{\frac{(1+s)}{s}} \right]^{\frac{s}{1+s}}$$

$$= \exp\left[- \frac{s}{1+s} E_o(s) \right] \qquad \text{(B.18.)}$$

since $\sum_{j=1}^{Q} f_j = 1$

Thus

$$E_b(s) \geq \frac{s}{1+s} E_o(s) \qquad \text{Q.E.D.}$$

Lemma 4

$$E_c(s) \geq \frac{1}{1+s} E_o(s)$$

Proof:

From (2.6.23.) with $\sigma = \frac{1}{1+s}$, $E_c(s)$ is defined by

$$\exp\left[-E_c(s)\right] = \sum_{j=1}^{Q} f_j^{\frac{s}{1+s}} \sum_{i=1}^{P} p_i q_{ij}^{\frac{1}{1+s}} \qquad \text{(B.19.)}$$

Using Holder's inequality again, with

$$a_j = f_j^{\frac{s}{1+s}}$$

$$b_j = \sum_{i=1}^{P} p_i q_{ij}^{\frac{1}{1+s}}$$

$$\lambda = \frac{s}{1+s}$$

we get

$$\exp\left[-E_c(s)\right] \leq \left[\sum_{j=1}^{Q} f_j^{\frac{s}{1+s}\cdot\frac{(1+s)}{s}}\right]^{\frac{s}{1+s}} \left[\sum_{j=1}^{Q}\left(\sum_{i=1}^{P} p_i q_{ij}^{\frac{1}{1+s}}\right)^{1+s}\right]^{\frac{1}{1+s}}$$

$$= \exp\left[-\frac{1}{1+s} E_o(s)\right] \qquad \text{(B.20.)}$$

Thus $E_c(s) \geq \frac{1}{1+s} E_0(s)$ Q.E.D.

APPENDIX C

A BRIEF RESUME OF ALGEBRAIC CODING

This appendix is intended to fix terminology and topoint out the main properties of algebraic coding which are directly relevant to the hybrid scheme. Peterson[8] is a comprehensive source book on algebraic coding and decoding. A direct, lucid exposition of algebraic codes, with emphasis on the Reed-Solomon codes, has been given by Forney[11].

Algebraic coding techniques are applicable to channels which have M inputs and M outputs (M + 1 outputs if erasures can occur), where M is a prime or an integer power of a prime. The channel is generally assumed to be symmetric from the input, with equal cross-over transition probabilities. Maximum-likelihood decoding for such channels is equivalent to decoding a received code word into the possible transmitted word which differs from it in the least number of symbols. In this report, the "algebraic" channel which generally concerns us is the M-ary ersure channel ($M = u^b$), in which all cross-over probabilities are zero.

Since the number of channel input letters is a power of a prime, the letters may be identified with members of a galois field, for which the operations of mod-M addition and multiplication are defined. A code with block length N is viewed as a set of N--tuples of field elements. The code has an underlying algebraic structure. An important property of an algebraic code is its <u>minimum distance</u> d, which is the minimum number of places

in which any two code words differ. Practical decoding techniques are available for certain classes of algebraic codes with specified minimum distance properties. These decoding techniques consist of a finite number of algebraic (galois field) operations which guarantee the correction of up to t errors and s erasures for any s and t such that

$$2t + s < d$$

With a few exceptions[40], no simple ways are known of correcting larger numbers of errors and erasures for a given d.

It is easy to show that no more than d-1 erasures are correctable. For, suppose that a word containing d erasures is received. This is decodable without ambiguity only if there is a unique code word which matches the received word in the N-d unerased places. This is only possible if there is a code word which differs from all other code words in d + 1 places; since the minimum distance is d, there is no such code word and hence d or more erasures cannot be corrected without ambiguity.

Algebraic codes whose dimensionless rate is R = k/N can be put in systematic form, in which the first k symbols of each code word are information symbols, and the remaining N-k symbols are check symbols (linear combinations, mod-M, of the information symbols). Each information symbol can be considered to be supplied by an M-ary source, the code comprises $M^k = M^{RN}$ code words.

A pair of code words may differ in as little as one information symbol. Thus, the minimum distance d of an algebraic code can be no

greater than the number of check symbols plus one, i.e.

$$d \leq (1 - R)\, N + 1$$

For interesting values of N, this upper bound on d may be achieved by Reed-Solomon codes.

A Reed-Solomon code with any dimensionless rate R may be defined for an M-ary channel, provided the block length N is less than M-1. The minimum distance of Reed-Solomon codes is the largest possible, $(1-R)N + 1$. The algebraic formulation of Reed-Solomon codes, and implementation of encoding and decoding operations have been described elsewhere[42,30,11]. It can be shown that the number of decoding operations to correct errors and erasures is roughly proportional to d^3 and d^2, respectively.

REFERENCES

1. C.E. Shannon, "A Mathematical Theory of Communication", Bell System Tech. J. 27, 379 (1948); and 27, 623 (1948).

2. R.M. Fano, Transmission of Information (M.I.T. Press, Cambridge and Wiley, New York, 1961).

3. R.G. Gallager, "A Simple Derivation of the Coding Theorem and Some Applications", IEEE Trans. on Information Theory, IT-11, No. 1, 3-18 (1965).

4. A.J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm", to be pub. IEEE Trans. on Information Theory.

5. R.G. Gallager, Low Density Parity Check Codes (M.I.T. Press, Cambridge and Wiley, New York, 1962).

6. J.M. Wozencraft and B. Reiffen, Sequential Decoding (M.I.T. Press, Cambridge and Wiley, New York, 1961).

7. R.M. Fano, "A Heuristic Discussion of Probabilistic Decoding", IEEE Trans. on Information Theory IT-9, 64-74 (April, 1963).

8. W.W. Peterson, Error-Correcting Codes (Wiley, New York, 1961).

9. J. Ziv, "A New Efficient Coding and Decoding Scheme for Memoryless Channels", to be published.

10. M.S. Pinsker, "Complexity of the Decoding Process", Probl. Peredachi Informatsii, 1, No. 1, 113-116, (1965).

11. G.D. Forney, "Concatenated Codes", Sc.D. thesis, Dept. of Electrical Engineering, M.I.T., Cambridge, Mass. (June, 1965), also R.L.E. Technical Report No. 440, M.I.T., (Dec.,1965).

12. R.G. Gallager, C.E. Shannon, and E.R. Berlekamp, "Lower Bounds to Error Probability for Coding on Discrete Memoryless Channels", to be published.

13. J.M. Wozencraft, "Sequential Decoding for Reliable Communications", Sc. D. thesis, Dept. of Electrical Engineering, M.I.T., Cambridge, Mass. (June, 1957).

14. B. Reiffen, "Sequential Encoding and Decoding for the Discrete Memoryless Channel", R.L.E. Technical Report No. 374, M.I.T., (August, 1960).

15. J. Ziv, "Coding and Decoding for Time-Discrete Amplitude-Continuous Memoryless Channels", Sc.D.thesis, Dept.of Electrical Engineering, M.I.T. Cambridge, Mass. (February, 1962).

16. I.M. Jacobs and E.R. Berlekamp, "A Lower Bound to the Distribution of Computation for Sequential Decoding", to be pub. IEEE Trans. on Information Theory.

17. D.V. Lindley, "Theory of Queues with a Single Server", Proc. Cambridge Phil. Soc., 48, 277, (1952).

18. R.J. Gladstone, "Queues in Channels Employing Error Detection and Retransmission", E.E.thesis, Dept. of Electrical Engineering, M.I.T. Cambridge, Mass. (June, 1963).

19. W. Feller, An Introduction to Probability Theory and its Applications, I, 2nd ed.,(Wiley, New York, 1957).

20. J.M. Wozencraft and I.M. Jacobs, Principles of Communication Engineering, (Wiley, New York, 1965).

21. J.M. Wozencraft and M. Horstein, "Coding for Two-Way Channels", Proc. 4th London Symp. on Information Theory, C. Cherry (ed.), (Butterworth, London, 1961).

22. I. Stiglitz, "Sequential Decoding with Feedback", Ph.D.thesis, Dept.of Electrical Engineering, M.I.T. Cambridge, Mass. (August, 1963).

23. H.L. Yudkin, "Channel State Testing in Information Decoding", Sc.D.thesis, Dept. Electrical Engineering, M.I.T. Cambridge, Mass. (Sept., 1964). These results apply to finite state channels, of which the DMC is a special case.

24. R. Gray, "Simulation of Sequential Decoding with Decision-Directed Channel Measurement", S.M.thesis, Dept. Electrical Engineering, M.I.T. Cambridge, Mass. (June, 1966).

25. D. Haccoun, "Simulated Communication with Sequential Decoding and Phase Estimation", S.M.thesis, Dept. Electrical Engineering, M.I.T. Cambridge, Mass. (Sept., 1966).

26. J.E. Savage, "The Computation Problem with Sequential Decoding", Ph.D.thesis, Dept. Electrical Engineering, M.I.T. Cambridge, Mass. (Feb., 1965), also R.L.E. Tech. Report No.439, M.I.T. (Feb., 1965).

27. H.L. Yudkin, private communication, (Sept., 1965).

28. G.H. Hardy, J.E. Littlewood and G. Polya, Inequalities,(Cambridge Un. Press, London, 1959).

29. D. Blackwell and M.A. Girshick, Theory of Games and Statistical Decisions, (Wiley, New York, 1954).

30. K. Jordan and G. Bluestein, "An Investigation of the Fano Sequential Decoding Algorithm by Computer Simulation", Group Report 62G-3, Lincoln Laboratory, M.I.T. (July, 1963).

31. K. Jordan, "The Performance of Sequential Decoding in Conjunction with Efficient Modulation", IEEE Trans. on Communication Technology, COM 14, No. 3, (June, 1966).

32. D.D. Falconer and C.W. Niessen, "Simulation of Sequential Decoding for a Telemetry Channel", R.L.E. Quarterly Prog. Report No.80, 183-193, (Jan., 1966).

33. I.M. Jacobs, "Performance Parameters for Sequential Decoding", J.P.L. Space Programs Summary No. 37-32, 303-307, (1965); also "Probabilities of Overflow and Error in Coded Phase-Shift-Keyed Systems with Sequential Decoding", J.P.L. Space Programs Summary No. 37-33, 296-299, (1965).

34. P.R. Drouilhet, Jr., "The Lincoln Experimental Terminal Signal Processing System", IEEE Communications Convention Record, Boulder, Colorado (June, 1965).

35. R.S. Kennedy, Performance Limitations of Fading Dispersive Channels, monograph to be published.

36. M.A. Epstein, "Algebraic Decoding for a Binary Erasure Channel", R.L.E. Tech. Report 340, M.I.T. (March 14, 1958).

37. C.W. Niessen, "An Experimental Facility for Sequential Decoding", Sc.D.thesis, Dept. Electrical Engineering, M.I.T., Cambridge, Mass. (Sept., 1965); also R.L.E. Tech. Report No. 450, (Sept., 1965).

38. B.F. Green, Jr., J.E.K. Smith and L. Klem, "Empirical Tests of an Additive Random Number Generator", J. Assoc. Compt. Mach. 6, 527-537, (March, 1959).

39. D. Gorenstein and N. Zierler, "A Class of Cyclic Linear Error-Correcting Codes in p^m Symbols", J.SIAM, 9, 207 (1961).

40. E.R. Berlekamp, "Practical BCH Decoders", to be published.

41. J.M. Wozencraft and R.S. Kennedy, "Modulation and Demodulation for Probabilistic Decoding", IEEE Trans. on Information Theory, IT-12, No.3, 291-297 (July, 1966).

42. I.S. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields", J. SIAM, 8, 300 (1960).

BIOGRAPHICAL NOTE

David Duncan Falconer was born on August 15, 1940 in Moose Jaw, Saskatchewan, Canada. He attended elementary and high schools in Toronto, Canada from 1945 to 1958. In May, 1962, he graduated with a B.A.Sc. degree in Engineering Physics from the University of Toronto. He received the S.M. degree in Electrical Engineering from M.I.T. in September, 1963.

Mr. Falconer was awarded the Ontario Association of Professional Engineers Gold Medal in 1962. He held an M.I.T. Whitney fellowship in 1962-63, a research assistantship in the Research Laboratory of Electronics from 1963 to 1965, and a Hughes Aircraft Fellowship in R.L.E. during 1965-66. He is a member of Tau Beta Pi and an associate member of Sigma Xi.

Mr. Falconer was employed during the summers of 1961 and 1962 by the Defence Research Telecommunications Establishment in Ottawa, Canada, and during the summer of 1964 by Bell Telephone Laboratories in Holmdel, New Jersey.

In 1965 he married the former Marcia Dreisbach of Allentown, Pennsylvania. They have been residing in Watertown, Massachusetts.